Lecture 5 Curve Fitting and Bias-Variance Decomposition

Key Concepts/Topics of This Lecture
parameter estimate for the linear fitting function $f_{\vec{\theta}}(x) = a\,x + b$ from data
parabolic loss function $J(\vec{\theta}) = 2^{-1}\sum[f_{\vec{\theta}}(x^{(i)}) - y^{(i)}]^2$ and its optimization
goodness of learning model $\longleftrightarrow$ decomposition of bias and variance
penalty term $J \to J + \lambda g$, data and belief in data, avoidance of singularity
normal equation $\vec{\Phi}^\top\vec{\Phi}\mathbf{w} = \vec{\Phi}^\top\mathbf{y}$ and its stochastic gradient descent search

§5.1 Linear Curve Fitting

Assume that we are given m data points $(x^{(i)}, y^{(i)})$ with $i = 1 \sim m$. The physical or underlying relation between $x^{(i)}$ and $y^{(i)}$ is assumed to be linear. A typical example is the relation between velocity v and acceleration a, given by $v = a\,t$.

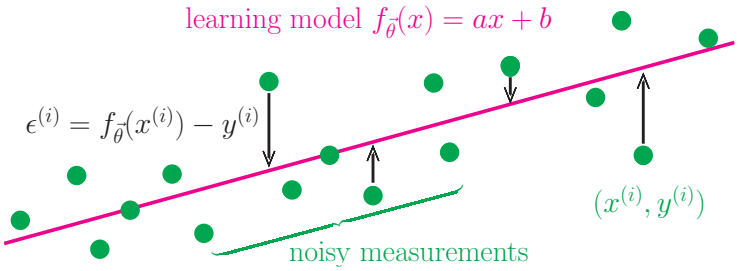

FIG. 5.1: Sketch of linear regression. The preliminary error for a data sample $x^{(i)}$ is defined as the difference between the model prediction $f_{\vec{\theta}}(x^{(i)})$ and the measurement $y^{(i)}$.

In practice, due to measurement uncertainties, the observed or experimental relation deviates from an exact linear form. Nevertheless, linear regression can still be applied to extract the model parameters from noisy data. We assume a linear model of the form $f_{\vec{\theta}}(x) = a\,x + b$, where a and b are parameters to be determined. These parameters are collectively denoted by $\vec{\theta} = (a, b)$. To determine $\vec{\theta}$, we minimize the discrepancy between the model prediction $f_{\vec{\theta}}(x^{(i)})$ and the measurement $y^{(i)}$.

A commonly used measure of error is the squared loss, $(f_{\vec{\theta}}(x^{(i)}) - y^{(i)})^2$. The quantity $f_{\vec{\theta}}(x^{(i)}) - y^{(i)}$ is referred to as the algebraic distance, which can be either positive or negative (see FIG. 5.1). The total loss is defined as

$$J(\vec{\theta}) = \frac{1}{2}\sum_{i=1}^{m}\left[f_{\vec{\theta}}(x^{(i)}) - y^{(i)}\right]^2. \tag{5.1}$$

The factor $1/2$ is introduced for convenience and does not affect the minimization. Note that J is a function of $\vec{\theta}$, or equivalently of a and b. In Lecture 11, we will revisit the loss function from a Bayesian perspective.

To determine a and b, we minimize J. Since J is a convex function (similar to the parabola x^2), the minimization reduces to solving $\partial J/\partial a = 0$ and $\partial J/\partial b = 0$. These equations yield the optimal parameters a^* and b^*. Expanding $J(\vec{\theta})$, we obtain

$$J(\vec{\theta}) = \frac{m}{2}\left[\langle x^2\rangle a^2 + b^2 + \langle y^2\rangle + 2\langle x\rangle a\, b - 2\langle x\, y\rangle a - 2\langle y\rangle b\right], \tag{5.2}$$

where the sample averages are defined as [∗5-1∗],

$$\langle x\rangle = \frac{1}{m}\sum_{i=1}^{m} x^{(i)},\ \ \langle y\rangle = \frac{1}{m}\sum_{i=1}^{m} y^{(i)}, \tag{5.3}$$

$$\langle x^2\rangle = \frac{1}{m}\sum_{i=1}^{m} x^{(i),2},\ \ \langle y^2\rangle = \frac{1}{m}\sum_{i=1}^{m} y^{(i),2},\ \ \langle x\, y\rangle = \frac{1}{m}\sum_{i=1}^{m} x^{(i)} y^{(i)}, \tag{5.4}$$

which can be computed directly from the data. After straightforward calculations, one finds

$$a^* = \frac{\langle x\, y\rangle - \langle x\rangle\langle y\rangle}{\langle x^2\rangle - \langle x\rangle^2},\ \ b^* = \frac{\langle x^2\rangle\langle y\rangle - \langle x\rangle\langle x\, y\rangle}{\langle x^2\rangle - \langle x\rangle^2}. \tag{5.5}$$

In particular, if $y \sim x$, the numerator of b^* satisfies $\langle x\rangle\langle x^2\rangle - \langle x^2\rangle\langle x\rangle = 0$.

