Linear Methods for Regression

In this chapter, we introduce several popular linear methods for regression problem, which have been widely used in practice. The advantage of linear methods is that they always have good theoretical guarantees due to the simplicity.

1.1. Linear regression

Linear regression is the simplest method in statistics and machine learning, and it often serves as a good illustrative example for understanding machine learning models and algorithms.

The hypothesis space of linear regression is given by

$$\mathcal{H} = \left\{ \boldsymbol{\beta}^T \boldsymbol{x} + \beta_0 \,:\, \boldsymbol{\beta} \in \mathbb{R}^d, \beta_0 \in \mathbb{R} \right\}.$$

In this case, $\theta = (\boldsymbol{\beta}, \beta_0)$ are the parameters to be learned from data. In machine learning, it is customary to introduce the extended coordinate $\tilde{\boldsymbol{x}} = (\boldsymbol{x}^T, 1)^T \in \mathbb{R}^{d+1}$ and let $\tilde{\boldsymbol{\beta}} = (\boldsymbol{\beta}^T, \beta_0)^T \in \mathbb{R}^{d+1}$. Then, we can write $\mathcal{H} = \left\{ \tilde{\boldsymbol{\beta}}^T \tilde{\boldsymbol{x}} \right\}$. In fact, it is often simply to write

$$\mathcal{H} = \left\{ \boldsymbol{\beta}^T \boldsymbol{x} \right\}.$$

Given the data set $\mathcal{S} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, the empirical risk is given by

$$\hat{\mathcal{R}}_n(\boldsymbol{\beta}) = \frac{1}{n} \sum_{j=1}^{n} \frac{1}{2} \left(\boldsymbol{\beta}^T \boldsymbol{x}_j - y_j \right)^2 = \frac{1}{2n} \| X \boldsymbol{\beta} - \boldsymbol{y} \|_2^2. \tag{1.1}$$

Here, $X = (\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n)^T \in \mathbb{R}^{n\times(d+1)}$ be the data matrix and $\boldsymbol{y} = (y_1, y_2, \cdots, y_n)^T \in \mathbb{R}^n$.

1.1.1. Ordinary least squares (OLS). In OLS, the solution is chosen to be the minimizer of the empirical risk (1.1). Set $\nabla\hat{\mathcal{R}}_n(\boldsymbol{\beta}) = 0$, and we obtain

$$\sum_{j=1}^{n} \left(\boldsymbol{\beta}^T \boldsymbol{x}_j\right) \boldsymbol{x}_j = \sum_{j=1}^{n} y_j \boldsymbol{x}_j. \tag{1.2}$$

One can then write (1.2) as

$$\left(X^T X\right) \boldsymbol{\beta} = X^T \boldsymbol{y}. \tag{1.3}$$

The above equation is known as the normal equation and $X^T X$ is called the Gram matrix.

Suppose that $X^T X$ is full rank. The OLS estimator can be expressed as

$$\hat{\boldsymbol{\beta}} = \left(X^T X\right)^{-1} X^T \boldsymbol{y}. \tag{1.4}$$

When $X^T X$ is singular, e.g., in the over-parameterized case: $d + 1 > n$, we usually pick up the minimum-norm solution:

$$\begin{aligned} &\text{minimize} \quad \|\boldsymbol{\beta}\|_2 \\ &\text{subject to} \quad X\boldsymbol{\beta} = \boldsymbol{y}. \end{aligned} \tag{1.5}$$

If $\operatorname{rank}(X) = n$, the minimum-norm solution can be expressed as

$$\boldsymbol{\beta} = X^T (X X^T)^{-1} \boldsymbol{y}. \tag{1.6}$$

In practice, the labels are often noisy:

$$y_i = \boldsymbol{\beta}^T \boldsymbol{x}_i + \xi_i$$

with $\xi_i \neq 0$. Therefore, we do not use the OLS estimator directly, since it overfits the noise, thereby hurting the generalization performance. To deal with this issue, the popular approach is to consider regularized methods, which minimizes following penalized empirical risk:

$$\frac{1}{2n}\|X\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \lambda\, r(\boldsymbol{\beta}).$$

Here, $r(\boldsymbol{\beta})$ is the penalization term, which incorporates our prior knowledge about the data. λ is the hyper-parameter that controls the trade-off between the fitting error and the penalty. The questions is: How do we choose the penalty function $r(\cdot)$ and set the value of hyper-parameter λ?

1.1.2. Ridge regression. In this section, we introduce the simplest regularized model: ridge regression, for which $r(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_2^2/2$. Thus, the objective function becomes

$$\frac{1}{2n}\|X\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \frac{1}{2}\lambda \|\boldsymbol{\beta}\|_2^2 . \tag{1.7}$$

One advantage of this regularization is that the minimizer has a closed-form expression:

$$\hat{\boldsymbol{\beta}} = \left(\frac{1}{n}X^T X + \lambda I_d\right)^{-1} X^T \boldsymbol{y}. \tag{1.8}$$

Note that the ridge regression is a special case of the Tikhonov regularization, where the objective function is

$$\frac{1}{2n}\|X\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \frac{1}{2}\lambda \|\Gamma\boldsymbol{\beta}\|_2^2 . \tag{1.9}$$

Here, Γ is the Tikhonov matrix, which controls the effect of regularization through different coordinates. Ridge regression corresponds to $\Gamma = I_d$.

1.1.3. Least absolute shrinkage and selection operator (Lasso). Another popular regularized linear model is Lasso:

$$\frac{1}{2n}\|X\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_1 . \tag{1.10}$$

Different from the ridge regression, Lasso penalize the ℓ_1 norm. The motivation to consider the ℓ_1 norm is to promote sparsity. Let us first make the sparsity assumption as follows.

Assumption 1.1 (Sparsity)**.** *Let* $\|\boldsymbol{\beta}\|_0 = \#\{i \in [d] : |\beta_i| > 0\}$. *Assume that the ground truth* $\boldsymbol{\beta}^*$ *satisfies* $\|\boldsymbol{\beta}^*\|_0 \ll d$.

This sparsity assumption is satisfied in many applications, where only a few coordinates/variables matter. In this case, we are not only interested in the prediction but also discovering of these important variables.

To promote sparsity, the natural regularized model is

$$\min_{\boldsymbol{\beta}} \frac{1}{2n}\|X\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_0. \tag{1.11}$$

However, (1.11) is computationally intractable since the ℓ_0 norm is non-continuous and non-convex. Here we use the terminology "norm" in a loose way and it mainly means a quantity that controls the model complexity. To circumvent this challenge, one can choose to relax ℓ_0 to ℓ_p with $p > 0$:

$$\min_{\boldsymbol{\beta}} \frac{1}{2n}\|X\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_p, \tag{1.12}$$

Shown in Figure 1 are the landscapes of one-dimensional ℓ_p norm for various p's. Obviously, ℓ_p norm is continuous as long as $p > 0$ but it is convex only when $p \geq 1$. Solving convex problems are often easier than solving non-convex ones. Hence, it is not surprising that $p = 1$ is preferred since it is the smallest p that ensures convexity.

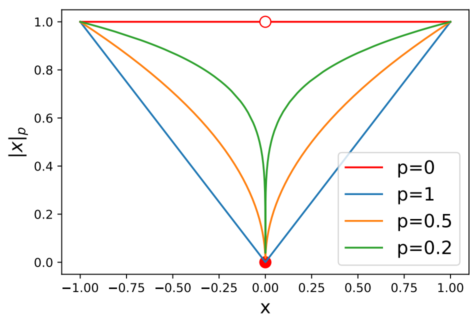

Figure 1. An illustration of the landscape of ℓ_p norm for various p's. Here, x is an one-dimensional variable.

