Chapter 1

High-Dimensional Linear Models

1.1 Introduction

High-dimensional linear regression is a framework for analyzing datasets where the number of predictors (p) is large relative to the number of observations (n). This arises in empirical applications such as growth analysis and health records, where numerous regressors are present. It also appears when transformations are applied to the data to create a dictionary of features. The linear regression model is given as:

$$Y = X\beta + e, \quad e \perp X, \tag{1.1}$$

where the dimension of X is p.

1.2 Over-fitting in High Dimension

When the number of predictors p is large relative to n, over-fitting becomes a critical challenge. OLS, by design, seeks the best in-sample fitting in terms of the sum of the squared residuals (SSR). The goal of regression is to accurately fit the signal; yet, with too many regressors, the noise is also fitted, leading to poor out-of-sample prediction.

If the data is generated by the linear model (1.1) with the true coefficient β_0, then in the population

$$\min_{\beta} E[(y_i - x_i'\beta)^2] = E[(y_i - x_i'\beta_0)^2] = E[e_i^2] = \sigma^2.$$

This is the minimal error that can be achieved in the population. In reality, we have a sample (Y, X) of n observations, and we estimate β by the OLS estimator $\hat{\beta} = (X'X)^{-1}X'y$. The expectation of the SSR is

$$E\left[\frac{1}{n}(Y - X\widehat{\beta})'(Y - X\widehat{\beta})\right] = \frac{1}{n}E\left[e'(I_n - X(X'X)^{-1}X)e\right] = \frac{\sigma^2}{n}(n-p) = \sigma^2\left(1 - \frac{p}{n}\right) < \sigma^2$$

If $p/n \to c \in (0,1)$, then the expected SSR is strictly smaller than the minimal population risk. The model is overfitted.

The in-sample overfitting has severe consequences for out-of-sample (OOS) prediction. Suppose (y^*, x^*) is a single new observation from the test data independent of the training data. The OOS risk is

$$\begin{aligned}\mathbb{E}[(y^* - x^{*\prime}\hat{\beta})^2] &= \mathbb{E}[(x^{*\prime}\beta_0 + e^* - x^{*\prime}\hat{\beta})^2] \\ &= \mathbb{E}[(x^{*\prime}(\hat{\beta} - \beta_0) - e^*)^2] \\ &= \sigma^2 + \mathbb{E}[(x^{*\prime}(\hat{\beta} - \beta_0))^2].\end{aligned}$$

Since σ^2 is irreducible by any method, we focus on the second term

$$\begin{aligned}\mathbb{E}[(x^{*\prime}(\hat{\beta} - \beta_0))^2] =& \mathbb{E}[x^{*\prime}(\hat{\beta} - \beta_0)(\hat{\beta} - \beta_0)^\prime x^*] \\ =& \text{trace}\left(\mathbb{E}[x^* x^{*\prime}(\hat{\beta} - \beta_0)(\hat{\beta} - \beta_0)^\prime]\right) \\ =& \text{trace}\left(\mathbb{E}[x^* x^{*\prime}]\mathbb{E}[(\hat{\beta} - \beta_0)(\hat{\beta} - \beta_0)^\prime]\right) \\ =& \text{trace}\left(\Sigma_x \mathbb{E}[(X^\prime X)^{-1} X^\prime e e^\prime X (X^\prime X)^{-1}]\right) \\ =& \frac{\sigma^2}{n} \text{trace}\left(\mathbb{E}\left[\left(\Sigma_x^{-\frac{1}{2}} \frac{X^\prime X}{n} \Sigma_x^{-\frac{1}{2}}\right)^{-1}\right]\right),\end{aligned}$$

which depends on the distribution of x_i. Consider a special case $X_i \sim \Sigma_x^{1/2} \cdot \text{Normal}(0, I_p)$, then

$$\Sigma_x^{-1/2} X^\prime X \Sigma_x^{-1/2} \sim \text{Wishart}(p, n, I_p)$$

and the OOS risk can be evaluated by the **Random Matrix Theory**. It is known if $p/n \to c \in (0, 1)$, then

$$\mathbb{E}[(y^* - x^{*\prime}\hat{\beta})^2] = \sigma^2 O(1)$$

and this $O(1)$ can be larger than 1. The worst case happens when $p/n \to 1$, where minimal eigenvalue of the Wishart matrix converges to 0.