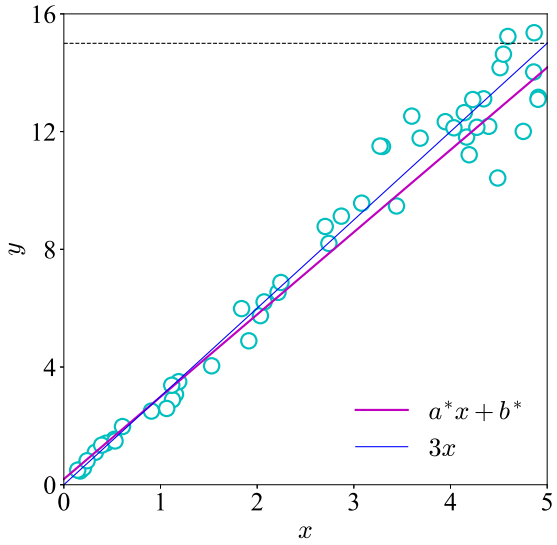

FIG. 5.2: Simulated samples, the fitted line $a^*x + b^*$, and the underlying physical model. The number of data points is $m = 50$.

The coefficient a^* can be rewritten as

$$a^* = \mathrm{cov}[x, y]/\mathrm{var}[x], \tag{5.6}$$

and the optimal prediction becomes

$$y^* = \mathrm{E}[y] + \frac{\mathrm{cov}[x,y]}{\mathrm{var}[x]}[x - \mathrm{E}[x]]. \tag{5.7}$$

The mean squared error (MSE) is defined as

$$\Delta^* = \mathrm{E}\big[y - y^*\big]^2 = \mathrm{var}[y]\big[1 - \rho^2[x,y]\big], \tag{5.8}$$

where $\rho[x,y]$ denotes the correlation coefficient. A stronger (absolute) correlation leads to a smaller MSE, i.e., if $|\rho(x,y)| = 1$, $\Delta^* = 0$. Conversely, if x and y are uncorrelated, $\Delta^* = \mathrm{var}[y]$ and $y^* = \mathrm{E}[y]$.

We now illustrate this with a simulation. Suppose the true model is $y = 3x$, i.e., $a = 3$ and $b = 0$. The data are generated via $y^{(i)} = a' x^{(i)}$, where $a' = 3 \pm \Delta$, and $\Delta \sim \mathcal{N}(0, \sigma^2)$ is Gaussian noise. Thus $a' = \mathcal{N}(3, \sigma^2)$. We fix $\sigma^2 = 0.3$ and draw $x^{(i)} \sim \mathrm{Unif}[0,5]$. FIG. 5.2 shows the simulated data ($m = 50$), the fitted model, and the true model. The learned model $a^* x + b^*$ deviates slightly from the true model $3x$, and in particular b^* is not exactly zero.