However, the preceding intuitive explanations only suggest that ℓ_1 is close to ℓ_0 in some sense. It is unclear if the ℓ_1 solution is sparse when the ground truth $\boldsymbol{\beta}^*$ is sparse. To see how the ℓ_1 relaxation works, we examine the following constraint problem (the relation with the problem (1.10) is discussed in the exercise):

$$\min_{\|\boldsymbol{\beta}\|_p \leq t} \frac{1}{2}\|X\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 \tag{1.13}$$

Shown in Figure 2 are the contour curves of $\hat{\mathcal{R}}_n(\beta) = \frac{1}{2}\|X\boldsymbol{\beta}^T - \boldsymbol{y}\|_2^2$ (dashed curves) and $r(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|_p$ (solid curves).

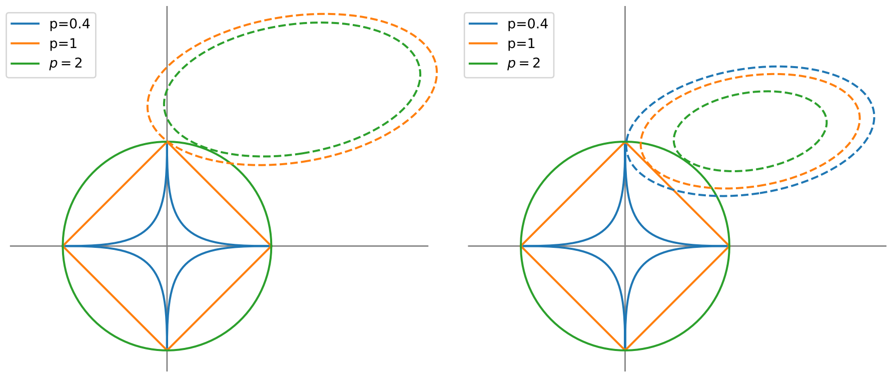

Figure 2. Illustration of how the ℓ_p norm promotes the sparsity when $0 < p \leq 1$. **Left:** Both ℓ_1 and $\ell_{0.4}$ succeeds; **Right:** ℓ_1 fails but $\ell_{0.4}$ succeeds.

We have the following observations:

- For $0 < p \leq 1$, the contour curves of $\hat{\mathcal{R}}_n(\cdot)$ tend to touch first at the "sharp" corners of the contour curves of $r(\cdot)$. In other words, the ℓ_p regularization promotes the sparsity when $0 < p \leq 1$. By contrast, the ℓ_2 norm does not show this preference.
- The smaller is p, the stronger is the sparsity. The right figure provides an example where ℓ_1 fails in promoting sparsity while $\ell_{0.4}$ succeeds.
- Geometrically speaking, it is the sharp corners that is most important for promiting sparsity.

Concerning both the computational feasibility and sparsity promotion, the natural choice is the Lasso (1.10), which can be viewed as a convex relaxation of (1.11).

Compressed sensing. By comparing the left and the right panel in Figure 2, we can conclude that the ℓ_1 regularization can promote sparsity only if the input data satisfy certain conditions. The theory of compressed sensing identifies some of these conditions.

Consider the problem of finding the sparsest solution:

$$\begin{aligned} \min \quad & \|\boldsymbol{\beta}\|_0\,, \\ \text{subject to} \quad & X\boldsymbol{\beta} = \boldsymbol{y}, \end{aligned} \tag{1.14}$$

where $X \in \mathbb{R}^{n\times d}$ with $d > n$ is a "fat" matrix .

Definition 1.2. A vector $\boldsymbol{\beta}$ is said to be s-sparse if $\|\boldsymbol{\beta}\|_0 \leq s$. Here s is a positive integer.

Definition 1.3. X is said to satisfy the restricted isometry property (RIP) if there exists a $\delta_s \in (0,1)$ such that

$$(1-\delta_s)\|\boldsymbol{\beta}\|_2 \leq \|X\boldsymbol{\beta}\|_2 \leq (1+\delta_s)\|\boldsymbol{\beta}\|_2$$

holds for all s-sparse vectors $\boldsymbol{\beta}$.

Theorem 1.4. *Let $\boldsymbol{\beta}_1$ be a solution of*

$$\begin{aligned} \min \quad & \|\boldsymbol{\beta}\|_1\,, \\ s.t. \quad & X\boldsymbol{\beta} = \boldsymbol{y}, \end{aligned} \tag{1.15}$$

and $\boldsymbol{\beta}_0$ be the solution of (1.14). Assume that $\delta_{2s} < \sqrt{2}-1$. Then

$$\|\boldsymbol{\beta}_1 - \boldsymbol{\beta}_0\|_1 \leq C_0\,\|\boldsymbol{\beta}_0 - T_s(\boldsymbol{\beta}_0)\|_1\,, \tag{1.16}$$

where T_s is a function defined as follows

$$(T_s(\boldsymbol{\beta}))_j = \begin{cases} (\boldsymbol{\beta})_j, & \textit{if } (\boldsymbol{\beta})_j \textit{ is among the } s \textit{ largest components of } \boldsymbol{\beta}, \\ 0, & \textit{otherwise.} \end{cases}$$

In particular, if $\boldsymbol{\beta}_0$ is s-sparse, then $\boldsymbol{\beta}_1 = \boldsymbol{\beta}_0$.

This theorem tells us that when the RIP condition is satisfied, ℓ_1 can recover ℓ_0 with the error depending on the sparsity of ℓ_0 solutions. In particular, when the ℓ_0 solution is sparse, the recovery is exact. It is of great importance to understand the RIP condition and the implications of this theorem. However, the proof is quite techinical and thus we do not present it here. Interested readers can find it in [**CRT06**].

Some analyses of Lasso. The following questions arise naturally:

- How should we choose λ?
- How big is the error?
- How much is the effect of the noise?

Consider the situation where the ground truth $\boldsymbol{\beta}^*$ is sparse and the signal $\boldsymbol{y}$ is contaminated by noise:

$$y_i = \boldsymbol{\beta}^{*T}\boldsymbol{x}_i + \varepsilon_i, \tag{1.17}$$

where ε_i is the random noise. Let $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n)^T \in \mathbb{R}^n$ be the noise vector. Denote by $\hat{\boldsymbol{\beta}}$ the Lasso estimator (1.10).

Proposition 1.5. For the Lasso estimator, if $\lambda_n \geq \frac{\|X^T\boldsymbol{\varepsilon}\|_\infty}{n}$, then

$$\frac{1}{2n}\|X(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*)\|_2^2 \leq 2\lambda_n\|\boldsymbol{\beta}^*\|_1. \tag{1.18}$$

(1.18) shows that the empirical risk is well-controlled when the penalization is relatively large with respect to the noise, i.e. $\lambda_n \geq \|X^T\boldsymbol{\varepsilon}\|_\infty/n$. The proof given below is simple but very representative for generalization analysis of regularized estimators. The main idea is to compare the estimator of interest with "ground truth" and to identify some conditions such that the estimator has properties similar to ground truth. This comparison trick will appear many times in this book and we will see in exercise that this comparison can also lead to the controlledness of the norm of Lasso estimator.