1.3 Ridge Regression

Regularization techniques, such as Ridge Regression and Lasso, are employed to mitigate over-fitting by introducing penalties on the model's complexity. They are statistical shrinkage methods.

The Ridge Regression objective function is expressed as:

$$\min_\beta \frac{1}{n}\|Y - X\beta\|_2^2 + \lambda\|\beta\|_2^2,$$

where $\lambda > 0$ is a tuning parameter controlling the penalty. It can also be written as:

$$\min_\beta \frac{1}{n}(Y - X\beta)^\prime(Y - X\beta) + \lambda\beta^\prime\beta,$$

for which the first-order conditions are:

$$-\frac{2}{n}X^\prime(Y - X\beta) + 2\lambda\beta = 0.$$

Rearranging the first-order conditions yields

$$\left(\frac{X'X}{n} + \lambda I_p\right)\beta = \frac{X'Y}{n},$$

which explicitly solves the coefficient

$$\hat{\beta} = \left(\frac{X'X}{n} + \lambda I_p\right)^{-1}\frac{X'Y}{n}.$$

The only difference of the ridge estimator from the OLS is attaching a ridge λI_p to the Gram matrix $X'X/n$.

The effect of the ridge can be better understood if we diagonalize the Gram matrix as

$$\frac{X'X}{n} = UDU',$$

where U is an orthonormal matrix of eigenvectors, and $D = \text{diag}(d_1, d_2, \ldots, d_p)$ is a diagonal matrix of eigenvalues in descending order (assuming $p < n$). The ridge enters the Gram matrix as

$$(X'X + \lambda I) = U(D + \lambda I)U' = U\begin{bmatrix} d_1+\lambda & 0 & \cdots & 0 \\ 0 & d_2+\lambda & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_p+\lambda \end{bmatrix}U'.$$

The ridge penalty shifts each eigenvalue to the right by λ to prevent it from being too close to 0, and thus ensures the invertibility

$$(X'X + \lambda I)^{-1} = U\begin{bmatrix} \frac{1}{d_1+\lambda} & 0 & \cdots & 0 \\ 0 & \frac{1}{d_2+\lambda} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{d_p+\lambda} \end{bmatrix}U'$$

even if some eigenvalues are 0.

Ridge Regression is asymptotically equivalent to OLS if p is fixed as $n \to \infty$ under the assumption $\liminf_{n\to\infty} d_p > 0$ and $\lambda = \lambda_n \to 0$. The ridge estimator maintains consistency and asymptotic normality. High-dimensional analysis ($p/n \to$ constant) is much more challenging and needs advanced mathematical tools.

1.4 Lasso

Economic theory is usually vague about which variables to include in a regression. "Betting on sparsity" is one way to reduce the complexity of high-dimensional linear models. Lasso introduces sparsity by penalizing the L_1-norm of β. The objective function is:

$$\min_{\beta} \frac{1}{2n}\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1,$$

where $\|\beta\|_1 = \sum_{i=1}^p |\beta_i|$ is the L_1-norm of a vector.

To understand the mechanism of inducing sparsity, notice that the absolute value function $|u|$ for $u \in \mathbb{R}$ is non-differentiable only at $u = 0$. Therefore, we introduce the *subgradient* of $|u|$ as

$$\frac{\mathrm{d}|u|}{\mathrm{d}u} = \begin{cases} 1, & \text{if } u > 0, \\ -1, & \text{if } u < 0, \\ \text{any value in } (-1, 1), & \text{if } u = 0. \end{cases}$$

Heuristically, the first-order condition with respect to β_j must satisfy

$$-\frac{1}{n} X_j'(Y - X\hat{\beta}) + \lambda \frac{\mathrm{d}|u|}{\mathrm{d}u}\bigg|_{u=\hat{\beta}_j} = 0.$$