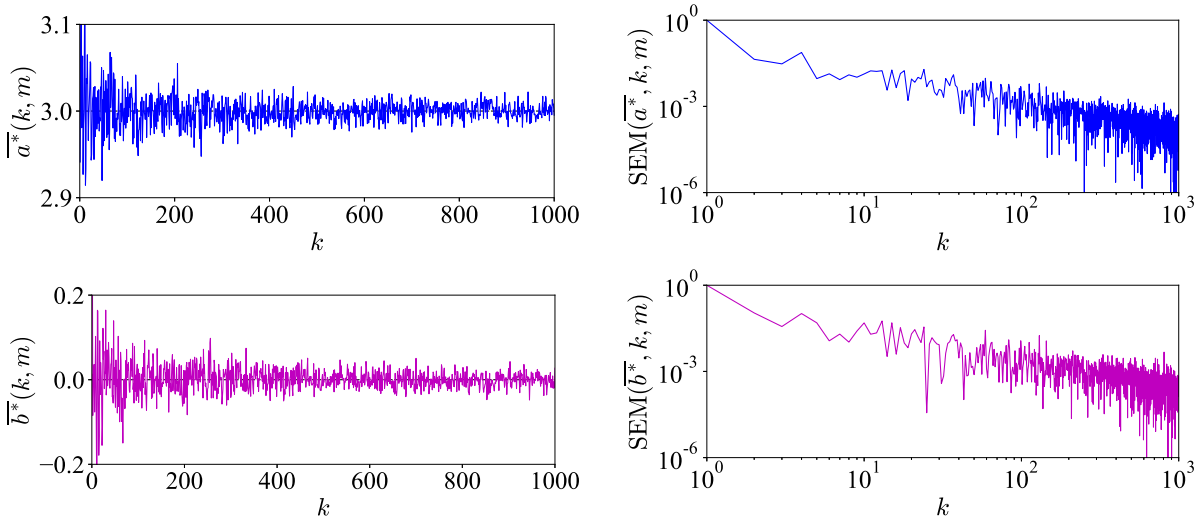

FIG. 5.3: k-dependence of $\overline{a^*}(k,m)$, $\overline{b^*}(k,m)$, SEM$(\overline{a^*},k,m)$, and SEM$(\overline{b^*},k,m)$, with $m = 10$ fixed.

Repeating the simulation k times yields k independent estimates of a^* and b^*. Their averages are

$$\overline{a^*}(k,m) \equiv \frac{1}{k}\sum_{j=1}^{k} a^{*,(j)}, \quad \overline{b^*}(k,m) \equiv \frac{1}{k}\sum_{j=1}^{k} b^{*,(j)}. \tag{5.9}$$

As shown in FIG. 5.3, these averages approach the true values as k increases. We also define the standard error of the mean (SEM) as

$$\mathrm{SEM}(\overline{a^*},k,m) = \sqrt{\frac{1}{k}\frac{1}{k-1}\sum_{j=1}^{k}\big(a^{*,(j)} - \overline{a^*}\big)^2}, \quad \mathrm{SEM}(\overline{b^*},k,m) = \sqrt{\frac{1}{k}\frac{1}{k-1}\sum_{j=1}^{k}\big(b^{*,(j)} - \overline{b^*}\big)^2}. \tag{5.10}$$

The lower panels of FIG. 5.3 show the dependence of SEM on k (with $m = 10$ fixed). In the log-log plot, the overall trend appears approximately linear.

EXERCISE 5-1. Does SEM of a^* depend on k via a power law in the log-log plot?

§5.2 Basic Concepts of Learning Algorithms

The linear learning model $f_{\vec{\theta}}(x) = a\,x + b$ is simple and easy to implement. However, it is often too restrictive to capture more complex features present in real data, such as irregularity and/or nonlinearity (see FIG. 5.4 for an illustration). In such situations, a linear model is generally inadequate, and more advanced learning techniques are required to describe these nonlinear characteristics.

To proceed, we introduce several fundamental concepts. First, there exists a physical model, denoted by $f_{\text{phys}}(\mathbf{x})$, where the input $\mathbf{x}$ is, in general, a vector. This physical model is unknown a priori and may be highly complicated. Although the physical model $f_{\text{phys}}(\mathbf{x})$ is not directly accessible, it generates observable data, which inevitably contain measurement noise. We denote the data samples as $(\mathbf{x}^{(i)}, y^{(i)})$. Due to noise, the observed output satisfies $y^{(i)} \neq f_{\text{phys}}(\mathbf{x}^{(i)})$ in general. For simplicity, we assume the output is scalar. In addition to the physical model, we introduce a learning model, denoted by $f_{\mathbf{w}}(\mathbf{x})$, where $\mathbf{w}$ represents a set of parameters characterizing the model, such as the $\vec{\theta} = (a, b)$ in the linear curve fitting problem. Given a fixed learning model, one can make predictions for each input $\mathbf{x}^{(i)}$ in the form $f_{\mathbf{w}}(\mathbf{x}^{(i)})$. In this framework, the learning model serves as an effective approximation to the physical model, which is otherwise too complex to handle directly.

In general, the prediction $f_{\mathbf{w}}(\mathbf{x}^{(i)})$ differs from the measured output $y^{(i)}$. The central task in machine learning and data analysis is therefore to minimize the discrepancy between predictions and measurements. This naturally leads to the introduction of an error (or cost/loss) function that quantifies this difference.