Proof. By comparing $\hat{\boldsymbol{\beta}}$ with the ground truth $\boldsymbol{\beta}^*$, we have

$$\frac{1}{2n}\|X\hat{\boldsymbol{\beta}} - \boldsymbol{y}\|_2^2 + \lambda_n\|\hat{\boldsymbol{\beta}}\|_1 \leq \frac{1}{2n}\|X\boldsymbol{\beta}^* - \boldsymbol{y}\|_2^2 + \lambda_n\|\boldsymbol{\beta}^*\|_1 \tag{1.19}$$

Notice that $\boldsymbol{y} = X\boldsymbol{\beta}^* + \boldsymbol{\varepsilon}$ implies for any $\boldsymbol{\beta} \in \mathbb{R}^d$ that

$$\|X\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 = \|X\boldsymbol{\beta} - X\boldsymbol{\beta}^*\|_2^2 + 2\langle X(\boldsymbol{\beta} - \boldsymbol{\beta}^*), \boldsymbol{\varepsilon}\rangle + \|\boldsymbol{\varepsilon}\|_2^2. \tag{1.20}$$

Inserting (1.20) into the comparison inequality (1.19), we obtain the following estimates.

$$\begin{aligned}\frac{1}{2n}\|X(\hat{\boldsymbol\beta}-\boldsymbol\beta^*)\|_2 &\le \frac{1}{n}\langle X(\hat{\boldsymbol\beta}-\boldsymbol\beta^*),\boldsymbol\varepsilon\rangle+\lambda_n\|\boldsymbol\beta^*\|_1-\lambda_n\|\hat{\boldsymbol\beta}\|_1\\ &\le \frac{\|X^T\boldsymbol\varepsilon\|_\infty}{n}\|\hat{\boldsymbol\beta}-\boldsymbol\beta^*\|_1+\lambda_n(\|\boldsymbol\beta^*\|_1-\|\hat{\boldsymbol\beta}\|_1).\end{aligned}$$

Since $\lambda_n \ge \frac{\|X^T\boldsymbol\varepsilon\|_\infty}{n}$, we have

$$\begin{aligned}\frac{1}{2n}\|X(\hat{\boldsymbol\beta}-\boldsymbol\beta^*)\|_2^2 &\le \lambda_n\left(\|\hat{\boldsymbol\beta}-\boldsymbol\beta^*\|_1-\|\hat{\boldsymbol\beta}\|_1+\|\boldsymbol\beta^*\|_1\right)\\ &\le 2\lambda_n\|\boldsymbol\beta^*\|_1.\end{aligned}$$

□

Theorem 1.6. *Assume $\|X_j\|_2^2=n,\forall j\in[d]$ where X_j is the j-th column of X and the noise $\varepsilon_i \overset{iid}{\sim} \mathcal{N}(0,\sigma^2)$ for $i=1,\dots,n$. For any $\delta\in(0,1)$, with prob. $1-\delta$ over the sampling of the random noise, we have*

$$\frac{\|X^T\boldsymbol\varepsilon\|_\infty}{n}\le C\sigma\sqrt{\frac{\log(d/\delta)}{n}},$$

and

$$\frac{1}{n}\|X(\hat{\boldsymbol\beta}-\boldsymbol\beta^*)\|_2^2\le\frac{C\sigma\sqrt{\log(d/\delta)}\|\boldsymbol\beta^*\|_1}{\sqrt{n}}.$$

This theorem suggests that λ_n should decrease with the sample size n, which is consistent with our intuition: A weaker regularization is preferred when more samples are used. In addition, when $\boldsymbol\beta^*$ is s-sparse, $\|\boldsymbol\beta^*\|_1=O(s)$ and the resulting error depends on the dimension only *logarithmically*. This also explains why ℓ_1 regularization is favorable in high dimension when the ground truth is sparse. It should be stressed that the preceeding analysis only provides an estimate of the training error. We will see later that a similar bound also holds for the generalization error, i.e., $\mathbb{E}_{\boldsymbol x}[|\boldsymbol x^T\hat{\boldsymbol\beta}-\boldsymbol x^T\boldsymbol\beta^*|^2]$.

Proof. By Proposition 1.5, what remains is to estimate $\|X^T\boldsymbol\varepsilon\|_\infty=\max_{j\in[d]}|X_j^T\boldsymbol\varepsilon|$. Since $\boldsymbol\varepsilon\sim\mathcal{N}(0,\sigma^2 I_d)$, $X_j^T\boldsymbol\varepsilon\sim\mathcal{N}(0,\|X_j\|_2^2\,\sigma^2)$. Then, we have

$$\begin{aligned}\mathbb{P}\left\{\max_{1\le j\le d}\left|X_j^T\boldsymbol\varepsilon\right|\ge t\right\} &\le d\,\mathbb{P}\left\{\left|X_1^T\boldsymbol\varepsilon\right|\ge t\right\}\\ &=2d\int_t^\infty\frac{1}{\sqrt{2\pi\sigma^2 n}}\,e^{-\frac{z^2}{2\sigma^2 n}}\,\mathrm{d}z\\ &=2d\frac{1}{\sqrt{2\pi}}\int_{\frac{t}{\sigma\sqrt{n}}}^\infty e^{-\frac{z^2}{2}}\,\mathrm{d}z.\end{aligned}$$

Notice that the tail of standard norm distribution satisfies

$$\frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-\frac{z^2}{2}}\,\mathrm{d}z \le e^{-\frac{x^2}{2}}, \qquad \forall x > 0.$$

Therefore,

$$\mathbb{P}\left\{\max_{1\le j\le d}\left|X_j^T\boldsymbol{\varepsilon}\right| \ge t\right\} \le \frac{2d}{\sqrt{2\pi}}e^{-\frac{t^2}{2\sigma^2 n}}.$$

Let the failure probability $\frac{2d}{\sqrt{2\pi}}e^{-\frac{t^2}{2\sigma^2 n}} \le \delta$. We have $t \ge C\sigma\sqrt{n\log(d/\delta)}$. Therefore, we can conclude that with probability $1-\delta$,

$$\frac{\|X^T\boldsymbol{\varepsilon}\|_\infty}{n} \le C\sigma\sqrt{\frac{\log(d/\delta)}{n}}.$$

□

The normalization condition $\|X_j\|_2^2 = \sum_{i=1}^n x_{i,j}^2 = n$ ensures that the value of each coordinate is roughly $O(1)$. From the proof, one can see the Gaussian assumption: $\varepsilon_i \sim \mathcal{N}(0,\sigma^2)$ is not essential and it can be replaced by any distribution as long as its tail decays like a Gaussian, i.e., $\mathbb{P}\{|\varepsilon_i| > t\} \le C_1e^{-C_2t^2}$ for some positive constants C_1, C_2. This kind of tail property is often referred to as being sub-Gaussian (see [**Ver18**, Section 2] for more details).

Variable Selection. Lasso has become one of most popular method in statistics since it can perform not only prediction but also variable selection. This is due to the sparsity-promoting effect of ℓ_1 regularization, which can set the coefficients of unimportant variables exactly to zeros (see Theorem 1.4). The variables with non-zero coefficients naturally account for the prediction, and this interpretability is very important in many applications. By contrast, ridge regression does not possess this property.

Define $\hat{\boldsymbol{\beta}} : [0,\infty) \mapsto \mathbb{R}^d$ be the *regularization path* of Lasso defined by

$$\hat{\boldsymbol{\beta}}(\lambda) = \arg\min_{\boldsymbol{\beta}} \frac{1}{2n}\|X\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1. \tag{1.21}$$

Analogously, we can define the regularization path of ridge regression. Shown in Figure 3 are the regularization paths of Ridge and Lasso, respectively. One can see that for Ridge, no matter how big is λ, all the coefficients are always non-zero (though their specific values may be small). By contrast, Lasso sequentially set the coefficients of unimportant variables to exact zeros as increasing λ. This intriguing property of Lasso regularization path facilitates the variable selection.