If $\hat{\beta}_j > 0$ or $\hat{\beta}_j < 0$, then:

$$X_j'(Y - X\hat{\beta})/n = \lambda \operatorname{Sign}(\hat{\beta}_j).$$

Otherwise $\hat{\beta}_j = 0$ and then

$$\left| X_j'(Y - X\hat{\beta})/n \right| < \lambda$$

since the first-order condition must hold for any value of the subgradient. This inequality means that a local perturbation of $\hat{\beta}_j$ cannot compensate for the penalty brought by λ. Minimization of the Lasso objective function coerces $\hat{\beta}_j$ to stay at 0. The above two expressions make the **Karush-Kuhn-Tucker condition** for Lasso.

Another way to understand the sparsity estimation of Lasso is by writing the optimization as

$$\min_{\beta} \frac{1}{2n} \|Y - X\beta\|_2^2 \quad \text{s.t. } \|\beta\|_1 \le C,$$

where there is a one-to-one relationship between the tuning parameter C in the constrained optimization and λ in the penalized optimization. Depending on the shape of the contour, corner solutions are likely to occur. (I will draw a diagram.)

1.4.1 Feature Engineering

Lasso has many variants. Consider

$$(2n)^{-1}(Y - X\beta)'(Y - X\beta) + \lambda \sum_{j=1}^p w_j |\beta_j|, \tag{1.2}$$

where w_j adjusts the penalty level on β_j. The original Lasso sets $w_j = 1$ for all $j = 1, 2, \ldots, p$. Such a scheme makes the Lasso estimator variant with the unit of the predictor. That is, if we change X_j by multiplying it with a non-zero constant c, the coefficient $\hat{\beta}_j$ will not change to $\hat{\beta}_j/c$ accordingly. This is an undesirable property.

In software packages, the regressors are mostly mean-scale normalized, that is, setting $w_j = \hat{\mathrm{sd}}(X_j)$ as the sample variance of the jth predictor. The intercept has no variance and therefore is usually not penalized.

Under some regularity conditions (most importantly, the true coefficient is sparse), if the tuning parameter is chosen as $\lambda = C\sqrt{\frac{\log p}{n}}$, then we have prediction risk consistency

$$\frac{1}{n}\|X'\hat{\beta} - X'\beta_0\|_2^2 \xrightarrow{p} 0,$$

and parameter estimation consistency

$$\|\hat{\beta} - \beta_0\|_1 \xrightarrow{p} 0, \quad \|\hat{\beta} - \beta_0\|_2 \xrightarrow{p} 0.$$

Lee, Shi and Gao (2022) calls the Lasso estimator under $w_j = 1$ the **plain Lasso (Plasso)**, and the one under $w_j = \hat{\text{std}}(X_j)$ the **standardized Lasso (Slasso)**. They find that even under a fixed p the asymptotic rates of convergence of Plasso and Slasso are the same if the data is i.i.d. or weakly stationary, but their behaviors are very different when some regressors are nonstationary. Mei and Shi (2024) further investigate the problem in the high-dimensional setting when $p \to \infty$.

1.4.2 Variable Selection

Lasso was originally motivated as a variable selector (Tibshirani, 1996). However, Zou (2006) finds that Lasso consistently selects the true model only under very restrictive conditions, and he recommends an easy modification called the **adaptive Lasso**. Adaptive Lasso is a two-step scheme: (i) run Lasso or Ridge and save the estimator $\hat{\beta}^{(1)}$; (ii) Solve (1.2) by setting $w_j = |\hat{\beta}_j^{(1)}|^{-a}$ and $a \geq 1$ is a constant. (Common choice is $a = 1$ or 2. I personally prefer $a = 1$ to make the estimator unit-invariant.) Adaptive Lasso enjoys the **oracle property**: it achieves variable selection consistency and (pointwise) asymptotic normality simultaneously.