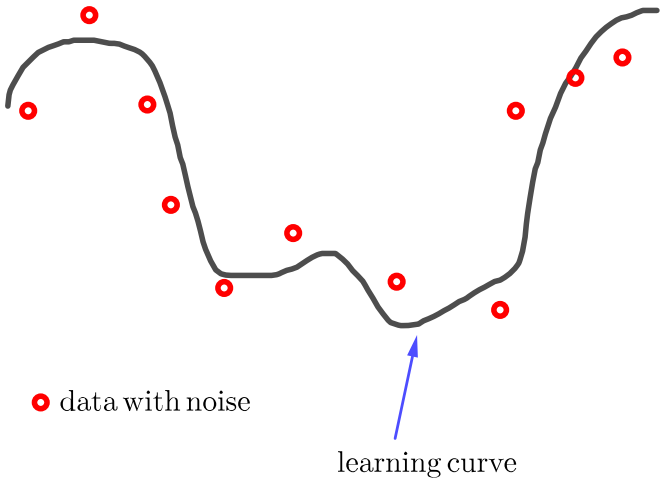

FIG. 5.4: Sketch of a nonlinear curve fitting problem.

A widely used choice of error function is

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m}\left[f_{\mathbf{w}}(\mathbf{x}^{(i)}) - y^{(i)}\right]^2, \tag{5.11}$$

which represents the sum of squared differences between predictions and measurements. This form has already been encountered in the linear model (see (5.1)) and is known as the least-squares (LS) method. As before, $J(\mathbf{w})$ is a function of the model parameters $\mathbf{w}$, rather than of the data $(\mathbf{x}^{(i)}, y^{(i)})$. Furthermore, model is often simpler than reality, as shown in FIG. 5.5.

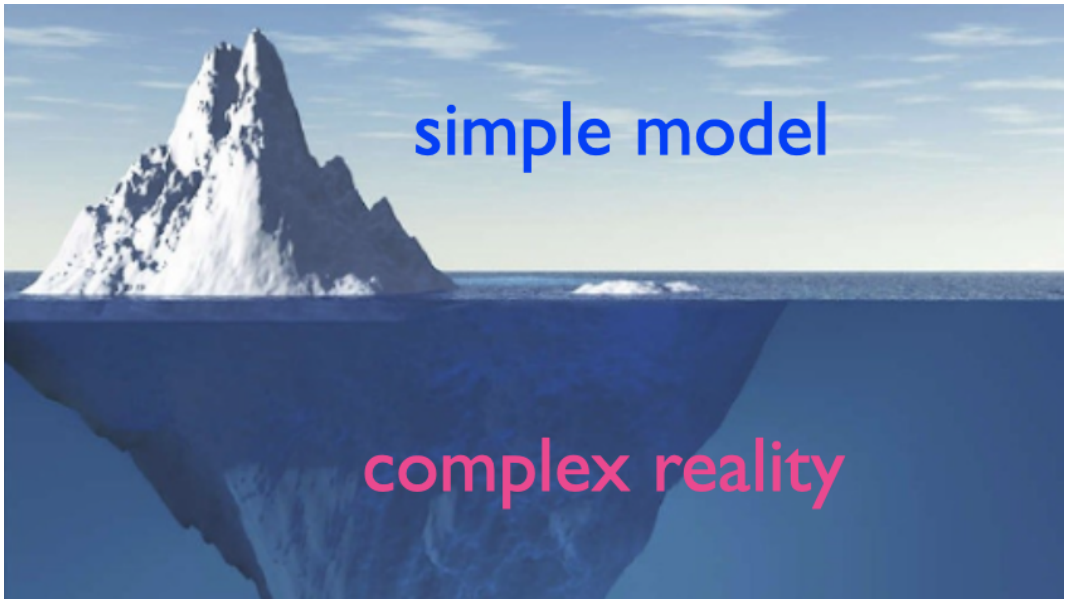

FIG. 5.5: Model is often simpler than reality.

The next step is to minimize $J(\mathbf{w})$. Since $J(\mathbf{w})$ is typically a convex function of $\mathbf{w}$, the minimization reduces to solving $\nabla_{\mathbf{w}} J \equiv \partial J(\mathbf{w})/\partial \mathbf{w} = 0$, or equivalently,

$$\partial J(\mathbf{w})/\partial w_j = 0, \quad j = 0, 1, 2, \cdots, n, \tag{5.12}$$

where the model contains $n+1$ parameters, namely w_0 through w_n. The second derivative of the loss function with respect to w_j is

$$\begin{aligned} \frac{\partial^2 J}{\partial w_j^2} &= \frac{\partial}{\partial w_j}\frac{\partial}{\partial w_j}\left[\frac{1}{2}\sum_{i=1}^{m}\left[f_{\mathbf{w}}(\mathbf{x}^{(i)}) - y^{(i)}\right]^2\right] \\ &= \sum_{i=1}^{m}\left(\frac{\partial}{\partial w_j} f_{\mathbf{w}}(x^{(i)})\right)^2 + \sum_{i=1}^{m}\left(f_{\mathbf{w}}(x^{(i)}) - y^{(i)}\right)\frac{\partial^2}{\partial w_j^2} f_{\mathbf{w}}(x^{(i)}). \end{aligned} \tag{5.13}$$