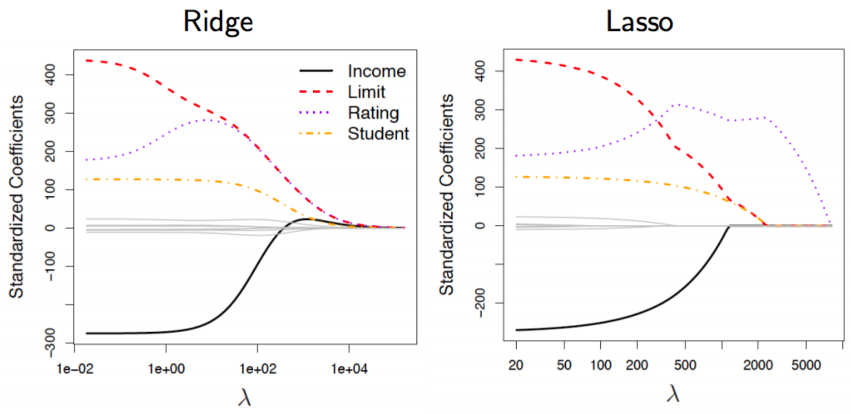

Figure 3. Regulation paths $\hat{\boldsymbol{\beta}}(\lambda)$ for ridge regression (**Left**) and Lasso regression (**Right**). The response is the average credit debt, the predictors are income, limit (credit limit), rating (credit rating), student (indicator), and others. (taken from Ryan Tibshirani's slides) [LW: Need to reproduce this figure.]

Algorithms for Lasso. Let us first look at the one-dimensional case:

$$S_\lambda(x) = \arg\min_t \frac{1}{2}(x-t)^2 + \lambda|t|. \tag{1.22}$$

In this case, the minimizer has a closed-form expression:

$$S_\lambda(x) = \begin{cases} x-\lambda, & \text{if } x > \lambda \\ 0, & \text{if } -\lambda \le x \le \lambda \\ x+\lambda, & \text{if } x < -\lambda. \end{cases} \tag{1.23}$$

$S_\lambda(\cdot)$ is called the **soft thresholding** function. As a comparison, the hard thresholding function is defined as

$$H_\lambda(x) = \arg\min_t \frac{1}{2}(x-t)^2 + \lambda|t|_0, \tag{1.24}$$

which also has a closed-form expression:

$$H_\lambda(x) = \begin{cases} x & \text{if } x > \lambda \\ 0, & \text{if } -\lambda \le x \le \lambda \\ x, & \text{if } x < -\lambda. \end{cases} \tag{1.25}$$

Figure 4 provides a visualization of S_λ and H_λ. By comparing them, we see that they both promote sparsity but in slightly different ways. Specifically, $S_\lambda(x)$ shrinks x to zero exactly when the absolute value of x is smaller than the threshold. This again explains why the ℓ_1 norm can promote sparsity and is useful for variable selection. As a comparison, let us look at the

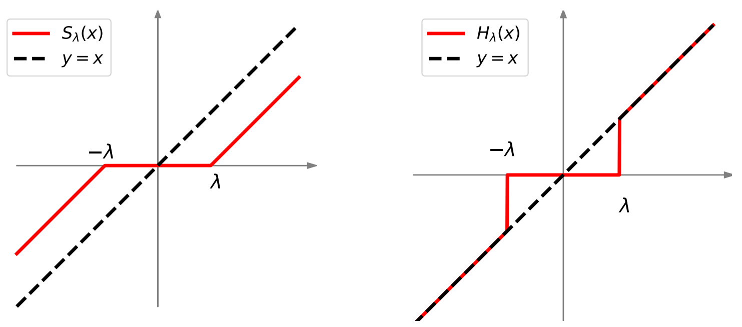

Figure 4. Left: The soft-thresholding function; **Right:** The hard-thresholding function.

solution of Ridge:

$$R_\lambda(x) = \arg\min_t \frac{1}{2}(t-x)^2 + \frac{\lambda}{2}|t|^2 = \frac{x}{1+\lambda}.$$

This suggests that Ridge does not shrink x to zero no matter how small x is. In other words, Ridge does not promote sparsity.

For the d-dimensional case, we can convert it into a sequence of one-dimensional problems by using the coordinate descent method. Consider the minimization of $f(x_1, \ldots, x_d)$. The coordinate descent method repeats the following cyclical iteration until convergence:

$$\begin{aligned} x_1^{(t+1)} &\in \arg\min f(x_1, x_2^{(t)}, x_3^{(t)} \ldots, x_d^{(t)}) \\ x_2^{(t+1)} &\in \arg\min f(x_1^{(t+1)}, x_2, x_3^{(t)} \ldots, x_d^{(t)}) \\ &\cdots \\ x_d^{(t+1)} &\in \arg\min f(x_1^{(t+1)}, x_2^{(t+1)}, x_3^{(t+1)} \ldots, x_d). \end{aligned}$$

Notice that the above is a Gauss-Seidel type update.

Let X_k with $k \in [d]$ be the k-th column of X.

$$\begin{aligned} \|X\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 &= \Big\|\boldsymbol{y} - \sum_{k\neq j} X_k\beta_k\Big\|_2^2 - 2\Big\langle \boldsymbol{y} - \sum_{k\neq j} X_k\beta_k, X_j\beta_j\Big\rangle + \|X_j\|_2^2\,\beta_j^2 \\ &= \|\tilde{y}_j\|_2^2 - 2\langle \tilde{y}_j, X_j\rangle\beta_j + \|X_j\|_2^2\,\beta_j^2, \end{aligned}$$

where $\tilde{y}_j = y - \sum_{k\neq j} X_k\beta_k$.

Then the update of j-coordinate is given by

$$\beta_j \leftarrow \arg\min_{\beta_j} \frac{1}{2n}\left(\|X_j\|_2^2\beta_j^2 - 2\langle \tilde{y}_j, X_j\rangle\beta_j\right) + \lambda|\beta_j|$$

$$\beta_j = S_{\frac{n\lambda}{\|X_j\|_2^2}}\left(\frac{\langle \tilde{y}_j, X_j\rangle}{\|X_j\|_2^2}\right). \tag{1.26}$$

Repeat (1.26) for $j = 1, 2, \ldots, d$ until convergence.

The above coordinate descent method is simple and easy to implement. However, in practice, the most popular methods of solving large-scale ℓ_1 optimizations is the alternating direction method of multipliers (ADMM) and its variants. We refer interested readers to [**BPC11**] for more details. However, it should be stressed that coordinate update is a very general way of designing method and can be applicable to many situations.

1.2. Kernel methods

The previous methods can only deal with linear problems. A direct extension to nonlinear cases is to consider the following model:

$$f(\boldsymbol{x};\boldsymbol{\beta}) = \sum_{j=1}^{m} \beta_j \phi_j(\boldsymbol{x}), \tag{1.27}$$

where $\{\phi_j\}_{j=1}^m$ are a set of nonlinear basis functions. The typical examples include the Fourier basis, (local) polynomials, splines, etc. Note that the model is still linear in parameters but the represented function can be nonlinear.

The basis functions are often called "features" in machine learning and the corresponding feature map $\Phi : \mathcal{X} \mapsto \mathbb{R}^m$ is given by

$$\Phi(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), \ldots, \phi_m(\boldsymbol{x}))^T \in \mathbb{R}^m,$$

where $\mathcal{X}$ is the input space and $\mathbb{R}^m$ is the feature space. In many applications, one often choose $m > d$, where the feature map lifts the input $\boldsymbol{x}$ to a higher-dimensional feature space.

