OPTIMIZATION FOR MACHINE LEARNING

Recall that ERM rule aims:

$$\hat{f}_{ERM} \in \underset{f \in \mathcal{H}}{\arg\min} \ \hat{\mathcal{R}}_S(f) := \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

Parameterization: This is the traditional methodology in statistics.

For a compact set $\Theta \subset \mathbb{R}^d$, a class of parameterized predictors is considered:

$$\mathcal{H}_\Theta = \{ x \mapsto f_\theta(x) : \theta \in \Theta, \ x \in \mathcal{X} \}.$$

Then,

$$\hat{\theta}_{ERM} \in \underset{\theta \in \Theta}{\arg\min} \ \hat{\mathcal{R}}_S(f_\theta),$$

leads to the empirical risk minimizer $f_{\hat{\theta}_{ERM}}$.

EXAMPLE (ReLU neural networks)

$\Theta \subset \mathbb{R}^{m \times d} \times \mathbb{R}^m$, $\quad \mathcal{X} \subset \mathbb{R}^d$, $\quad \mathcal{Y} = \mathbb{R}$

$$\mathcal{H}_\Theta = \left\{ x \mapsto \sum_{i=1}^{m} c_i \, \sigma(w_i^T x) \ : (W, c) \in \Theta, \ x \in \mathbb{R}^d \right\}$$

where $\sigma(z) = \max\{0, z\}$ is the so-called ReLU activation funct.,

$$W = \begin{bmatrix} — w_1^T — \\ \vdots \\ — w_m^T — \end{bmatrix} \in \mathbb{R}^{m \times d}, \qquad \longrightarrow \text{hidden-layer weights}$$

$$c \in \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix} \in \mathbb{R}^m \qquad \longrightarrow \text{output layer weights.}$$

$m \in \mathbb{Z}_+$ is the network-width.

The trainable (or learnable) parameter is $\theta = (W, c) \in \Theta$.

Thus, given a training set $S = \{ (x_i, y_i) \in \mathcal{X} \times \mathcal{Y} : i = 1, 2, \ldots, n \}$,

$$\hat{\theta}_{ERM} \in \underset{\theta \in \Theta}{\arg\min} \ \frac{1}{n} \sum_{i=1}^{n} \left(y_i - f_\theta(x_i) \right)^2,$$

$$f_\theta(x) := \sum_{i=1}^{m} c_i \, \sigma(w_i^T x).$$

Vital question: Can we find $\hat{\theta}_{ERM}$ in a computationally efficient way?

Thus, we cast ERM as an optimization problem:

$$g(\theta) \triangleq \hat{R}_S(f_\theta), \quad \theta \in \Theta$$

$$\hat{\theta}_{ERM} \in \underset{\theta \in \Theta}{\operatorname{argmin}}\ g(\theta)$$

Uniform Grid Method and Curse of Dimensionality:

Consider $\Theta = \{\theta \in \mathbb{R}^p : 0 \leq \theta_i \leq 1, \ i = 1, 2, \ldots, p\}$ [hypercube].

ALG (Uniform Grid)

input: $q \in \mathbb{Z}_+$

Step 0: Form q^p points:

$$\theta_\alpha = \left(\frac{2i_1 - 1}{2q}, \frac{2i_2 - 1}{2q}, \ldots, \frac{2i_p - 1}{2q}\right), \quad \alpha = (i_1, i_2, \ldots, i_p) \in \{1, 2, \ldots, q\}^p.$$

Step 1: Find the optimal point in the grid:

$$\alpha^* \in \underset{\alpha \in \{1, \ldots, q\}^p}{\operatorname{argmin}}\ g(\theta_\alpha)$$

Step 2: Return $\left(\theta_{\alpha^*}, g(\theta_{\alpha^*})\right)$.

Assumption (Lipschitz continuity)

$g : \mathbb{R}^p \to \mathbb{R}$ is L-Lipschitz continuous on Θ:

$$|g(\theta) - g(\theta')| \leq L \cdot \|\theta - \theta'\|_\infty, \quad \forall \theta, \theta' \in \Theta.$$

THEOREM (Oracle complexity of Uniform Grid)

Let g^* be a global optimum value of g. Then,

$$g(\theta_{\alpha^*}) - g^* \leq \frac{L}{2q}, \quad \text{for any } q \in \mathbb{Z}_+.$$

Then, $\left(\left\lfloor \frac{L}{2\varepsilon} \right\rfloor + 1\right)^p$ function evaluations should be performed

to achieve $g(\theta_{\alpha^*}) - g^* \leq \varepsilon$ for $\varepsilon > 0$.

Proof: For a multi-index $\alpha = (i_1, i_2, \ldots, i_p)$, define

$$X_\alpha = \left\{\theta \in \mathbb{R}^p : \|\theta - \theta_\alpha\|_\infty \leq \frac{1}{2q}\right\}.$$

Then, $\bigcup_{\alpha \in \{1, \ldots, q\}^p} X_\alpha = \Theta$. Let $\theta^* \in \Theta$ be the globally optimum solution. Then, $\exists \bar{\alpha} \in \{1, \ldots, q\}^p$ s.t. $\theta^* \in X_{\bar{\alpha}}$.

$$\Rightarrow \|\theta^* - \theta_{\bar{\alpha}}\|_\infty \leq \frac{1}{2q}.$$

$$\Rightarrow g(\theta_{\alpha^*}) - g^* \leq g(\theta_{\bar{\alpha}}) - g^* \leq \|\theta_{\bar{\alpha}} - \theta^*\|_\infty \leq \frac{1}{2q}.$$

Remark (Dimension)

For ε-optimality, a naive optimization method needs

$$O\left(\frac{1}{\varepsilon^p}\right) \text{ computations if there are } p$$

parameters. In our ERM search, this would mean

$$O\left(\frac{n}{\varepsilon^p}\right) \text{ computations.}$$

In modern deep learning, p is huge. Especially in the overparameterized regime, where $p \gg n$, $O\left(\frac{n}{\varepsilon^p}\right)$ would imply completely impractical optimization.

For $\varepsilon = 0.001$ error, $O(n \cdot 10^{3p})$ computations.

Solutions: For tractability, inspired by convex optimization theory, local iterative methods (e.g., gradient descent) are used. As we will see, they will lead to dimension-free (i.e., independent of p) iteration complexities.

Furthermore, since n is very large in deep learning, stochastic gradient methods are developed.

As we will study, these will bring elimination of n in complexity bounds.