If the learning model is linear in the parameters, the second term vanishes at the optimal point, yielding

$$\frac{\partial^2 J}{\partial w_j^2} = \sum_{i=1}^{m}\left(\frac{\partial}{\partial w_j} f_{\mathbf{w}}(x^{(i)})\right)^2, \tag{5.14}$$

which is manifestly positive. Even for nonlinear models, one can show that $\partial^2 J/\partial w_j^2 > 0$, for example using the normal equation discussed shortly after in this lecture. It is important to distinguish between the symbols n and m, which denote the number of model parameters and the number of data points, respectively.

For scalar input data (x instead of $\mathbf{x}$), a commonly used and flexible model is the polynomial of order n, given by

$$f_{\mathbf{w}}(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_n x^n = \sum_{j=0}^{n} w_j x^j, \tag{5.15}$$

where the parameter set consists of $n+1$ coefficients w_0 through w_n. The process of determining these parameters from the data $(x^{(i)}, y^{(i)})$ is referred to as learning from data. Although $f_{\mathbf{w}}(x)$ is nonlinear in the variable x, it remains linear in the parameters w_j. For this reason, it is still classified as a linear model.

§5.3 Trade-off between Bias and Variance

We now study the nonlinear curve fitting problem in detail to illustrate several key aspects of machine learning, namely how one can "learn from data" and what is actually being learned. According to Eq. (5.12), the model parameters $w_0, w_1, w_2, \cdots, w_n$ satisfy the following system of equations,

$$\begin{cases} \langle 1\rangle w_0 + \langle x\rangle w_1 + \langle x^2\rangle w_2 + \cdots + \langle x^n\rangle w_n = \langle y\rangle, \\ \langle x\rangle w_0 + \langle x^2\rangle w_1 + \langle x^3\rangle w_2 + \cdots + \langle x^{n+1}\rangle w_n = \langle xy\rangle, \\ \vdots, \\ \langle x^n\rangle w_0 + \langle x^{n+1}\rangle w_1 + \langle x^{n+2}\rangle w_2 + \cdots + \langle x^{2n}\rangle w_n = \langle x^n y\rangle, \end{cases} \tag{5.16}$$

where

$$\langle 1\rangle = \frac{1}{m}\sum_{i=1}^{m} 1 = 1, \quad \langle x^k\rangle = \frac{1}{m}\sum_{i=1}^{m} x^{(i),k} = \frac{1}{m}\left(x^{(1),k} + x^{(2),k} + \cdots + x^{(m),k}\right), \tag{5.17}$$

$$\langle x^k y\rangle = \frac{1}{m}\sum_{i=1}^{m} x^{(i),k} y^{(i)} = \frac{1}{m}\left(x^{(1),k} y^{(1)} + x^{(2),k} y^{(2)} + \cdots + x^{(m),k} y^{(m)}\right). \tag{5.18}$$

To see how these equations arise, we explicitly compute

$$\begin{aligned} \frac{\partial J}{\partial w_j} &= \sum_{i=1}^{m}\left(f_{\mathbf{w}}(x^{(i)}) - y^{(i)}\right)\frac{\partial}{\partial w_j} f_{\mathbf{w}}(x^{(i)}) \\ &= \sum_{i=1}^{m} f_{\mathbf{w}}(x^{(i)}) x^{(i),j} - \sum_{i=1}^{m} y^{(i)} x^{(i),j} \\ &= \sum_{i=1}^{m}\sum_{j'=0}^{n} w_{j'} x^{(i),j'+j} - \sum_{i=1}^{m} y^{(i)} x^{(i),j} \\ &= \sum_{j'=0}^{n} w_{j'} \sum_{i=1}^{m} x^{(i),j'+j} - \sum_{i=1}^{m} y^{(i)} x^{(i),j} \\ &= \sum_{j'=0}^{n} w_{j'} \langle x^{j'+j}\rangle - \langle x^j y\rangle, \end{aligned} \tag{5.19}$$

and the related equations are obtained by setting this derivative to zero,

$$\boxed{\sum_{j'=0}^{n} w_{j'} \langle x^{j'+j} \rangle = \langle x^j y \rangle,} \tag{5.20}$$

with $j = 0 \sim n$.