To fit the model (1.27), consider the ridge regression in feature space:

$$\begin{aligned}\hat{\boldsymbol{\beta}} &= \arg\min_{\boldsymbol{\beta}} \frac{1}{2n}\sum_{i=1}^{n}\left(\sum_{j=1}^{m}\beta_j\phi_j(\boldsymbol{x}_i) - y_i\right)^2 + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2 \\ &= \arg\min_{\boldsymbol{\beta}} \frac{1}{2n}\|\hat{\Phi}\boldsymbol{\beta} - \boldsymbol{y}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2, \end{aligned} \tag{1.28}$$

where $\hat{\Phi} = (\phi_j(\boldsymbol{x}_i))_{i,j} \in \mathbb{R}^{n\times m}$ is the feature matrix. Then,

$$\hat{\boldsymbol{\beta}} = \left(\frac{1}{n}\hat{\Phi}^T\hat{\Phi} + \lambda I_m\right)^{-1}\hat{\Phi}^T\boldsymbol{y}. \tag{1.29}$$

The function represented by $\hat{\boldsymbol{\beta}}$ is given by

$$\hat{f}(\boldsymbol{x}) = \sum_{j=1}^{m} \hat{\beta}_j \phi_j(\boldsymbol{x}). \tag{1.30}$$

The following theorem shows that $\hat{f}$ can be written as a linear combination of

$$k(\boldsymbol{x}, \boldsymbol{x}') = \Phi(\boldsymbol{x})^T \Phi(\boldsymbol{x}) = \sum_{j=1}^{m} \phi_j(\boldsymbol{x}) \phi_j(\boldsymbol{x}'). \tag{1.31}$$

Here, $k : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ is called the kernel function, and the rigorous definition is given later. Specifically, we have the following representation theorem for the solution (1.28).

Theorem 1.7. *Let $G = (k(\boldsymbol{x}_i, \boldsymbol{x}_j)) \in \mathbb{R}^{n\times n}$ be the Gram matrix. For any $\lambda > 0$, let $\hat{\boldsymbol{\alpha}} = \left(\frac{1}{n}G + \lambda I_n\right)^{-1} \boldsymbol{y}$. The solution of (1.28) can be rewritten as*

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{n} \hat{\alpha}_i k(\boldsymbol{x}_i, \boldsymbol{x}). \tag{1.32}$$

Proof. For any $A \in \mathbb{R}^{n\times m}$, we have the following identity

$$(A^T A + I_m)^{-1} A^T = A^T (A A^T + I_n)^{-1}.$$

Hence,

$$\begin{aligned} \hat{\boldsymbol{\beta}} &= \left(\frac{1}{n}\hat{\Phi}^T \hat{\Phi} + \lambda I_m\right)^{-1} \hat{\Phi}^T \boldsymbol{y} = \hat{\Phi}^T \left(\frac{1}{n}\hat{\Phi}\hat{\Phi}^T + \lambda I_n\right)^{-1} \boldsymbol{y} = \hat{\Phi}^T \hat{\boldsymbol{\alpha}} \\ &= \sum_{i=1}^{n} \Phi(\boldsymbol{x}_i) \alpha_i. \end{aligned}$$

Then, we have

$$\begin{aligned} \hat{f}(\boldsymbol{x}) &= \sum_{j=1}^{m} \phi_j(\boldsymbol{x}) \hat{\beta}_j = \sum_{j=1}^{m} \phi_j(\boldsymbol{x}) \sum_{i=1}^{n} \phi_j(\boldsymbol{x}_i) \hat{\alpha}_i \\ &= \sum_{i=1}^{n} \hat{\alpha}_i \left(\sum_{j=1}^{m} \phi_j(\boldsymbol{x}_i) \phi_j(\boldsymbol{x}) \right) = \sum_{i=1}^{n} \hat{\alpha}_i k(\boldsymbol{x}_i, \boldsymbol{x}). \end{aligned}$$

□

In (1.29), the computation overhead mainly comes from the inverse of a $m \times m$ matrix, while for (1.32), it becomes the inverse of $n \times n$ matrix. Therefore, (1.32) is computationally more efficient than (1.29) when $m > n$, i.e., the dimension of feature space is higher than the sample size.

Another intriguing observation is that the method only needs to specify the kernel $k(\cdot,\cdot)$ without needing to evaluate the actual features $\{\phi_j\}_{j=1}^m$. This insight applies to all the methods that only depend on the Gram matrix, where we can do the following replacement:

$$\boldsymbol{x}_i^T\boldsymbol{x}_j \longrightarrow \Phi(\boldsymbol{x}_i)^T\Phi(\boldsymbol{x}_j) = k(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

In the literature, this is called the *kernel trick*. This trick is applicable to many methods, such as PCA, density estimation, Fisher discrimination analysis. In this chapter, we focus on the ridge regression.

1.2.1. Kernel Methods. We generalize the previous observations to very general cases.

A "feature map" is defined as a map $\Phi : \mathcal{X} \mapsto \mathcal{H}$ where $\mathcal{X}$ is the input space and $\mathcal{H}$ is feature space. Here $\mathcal{H}$ can be any Hilbert space. Taking the example (1.27), $\mathcal{H} = \mathbb{R}^m$ and

$$\Phi(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), \phi_2(\boldsymbol{x}), \ldots, \phi_m(\boldsymbol{x}))^T \in \mathbb{R}^m.$$

Definition 1.8. $k : \mathcal{X}\times\mathcal{X} \mapsto \mathbb{R}$ is said to be a kernel if there exists a feature map $\Phi : \mathcal{X} \mapsto \mathcal{H}$ such that

$$k(\boldsymbol{x},\boldsymbol{x}') = \langle\Phi(\boldsymbol{x}),\Phi(\boldsymbol{x}')\rangle.$$

Notice that $\langle\cdot,\cdot\rangle$ denotes the inner product in $\mathcal{H}$. To simply notations, we omit the dependence on $\mathcal{H}$ when it is clear from the context.

Definition 1.9 (SPD function)**.** A function $k : \mathcal{X}\times\mathcal{X} \mapsto \mathbb{R}$ is a semi-positive definite (SPD) if

- $k(\boldsymbol{x},\boldsymbol{x}') = k(\boldsymbol{x}',\boldsymbol{x})$ for any $\boldsymbol{x},\boldsymbol{x}' \in \mathcal{X}$.
- The kernel matrix $K = (k(\boldsymbol{x}_i,\boldsymbol{x}_j)) \in \mathbb{R}^{n\times n}$ is SPD for any $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n \in \mathcal{X}$, i.e.,

$$\sum_{i,j=1}^n \alpha_i\alpha_j k(\boldsymbol{x}_i,\boldsymbol{x}_j) \geq 0, \quad \forall\boldsymbol{\alpha}\in\mathbb{R}^n.$$

Obviously, any kernel k is SPD:

$$\sum_{i,j=1}^n \alpha_i\alpha_j k(\boldsymbol{x}_i,\boldsymbol{x}_j) = \sum_{i,j=1}^n \alpha_i\alpha_j\langle\Phi(\boldsymbol{x}_i),\Phi(\boldsymbol{x}_j)\rangle = \|\sum_{i=1}^n \alpha_i\Phi(\boldsymbol{x}_i)\|^2 \geq 0.$$

The following theorem shows that the converse direction also holds and we will rigorously discuss it later in Moore-Aronszajn theorem.

Theorem 1.10. *For any SPD function $k : \mathcal{X}\times\mathcal{X} \mapsto \mathbb{R}$, there exists a feature map $\Phi : \mathcal{X} \mapsto \mathcal{H}$ with $\mathcal{H}$ be a Hilbert space such that*

$$k(\boldsymbol{x},\boldsymbol{x}') = \langle\Phi(\boldsymbol{x}),\Phi(\boldsymbol{x}')\rangle.$$

Hence, k is a kernel.

Thus we have that SPD functions and kernels are equivalent. Therefore, we can check if $k(\cdot,\cdot)$ is a kernel by verifying if $k(\cdot,\cdot)$ is SPD, without the need to know the feature map.