Another alternative variable selection method is smoothly-clipped-absolute-deviation (SCAD) (Fan and Li, 2001). Its criterion function is

$$(2n)^{-1}(Y - X\beta)'(Y - X\beta) + \sum_{j=1}^{d} \rho_\lambda(|\beta_j|)$$

where $\rho'_\lambda(\theta) = \lambda\left\{1\{\theta \leq \lambda\} + \frac{(a\lambda-\theta)_+}{(a-1)\lambda} \cdot 1\{\theta > \lambda\}\right\}$ for some $a > 2$ and $\theta > 0$. SCAD also enjoys the oracle property.

1.5 Bias-variance trade-off

Consider the density estimation given a sample $(x_1, \ldots, x_n)$. If we assume that the sample is drawn from a parametric family, for example the normal distribution, then we can use the maximum likelihood estimation to learn the mean and the variance. Nevertheless, when the parametric family is misspecified, the MLE estimation is inconsistent in theory, and we can at best identify a pseudo-true value.

In practice, the correct parametric family of the data generating process is unknown. We will have to use an infinite number of parameters to fully characterize the density. One well-known nonparametric estimation is the histogram. The shape of the bars of the histogram

depends on the partition of the support. If the grid system on the support is too fine, then each bin will have only a few observations. Despite small bias, the estimation will suffer a large variance. On the other hand, if the grid system is too coarse, then each bin will be wide. It causes big bias, though the variance is small because each bin contains many observations. This is the fundamental **bias-variance tradeoff** that emerges in all machine learning methods.

1.5.1 Mean-Squared Error

Histogram is an example of unsupervised learning. For supervised learning with a continuously distributed target variable, the most popular objective is the MSE loss:

$$\text{MSE loss} = \frac{1}{n} \|y - f(x)\|_2^2,$$

which is the sample version of the population MSE

$$\text{population MSE} = \mathbb{E}\left[(y - f(x))^2\right]$$

We have shown in the first lecture that $m(x) = \mathbb{E}(y \mid x)$ minimizes the population MSE, and it can be decomposed as

$$\mathbb{E}\left[(y - f(x))^2\right] = \mathbb{E}\left[(m(x) - \widehat{f}(x))^2\right] + \sigma^2,$$

where $\sigma^2 = \text{var}\,(y - m(x))$ is the variance of the error term. For any estimator $\hat{f}$, we can further decompose

$$\begin{aligned}\mathbb{E}\left[(m(x) - \widehat{f}(x))^2\right] &= \mathbb{E}\left[\left(m(x) - \mathbb{E}[\hat{f}(x)] + \mathbb{E}[\hat{f}(x)] - \widehat{f}(x)\right)^2\right] \\ &= \mathbb{E}\left[\left(\mathbb{E}[\hat{f}(x)] - \widehat{f}(x) - bias\right)^2\right] = bias^2 + var\end{aligned}$$

where

$$\begin{aligned}bias &:= \mathbb{E}[\hat{f}(x)] - m(x) \\ var &:= \mathbb{E}\left[\left(\widehat{f}(x) - \mathbb{E}[\hat{f}(x)]\right)^2\right]\end{aligned}$$

and because

$$\mathbb{E}\left[bias\left(\mathbb{E}[\hat{f}(x)] - \widehat{f}(x)\right)\right] = bias \cdot \mathbb{E}\left[\left(\mathbb{E}[\hat{f}(x)] - \widehat{f}(x)\right)\right] = 0.$$

We thus have

$$\text{population MSE} = \sigma^2 + bias^2 + var.$$

1.6 Hyperparameters

Given a class of estimators that vary with tuning parameters, we want to seek reasonable values of tuning parameters to reduce MSE. Tuning parameters are also called **hyperparameters**. They are distinguished from the (plain) parameters since they cannot be easily embedded into an objective function for optimization. Suitable choice of tuning parameters is the key to achieve a balance of the bias and variance.