As a special case, for $n = 1$ (linear fitting), Eq. (5.16) reduces to

$$\langle 1 \rangle w_0 + \langle x \rangle w_1 = \langle y \rangle, \;\; \langle x \rangle w_0 + \langle x^2 \rangle w_1 = \langle x\, y \rangle. \tag{5.21}$$

Furthermore, Eq. (5.16) can be written in a compact matrix form as $\mathbf{Fw} = \mathbf{G}$, where

$$\boxed{\mathbf{F} = \begin{pmatrix} \langle 1 \rangle & \langle x \rangle & \cdots & \langle x^n \rangle \\ \langle x \rangle & \langle x^2 \rangle & \cdots & \langle x^{n+1} \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle x^n \rangle & \langle x^{n+1} \rangle & \cdots & \langle x^{2n} \rangle \end{pmatrix}, \;\; \mathbf{w} = \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{pmatrix}, \;\; \mathbf{G} = \begin{pmatrix} \langle y \rangle \\ \langle x\, y \rangle \\ \vdots \\ \langle x^n y \rangle \end{pmatrix}.} \tag{5.22}$$

The matrix $\mathbf{F}$ is symmetric, i.e., $F_{ij} = F_{ji}$, which is useful when computing its inverse.

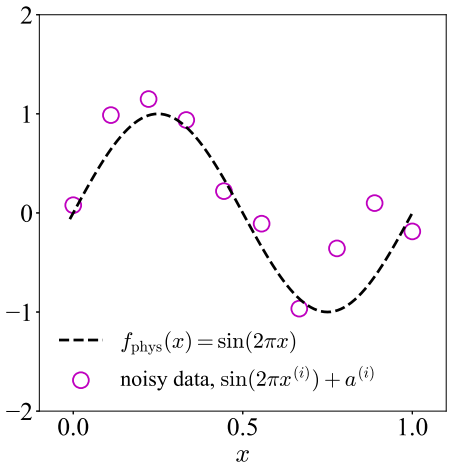

FIG. 5.6: Data preparation for nonlinear learning model, $\ell = 0.8$.

We now construct a concrete example. The physical model is chosen as $f_{\text{phys}}(x) = \sin(2\pi x)$ with $0 \le x \le 1$. We generate $m = 10$ data points uniformly in this interval, $x^{(i)} = i/9, i = 0, 1, 2, \cdots, 9$. The output data are given by $y^{(i)} = \sin(2\pi x^{(i)}) + a^{(i)}$, where $a^{(i)} \sim \text{Unif}[-\ell, \ell]$ represents noise. FIG. 5.6 shows both the physical model (dashed line) and the generated data (points). Solving $\mathbf{Fw} = \mathbf{G}$ yields the optimal parameters $\mathbf{w}^*$ and the corresponding learning model $f_{\mathbf{w}^*}(x)$. Results for different values of n are displayed in FIG. 5.7. Consider first $n = 0$, where $f_{\mathbf{w}}(x) = w_0$. The optimal parameter is simply the average of the outputs, $w_0^* = (y^{(1)} + y^{(2)} + \cdots + y^{(10)})/10$. For $n = 1$, the result reduces to linear regression.

Several important features can be observed. For small n, the learning curve (blue line) fits the data poorly. As n increases, the fit improves. In the extreme case $n = 9$, the model perfectly interpolates the data, i.e., $f_{\mathbf{w}}(x^{(i)}) = y^{(i)}$ for all data points. However, the resulting curve becomes increas-

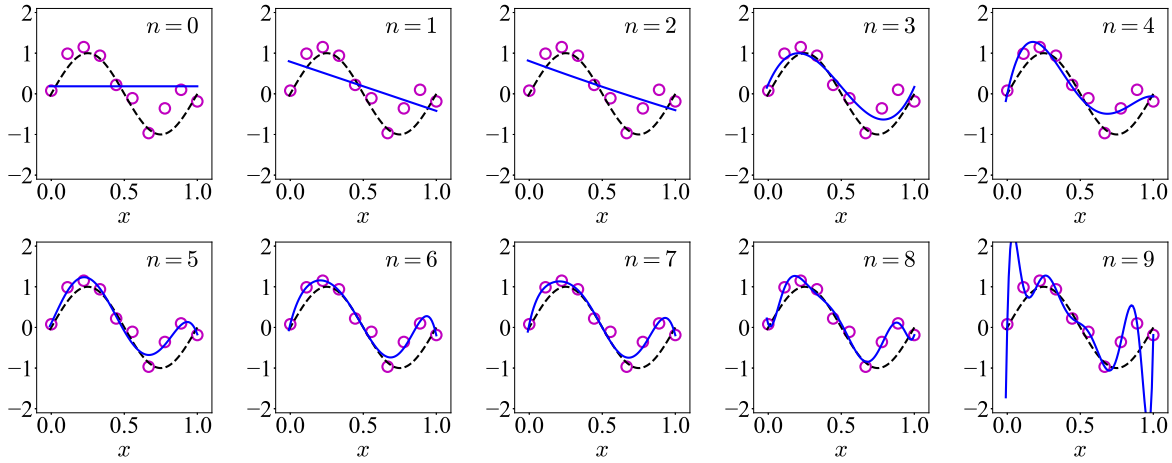

FIG. 5.7: Learning processes with different n.