1.2.2. Examples of kernels. Here, we provide a list of popular kernels.

Polynomial kernel: $k(\boldsymbol{x},\boldsymbol{x}') = (1+\boldsymbol{x}^T\boldsymbol{x}')^s$ is SPD for any $s \in \mathbb{N}_+$.

- Linear ($s=1$). We have $k(\boldsymbol{x},\boldsymbol{x}') = \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}')\rangle$ with
$$\Phi(\boldsymbol{x}) = (1, x_1, \ldots, x_d).$$
- Quadratic ($s=2$): The feature map is given by

$$\Phi(\boldsymbol{x}) = (\underbrace{x_d^2,\ldots,x_1^2}_{\text{quadratic}}, \underbrace{\sqrt{2}x_dx_{d-1},\ldots,\sqrt{2}x_dx_1,\sqrt{2}x_{d-1}x_{d-2},\ldots,\sqrt{2}x_2x_1}_{\text{cross terms}}, \underbrace{\sqrt{2}x_d,\ldots,\sqrt{2}x_1}_{\text{linear terms}}, \underbrace{1}_{\text{constant}}).$$

$$\begin{aligned}\langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}')\rangle &= \sum_{i=1}^d (x_i)^2(x_i')^2 + 2\sum_{i\neq j} x_ix_jx_i'x_j' + 2\sum_i x_ix_i' + 1\\ &= (\sum_{i=1}^d x_ix_i')^2 + 2\sum_i x_ix_i' + 1\\ &= (\boldsymbol{x}^T\boldsymbol{x}'+1)^2 \qquad (1.33)\end{aligned}$$

Gaussian kernel: $k(\boldsymbol{x},\boldsymbol{x}') = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|_2^2}{2}}$. Considering $d=1$, we have

$$\begin{aligned}k(x,x') &= e^{-\frac{x^2}{2}-\frac{x'^2}{2}}e^{xx'} = e^{-\frac{x^2}{2}-\frac{x'^2}{2}}\sum_n \frac{1}{n!}(x)^n(x')^n\\ &= \langle \Phi(x), \Phi(x)\rangle,\end{aligned}$$

where $\Phi(x) = e^{-\frac{x^2}{2}}(1, x, \frac{1}{\sqrt{2}}x^2, \ldots, \frac{1}{\sqrt{n!}}x^n, \ldots)$.

The general Gaussian kernel is defined by

$$k(\boldsymbol{x},\boldsymbol{x}') = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|_2^2}{2\sigma^2}}.$$

Intuitively, the Gaussian kernel sets the inner product in the feature space between $\boldsymbol{x}$ and $\boldsymbol{x}'$ to be close to zero if they are far away from each other in the original space. σ is the parameter (often referred to as the bandwidth) that controls the scale determining what we mean by "close".

Laplace kernel:

$$k(\boldsymbol{x},\boldsymbol{x}') = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|_2}{\sigma}}.$$

This kernel is less smooth than the Gaussian kernel. Recently, it has been shown that the Laplace kernel is intimately related to neural network models (in kernel regime)[CX20, GYK+20].

Dot-product kernels [1]**:** Let $\mathbb{S}^{d-1} = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 = 1\}$. A kernel $k : \mathbb{S}^{d-1} \times \mathbb{S}^{d-1} \mapsto \mathbb{R}$ is said to be dot-product if there exists a $\kappa : [-1, 1] \mapsto \mathbb{R}$ such that

$$k(\boldsymbol{x}, \boldsymbol{x}') = \kappa(\boldsymbol{x}^T \boldsymbol{x}'),$$

which means that the kernel value only depends on the inner-product of two inputs. For instance, the Laplace kernel constrained on spheres is dot-product:

$$k(\boldsymbol{x}, \boldsymbol{x}') = e^{-\|\boldsymbol{x}-\boldsymbol{x}'\|_2} = e^{-\sqrt{2-2\boldsymbol{x}^T\boldsymbol{x}'}} = \kappa(\boldsymbol{x}^T \boldsymbol{x}'),$$

where $\kappa(t) = e^{-\sqrt{2-2t}}$. We can see that κ is not differetiable at $t = 1$.

Similarly, the Gaussian kernel constrained on spheres is also dot-product:

$$k(\boldsymbol{x}, \boldsymbol{x}') = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|_2^2}{2}} = e^{-1+\boldsymbol{x}^T\boldsymbol{x}'} = \kappa(\boldsymbol{x}^T \boldsymbol{x}'),$$

where $\kappa(t) = Ce^t$. This time κ is analytic on the whole domain.

One can also construct new kernels by using existing kernels. Let k_1, k_2 are two kernels. Then,

- $k(\boldsymbol{x}, \boldsymbol{x}') = k_1(\boldsymbol{x}, \boldsymbol{x}') + k_2(\boldsymbol{x}, \boldsymbol{x}')$,
- $k(\boldsymbol{x}, \boldsymbol{x}') = ck_1(\boldsymbol{x}, \boldsymbol{x}')$, for all $c > 0$,
- $k(\boldsymbol{x}, \boldsymbol{x}') = k_1(\boldsymbol{x}, \boldsymbol{x}') + c$ for all $c > 0$,
- $k(\boldsymbol{x}, \boldsymbol{x}') = k_1(\boldsymbol{x}, \boldsymbol{x}')k_2(\boldsymbol{x}, \boldsymbol{x}')$,
- $k(\boldsymbol{x}, \boldsymbol{x}') = k_1(f(\boldsymbol{x}), f(\boldsymbol{x}'))$ for any function f

are also kernels. In particular, $k(\boldsymbol{x}, \boldsymbol{x}') = k_1(\boldsymbol{x}, \boldsymbol{x}')k_2(\boldsymbol{x}, \boldsymbol{x}')$ is kernel can be proved by verifying k is SPD, which follows directly from the Schur product theorem: $A \circ B$ is SPD if A and B are SPD. Here, $\circ$ denotes the Hadamard product: $(A \circ B)_{i,j} = A_{i,j}B_{i,j}$.

Lastly, we remark that for a specific problem, choosing appropriate kernels is highly non-trivial. One may need to incorporate the domain knowledge into the kernel design.

1.2.3. Representer theorem. Given a kernel $k(\cdot, \cdot)$, let $\Phi : \mathcal{X} \mapsto \mathcal{H}$ be the associated feature map. such that $k(\boldsymbol{x}, \boldsymbol{x}') = \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}') \rangle$. Consider a generalized feature-based model:

$$f(\boldsymbol{x}; \boldsymbol{\beta}) = \langle \boldsymbol{\beta}, \Phi(\boldsymbol{x}) \rangle,$$

[1]Also called inner-product kernels.

where the parameter $\boldsymbol{\beta} \in \mathcal{H}$. Let $\|\boldsymbol{\beta}\| = \sqrt{\langle \boldsymbol{\beta}, \boldsymbol{\beta} \rangle}$. Consider the following regularized model:

$$\hat{\mathcal{R}}_n(\boldsymbol{\beta}) = \frac{1}{2n} \sum_{i=1}^{n} \big(f(\boldsymbol{x}_i; \boldsymbol{\beta}) - y_i\big)^2 + \lambda r(\|\boldsymbol{\beta}\|), \tag{1.34}$$

where $r : [0, \infty) \mapsto [0, \infty)$ is a non-decreasing penalty function.

Theorem 1.11 (Representer theorem)**.** *Let $\hat{\boldsymbol{\beta}}$ the a minimizer of (1.34). Then, there must exist $a_1, \dots, a_n \in \mathbb{R}$ such that the function represented by $\hat{\boldsymbol{\beta}}$ has the form:*

$$f(\boldsymbol{x}; \hat{\boldsymbol{\beta}}) = \langle \hat{\boldsymbol{\beta}}, \Phi(\boldsymbol{x}) \rangle = \sum_{i=1}^{n} a_i k(\boldsymbol{x}_i, \boldsymbol{x}). \tag{1.35}$$

Moreover, $\hat{\boldsymbol{\beta}}$ can be reached in the linear span of $\Phi(\boldsymbol{x}_1), \dots, \Phi(\boldsymbol{x}_n)$.