1.6.1 Information Criteria

Traditionally, econometrics uses information criteria to determine the number of regressors, for example in the ARMA models. They can also help determined the tuning parameter λ. The most popular information criteria are the **Akaike information criterion (AIC)**

$$\text{AIC: } \log \hat{\sigma}^2 + \frac{2\hat{p}(\lambda)}{n},$$

and the **Bayesian information criterion (BIC)**

$$\text{BIC: } \log \hat{\sigma}^2 + \frac{\log n \cdot \hat{p}(\lambda)}{n}.$$

where $\hat{p}(\lambda)$ is the **effective degrees of freedom**. Ridge's $\hat{p}(\lambda) = \sum_j \frac{d_j}{d_j+\lambda}$ and Lasso's $\hat{p}(\lambda)$ can be set as the number of non-zero coefficients.

1.6.2 Sample splitting

The validity of the information criteria can be proved under relatively restrictive ideas, and the degree of efficient parameters is not easy to compute in complex models. Therefore, if we have a large dataset, machine learning uses data-driven methods by partitioning a big dataset into multiple blocks.

Many machine learning methods do not explicitly specify a data generating process. Instead, they focus on the predictive performance in test data.

1. Training: use $(Y, X)_{\text{train}}$, find $\hat{f}_\lambda$ given a grid system of λ.

2. Validation: in the validation data $\hat{\lambda}$, check which λ minimizes the objective. Mark it as $\hat{\lambda}$ and then we obtain a pre-trained model $\hat{f}_{\hat{\lambda}}$.

3. Test data: $(Y, X)_{\text{test}}$. We use $\hat{f}_{\hat{\lambda}}(X_{\text{test}})$ to predict y_{test} and check the performance.

Training data is used to fit the plain parameter given the tuning parameters. **Validation data** is used to compare the out-of-sample performance under a grid system of tuning parameters. It helps decide a set of desirable tuning parameters.

Ideally, **test data** should be kept by a third party away from the modeler, and is only revealed after the model is presented. The test sample is the final judge of the relative merit of the fitted models.

1.6.3 Cross-validation

An S-fold cross validation partitions the dataset into S disjoint sections. In each iteration, it picks one of the sections as the validation sample and the other $S-1$ sections as the training sample, and computes an out-of-sample goodness-of-fit measurement, for example **mean-squared prediction error** ${n_v}^{-1}\sum_{i\in val}(y_i-\hat{y}_i)^2$ where val is the validation set and n_v is its cardinality, or **mean-absolute prediction error** ${n_v}^{-1}\sum_{i\in val}|y_i-\hat{y}_i|$. Repeat this process for S times so that each of the S sections are treated as the validation sample, and average the goodness-of-fit measurement over the S sections to determine the best tuning parameter. If $S=n$, it is called **leave-one-out cross validation**, but it can be computationally too expensive when n is big. Instead, in practice we can $S=5$ for 10, called 5-fold cross validation or 10-fold cross validation, respectively.

In time series context, cross validation must preserve the dependence structure. If the time series is stationary, we can partition the data into S consecutive blocks. If the purpose is ahead-of-time forecasting, then we can use nested CV.

1.7 Conclusion

Econometrics	**Machine Learning**
$y=x'\beta+e$	$y=f(x)+e$
learn β	learn $f(x)$
no tuning parameter	tuning parameter
focus on inference of β	focus on predictability

Some machine learning methods are viewed as black-boxes as their learning mechanisms are not well-understood and they results cannot be easily interpreted.

This workflow of machine is quite different from the classical econometrics workflow, which starts from a generative model and checks the identification of the key parameter of interest in a thought experiment. If the parameters can be identified, we go ahead to use data for point estimation and interval estimation (statistical inference). Finally, based on the empirical results, we provide some economic interpretation.

The theory of machine learning is my main focus of research.

- Variable selection (*Lee, Shi & Gao, 2022*).
- Feature engineering (*Mei & Shi, 2024*).
- GMM-Lasso (*Shi, 2016*).
- Forward selection (*Shi & Huang, 2023*).
- Ridge-type boosting (*Phillips & Shi, 2021*).