	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$
w_0^*	0.19	0.79	0.81	0.23	-0.00	0.10	0.08	0.08	0.08	0.08
w_1^*	∅	−1.21	−1.32	8.11	16.93	7.42	13.99	16.69	−9.48	145.11
w_2^*	∅	∅	0.11	−24.74	−69.30	10.00	−71.10	−115.61	412.07	−3158
w_3^*	∅	∅	∅	16.57	88.03	−136.10	212.92	476.91	−3503	28658
w_4^*	∅	∅	∅	∅	−35.73	221.18	−453.31	−1199	13815	−137275
w_5^*	∅	∅	∅	∅	∅	−102.76	497.20	1585	−29673	381953
w_6^*	∅	∅	∅	∅	∅	∅	−199.99	−989.95	35479	−638534
w_7^*	∅	∅	∅	∅	∅	∅	∅	225.70	−22102	631648
w_8^*	∅	∅	∅	∅	∅	∅	∅	∅	5582	−340299
w_9^*	∅	∅	∅	∅	∅	∅	∅	∅	∅	76862

TAB. 5.1: Optimal parameters w_j^*, $j=0\sim n$ in different learning models.

ingly irregular, whereas for small n it remains smooth[∗5-2∗]. The smoothness and the strangeness characterize two important aspects of the learning model: when the learning curve is smoother, we say it has a smaller variance; when the curve is closer to the measurements, we say it has a smaller bias. Thus, when n is small, the learning model has a small variance and a large bias, whereas when n is large, the learning model has a large variance and a small bias. This phenomenon is sometimes referred to as the "no-free-lunch theorem" [∗5-3∗], in the sense that one cannot obtain a learning model with both small variance and small bias simultaneously [∗5-4∗]. This is a very general feature of learning problems in data analysis. The "no-free-lunch theorem" is also commonly referred to as the bias-variance trade-off or the bias-variance decomposition. The learning model with large n is highly complex and possesses strong fitting capability for the given data, but it generally has limited predictive power for new data. Therefore, we often interpret n as the complexity of the model. In our example, $n\approx 3\sim 6$ is a reasonable choice. In TAB. 5.1, the optimal parameters $\mathbf{w}_j^*$ are listed for models with different values of n. An important feature is that, as the model complexity n increases, the magnitudes of w_j^* tend to grow. This can be problematic, because the final prediction relies on the cancellation among large terms $w_j^* x^j$. This phenomenon is referred to as fine-tuning. One of

the most common approaches to mitigate fine-tuning is to introduce a regularization term into the model. We will discuss this in the following paragraphs.

EXERCISE 5-2. Derive the expressions for w_j's in the case of $n=2,3$, write down the explicit form of the matrix $\mathbf{F}^{-1}$.

EXERCISE 5-3. Prove $\mathbf{F}$ could be written as $\vec{\Phi}^\top\vec{\Phi}$ with the element of $\vec{\Phi}$ given as $\phi_{ji}=\phi_j(x^{(i)})$ with $j=0\sim n, i=1\sim m$, see (5.38).

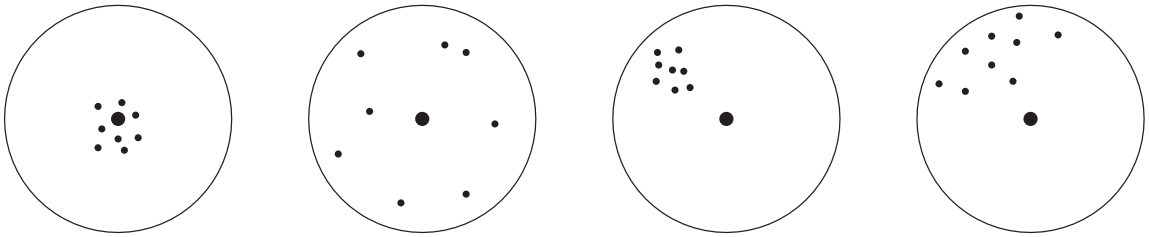

FIG. 5.8: Patterns of bias and variance.