Proof. Let $V_n = \operatorname{span}\{\Phi(\boldsymbol{x}_1), \dots, \Phi(\boldsymbol{x}_n)\} \subset \mathcal{H}$. For any $\boldsymbol{\beta} \in \mathcal{H}$, we can decompose it as follows $\boldsymbol{\beta} = \boldsymbol{\beta}_\parallel + \boldsymbol{\beta}_\perp$, where $\boldsymbol{\beta}_\parallel \in V_n, \boldsymbol{\beta}_\perp \in V_n^\perp$. Hence, $\|\boldsymbol{\beta}\|^2 = \|\boldsymbol{\beta}_\parallel\|^2 + \|\boldsymbol{\beta}_\perp\|^2$. Since $r(\cdot)$ is non-decreasing, we have

$$r(\|\boldsymbol{\beta}\|) \geq r(\|\boldsymbol{\beta}_\parallel\|). \tag{1.36}$$

On the other hand, for any $\boldsymbol{x}_i$,

$$f(\boldsymbol{x}_i; \boldsymbol{\beta}) = \langle \boldsymbol{\beta}, \Phi(\boldsymbol{x}_i) \rangle = \langle \boldsymbol{\beta}_\parallel, \Phi(\boldsymbol{x}_i) \rangle + \langle \boldsymbol{\beta}_\perp, \Phi(\boldsymbol{x}_i) \rangle = \langle \boldsymbol{\beta}_\parallel, \Phi(\boldsymbol{x}_i) \rangle, \tag{1.37}$$

where the last equality is due to $\boldsymbol{\beta}_\perp \in V_n^\perp$. Combining (1.36) and (1.37), we have $\hat{\mathcal{R}}_n(\hat{\boldsymbol{\beta}}) \geq \hat{\mathcal{R}}_n(\hat{\boldsymbol{\beta}}_\parallel)$. Let $\hat{\boldsymbol{\beta}}_\parallel = \sum_{i=1}^n a_i \Phi(\boldsymbol{x}_i)$. Then, the function represented can be written as

$$f(\boldsymbol{x}; \hat{\boldsymbol{\beta}}) = \langle \hat{\boldsymbol{\beta}}_\parallel, \Phi(\boldsymbol{x}) \rangle = \sum_{i=1}^{n} a_i \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}) \rangle = \sum_{i=1}^{n} a_i k(\boldsymbol{x}_i, \boldsymbol{x}).$$

□

This theorem generalizes Theorem 1.7 to general feature maps and regularizations, and in particular, it works for the case of $m = \infty$. This theorem allows us to transform the infinite-dimensional optimization problem (1.34) into a finite dimensional problem. In the literature, this theorem is called the *representer theorem*, which plays a fundamental role in kernel methods.

Reduced regularized models. By the representer theorem, to solve (1.34), we only need to consider $\boldsymbol{\beta} = \sum_{j=1}^n a_j \Phi(\boldsymbol{x}_j)$. In this case,

$$f(\boldsymbol{x}; \boldsymbol{\beta}) = \sum_{j=1}^{n} a_j k(\boldsymbol{x}_j, \boldsymbol{x})$$

and the corresponding ridge penality:

$$\|\boldsymbol{\beta}\|^2 = \left\langle \sum_{i=1}^n a_i \Phi(\boldsymbol{x}_i), \sum_{j=1}^n a_j \Phi(\boldsymbol{x}_j) \right\rangle = \sum_{i,j=1}^n a_i a_j k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{a}^T K \boldsymbol{a},$$

which surprisingly is a squared ℓ_2 norm weighted by the kernel matrix. As a comparison, the norm is un-weighted in the original space.

The infinite-dimensional problem (1.34) becomes

$$\begin{aligned} \hat{\mathcal{R}}_n(\boldsymbol{a}) &= \frac{1}{2n} \sum_i \left(\sum_j a_j k(\boldsymbol{x}_j, \boldsymbol{x}_i) - y_i \right)^2 + \lambda\, r\left(\sqrt{\sum_{i,j} a_i a_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)} \right) \\ &= \frac{1}{2n} \|K\boldsymbol{a} - \boldsymbol{y}\|_2^2 + \lambda\, r\left(\sqrt{\boldsymbol{a}^T K \boldsymbol{a}} \right). \end{aligned} \tag{1.38}$$

Note that the kernel matrix $K = (k(\boldsymbol{x}_i, \boldsymbol{x}_j))$ is often referred to as the Gram matrix as well.

The popular kernel ridge regression (KRR) corresponds to $r(t) = t^2/2$, where the reduced model becomes

$$\hat{\mathcal{R}}_n(\boldsymbol{a}) = \frac{1}{2n} \|K\boldsymbol{a} - \boldsymbol{y}\|_2^2 + \frac{\lambda}{2} \boldsymbol{a}^T K \boldsymbol{a}.$$

Similar to the standard ridge regression, KRR has a closed-form expression:

$$\hat{\boldsymbol{a}} = \left(\frac{1}{n} K + \lambda I_n \right)^{-1} \boldsymbol{y}.$$

Lastly, we remark that for a specific kernel, the associated feature maps are not necessarily unique. But the representer theorem shows that the solutions only depend on the kernel and is independent of the specific choice of feature maps.

1.3. Random feature approximations

Let $(\Omega, \mathcal{F}, \pi)$ be a probability space and $\varphi : \mathcal{X} \times \Omega \mapsto \mathbb{R}$ is a parametric feature. The hypothesis of a random feature models is given by

$$f(\boldsymbol{x}; \boldsymbol{\beta}) = \frac{1}{m} \sum_{j=1}^m \beta_j \varphi(\boldsymbol{x}; \boldsymbol{\omega}_j), \tag{1.39}$$

with $\{\boldsymbol{\omega}_j\}_{j=1}^m$ be iid samples drawn from π. Here, $\varphi(\cdot; \boldsymbol{\omega})$ is called the "random feature" since $\{\boldsymbol{\omega}_j\}_{j=1}^m$ are randomly sampled.

Ridge regression with random features is given by

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^m} \frac{1}{2n} \sum_{i=1}^n \left(\frac{1}{m} \sum_{j=1}^m \beta_j \varphi(\boldsymbol{x}_i; \boldsymbol{\omega}_j) - y_i \right)^2 + \frac{\lambda}{m} \|\boldsymbol{\beta}\|_2^2. \tag{1.40}$$

Here, the $1/m$ factor is added to enforce $\frac{1}{m}\|\boldsymbol{\beta}\|_2^2 = O(1)$.

According to the representer theorem 1.11, (1.40) is equivalent to the kernel ridge regression with the kernel:

$$k_m(\boldsymbol{x}, \boldsymbol{x}') := \frac{1}{m}\sum_{j=1}^{m} \varphi(\boldsymbol{x}; \boldsymbol{\omega}_j)\varphi(\boldsymbol{x}'; \boldsymbol{\omega}_j). \tag{1.41}$$

Here, we impose a scaling factor $1/m$ into the expression of k_m. This manipulation allows us to take the limit, while does not change the function represented. As $m \to \infty$, k_m converges to

$$k(\boldsymbol{x}, \boldsymbol{x}') := \mathbb{E}_{\boldsymbol{\omega}\sim\pi}[\varphi(\boldsymbol{x}; \boldsymbol{\omega})\varphi(\boldsymbol{x}'; \boldsymbol{\omega})], \tag{1.42}$$

due to $\{\boldsymbol{\omega}_j\} \overset{iid}{\sim} \pi$.

Notice that (1.41) is a Monte-Carlo approximation of (1.42), and the standard Monte-Carlo estimate tells us that

$$k_m(\boldsymbol{x}, \boldsymbol{x}') - k(\boldsymbol{x}, \boldsymbol{x}') \sim \frac{\mathrm{Var}_{\boldsymbol{\omega}}[\varphi(\boldsymbol{x}; \boldsymbol{\omega})\varphi(\boldsymbol{x}'; \boldsymbol{\omega})]}{\sqrt{m}}. \tag{1.43}$$

In this way, the random feature model can be viewed as a Monte-Carlo/random approximations of kernel methods.

Random feature approximations are usually applied for the large-scale dataset, where the sample size n is huge, e.g, $n = 10^6$. For standard kernel methods, the memory to store the Gram matrix is $O(n^2)$ and the computational cost to invert the Gram matrix is roughly $O(n^3)$. These complexities are often unacceptable for large-scale dataset. By contrast, with random feature approximations, the storage and computational cost are $O(mn)$ and $O(m^2n)$, respectively. This is much smaller than that of standard kernel methods when $m \ll n$.

In other words, as long as a kernel can be expressed in the expectation form (1.42), we can apply random feature approximations and use the resulting random feature model (1.39) to solve the original problem. Then a natural question is: What kind of kernel functions can can be written in the form (1.42)? Next, we provide an answer to this question for translation-invariant kernels.

1.3.1. Random Fourier features.

Here, we introduce the popular random Fourier features:

$$\varphi(\boldsymbol{x}; \boldsymbol{\omega}) = e^{i\boldsymbol{x}^T\boldsymbol{\omega}}.$$

Bochner theorem given below shows that any translation invariant kernel can be approximated by the random Fourier features.

Theorem 1.12 (Bochner)**.** *A continuous kernel $k(\boldsymbol{x}, \boldsymbol{x}') = \kappa(\boldsymbol{x} - \boldsymbol{x}')$ on $\mathbb{R}^d$ is semi-positive definite if and only if $\kappa(\cdot)$ is the Fourier transform of a non-negative measure.*

Assuming $\kappa(\cdot)$ to be the Fourier transform of a non-negative measure π, we have

$$
\begin{aligned}
k(\boldsymbol{x}, \boldsymbol{x}') = \kappa(\boldsymbol{x} - \boldsymbol{x}') &= \int_{\mathbb{R}^d} \pi(\boldsymbol{\omega}) e^{i\boldsymbol{\omega}^T(\boldsymbol{x}-\boldsymbol{x}')} d\boldsymbol{\omega} \\
&= \int_{\mathbb{R}^d} \varphi(\boldsymbol{x}, \boldsymbol{\omega}) \overline{\varphi(\boldsymbol{x}', \boldsymbol{\omega})} d\pi(\boldsymbol{\omega}) \qquad (1.44) \\
&=: \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}) \rangle_{L^2(\pi)},
\end{aligned}
$$

where $\Phi : \mathcal{X} \mapsto L^2(\pi)$ with $\Phi(\boldsymbol{x}) = \varphi(\boldsymbol{x}; \cdot)$. Therefore, $\kappa(\boldsymbol{x} - \boldsymbol{x}')$ must be a kernel. Here, we only show a proof of the direction that if $\kappa(\cdot)$ is the Fourier transform of a non-negative measure, then $\kappa(\boldsymbol{x} - \boldsymbol{x}')$ must be a SDP kernel. For a full proof, we refer to [**Rud17**].

Table 1 lists some popular translation-invariant kernels and their Fourier transforms, which allows us to approximate them with random Fourier features.

kernel name	$k(\boldsymbol{z})$	$\pi(\boldsymbol{\omega})$
Gaussian	$e^{-\Vert\boldsymbol{z}\Vert_2^2/2}$	$\frac{1}{(2\pi)^{d/2}} e^{-\frac{\Vert\boldsymbol{\omega}\Vert_2^2}{2}}$
Laplace	$e^{-\Vert\boldsymbol{z}\Vert_1}$	$\prod_{j=1}^d \frac{1}{\pi(1+\omega_j^2)}$

Table 1. Some popular translation-invariant kernels and their Fourier transforms. See [**RR07**] for more examples.

Let us examine the specific Gaussian kernel. First, the Fourier transform tells us that

$$
e^{-\frac{\Vert\boldsymbol{x}-\boldsymbol{y}\Vert_2^2}{2}} = \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} e^{-\frac{\Vert\boldsymbol{\omega}\Vert_2^2}{2}} e^{i(\boldsymbol{x}-\boldsymbol{y})\cdot\boldsymbol{\omega}} d\boldsymbol{\omega}. \qquad (1.45)
$$

Therefore,

$$
\begin{aligned}
e^{-\frac{\Vert\boldsymbol{x}-\boldsymbol{y}\Vert_2^2}{2}} &= \mathbb{E}_{\boldsymbol{\omega}\sim\mathcal{N}(0,I_d)}[e^{i\boldsymbol{\omega}^T\boldsymbol{x}} e^{-i\boldsymbol{\omega}^T\boldsymbol{y}}] \\
&= \mathbb{E}_{\boldsymbol{\omega}\sim\mathcal{N}(0,I_d)}[\cos(\boldsymbol{\omega}^T\boldsymbol{x})\cos(\boldsymbol{\omega}^T\boldsymbol{y}) + \sin(\boldsymbol{\omega}^T\boldsymbol{x})\sin(\boldsymbol{\omega}^T\boldsymbol{y})],
\end{aligned}
$$

where the second equality is due to that $k(\cdot, \cdot)$ is real. In practice, instead of using $e^{i\boldsymbol{\omega}^T\boldsymbol{x}}$ as the feature, one often use

$$
\varphi(\boldsymbol{x}; \boldsymbol{\omega}) = \begin{pmatrix} \cos(\boldsymbol{\omega}^T\boldsymbol{x}) \\ \sin(\boldsymbol{\omega}^T\boldsymbol{x}) \end{pmatrix}
$$

to keep all the operations in the real space. In a summary, the random feature approximation of Gaussian kernel is given by

$$e^{-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|_2^2}{2}} \approx \frac{1}{m}\sum_{j=1}^{m} \varphi(\boldsymbol{x};\boldsymbol{\omega}_j)\varphi(\boldsymbol{y};\boldsymbol{\omega}_j)$$

with $\{\boldsymbol{\omega}_j\} \overset{iid}{\sim} \mathcal{N}(0, I_d)$.

Bibliography

[BPC11] Stephen Boyd, Neal Parikh, and Eric Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Now Publishers Inc, 2011.

[CRT06] Emmanuel J Candès, Justin Romberg, and Terence Tao, *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*, IEEE Transactions on information theory **52** (2006), no. 2, 489–509.

[CX20] Lin Chen and Sheng Xu, *Deep neural tangent kernel and Laplace kernel have the same RKHS*, International Conference on Learning Representations, 2020.

[GYK$^+$20] Amnon Geifman, Abhay Yadav, Yoni Kasten, Meirav Galun, David Jacobs, and Ronen Basri, *On the similarity between the Laplace and neural tangent kernels*, arXiv preprint arXiv:2007.01580 (2020).

[RR07] Ali Rahimi and Benjamin Recht, *Random features for large-scale kernel machines.*, NIPS, vol. 3, Citeseer, 2007, p. 5.

[Rud17] Walter Rudin, *Fourier analysis on groups*, Courier Dover Publications, 2017.

[Ver18] Roman Vershynin, *High-dimensional probability: An introduction with applications in data science*, vol. 47, Cambridge university press, 2018.