The bias-variance decomposition provides a general understanding of prediction errors. Suppose the true model is $f_{\text{phys}}(x)$ and the observed output is $y=f_{\text{phys}}(x)+a$, where $\mathrm{E}[a]=0$ and $\mathrm{var}[a]$ is fixed. Let the learning model be $\widehat{f}(x)$. For a test input $\overline{x}$, the mean squared error is

$$\boxed{\Delta=\mathrm{E}[(f_{\text{phys}}(\overline{x})+a-\widehat{f}(\overline{x}))^2].} \tag{5.23}$$

By expanding this expression, we obtain

$$\begin{aligned}
\Delta=&\mathrm{E}\Big[\big(f_{\text{phys}}(\overline{x})+a-\widehat{f}(\overline{x})\big)^2\Big]\\
=&\mathrm{E}\Big[f_{\text{phys}}^2(\overline{x})+a^2+\widehat{f}^2(\overline{x})+2a\,f_{\text{phys}}(\overline{x})-2a\,\widehat{f}(\overline{x})-2f_{\text{phys}}(\overline{x})\widehat{f}(\overline{x})\Big]\\
=&\mathrm{E}\Big[f_{\text{phys}}^2(\overline{x})+a^2+\widehat{f}^2(\overline{x})-2f_{\text{phys}}(\overline{x})\widehat{f}(\overline{x})\Big]\\
=&\mathrm{E}\big[a^2\big]+\mathrm{E}\Big[f_{\text{phys}}^2(\overline{x})\Big]+\mathrm{E}\big[\widehat{f}^2(\overline{x})\big]-2\mathrm{E}\big[f_{\text{phys}}(\overline{x})\widehat{f}(\overline{x})\big]\\
=&\mathrm{E}\big[a^2\big]-\mathrm{E}^2[a]+\mathrm{E}\Big[f_{\text{phys}}^2(\overline{x})\Big]+\mathrm{E}\big[\widehat{f}^2(\overline{x})\big]-2\mathrm{E}\big[f_{\text{phys}}(\overline{x})\widehat{f}(\overline{x})\big]+\mathrm{E}^2\big[\widehat{f}(\overline{x})\big]-\mathrm{E}^2\big[\widehat{f}(\overline{x})\big]\\
=&\mathrm{var}[a]+f_{\text{phys}}^2(\overline{x})+\mathrm{E}\big[\widehat{f}^2(\overline{x})\big]-2f_{\text{phys}}(\overline{x})\mathrm{E}\big[\widehat{f}(\overline{x})\big]+\mathrm{E}^2\big[\widehat{f}(\overline{x})\big]-\mathrm{E}^2\big[\widehat{f}(\overline{x})\big]\\
=&\mathrm{var}[a]+\mathrm{E}^2\big[\widehat{f}(\overline{x})\big]-2f_{\text{phys}}(\overline{x})\mathrm{E}\big[\widehat{f}(\overline{x})\big]+f_{\text{phys}}^2(\overline{x})+\mathrm{E}\big[\widehat{f}^2(\overline{x})\big]-\mathrm{E}^2\big[\widehat{f}(\overline{x})\big]\\
=&\mathrm{var}[a]+\big[\mathrm{E}\big[\widehat{f}(\overline{x})\big]-f_{\text{phys}}(\overline{x})\big]^2+\mathrm{E}\Big[\big[\widehat{f}(\overline{x})-\mathrm{E}\big[\widehat{f}(\overline{x})\big]\big]^2\Big],
\end{aligned} \tag{5.24}$$

the meaning of each term is clear:

(a) The noise $\mathrm{var}[a]$ could not be reduced once the physical model is fixed.

(b) $\mathrm{E}[\widehat{f}(\overline{x})]-f_{\text{phys}}(\overline{x})$ is the bias between the learning model $\widehat{f}$ (characterized by its mean $\mathrm{E}[\widehat{f}(\overline{x})]$) and the physical model f_{phys}.

(c) $\mathrm{E}[[\widehat{f}(\overline{x})-\mathrm{E}[\widehat{f}(\overline{x})]]^2]$ is the variance of the learning model $\widehat{f}$ on the testing data.

thus

$$\boxed{\mathrm{E}\Big[\big(f_{\text{phys}}(\overline{x})+a-\widehat{f}(\overline{x})\big)^2\Big]=\mathrm{var}[a]+\big[\text{bias of }\widehat{f}(\overline{x})\big]^2+\text{variance of }\widehat{f}(\overline{x}).} \tag{5.25}$$

EXERCISE 5-4. The w_0^* given in TAB. 5.1 under different n shows similar value. However the value for w_j^* with $j \geq 1$ changes a lot as n increases. Explain the possible reason.

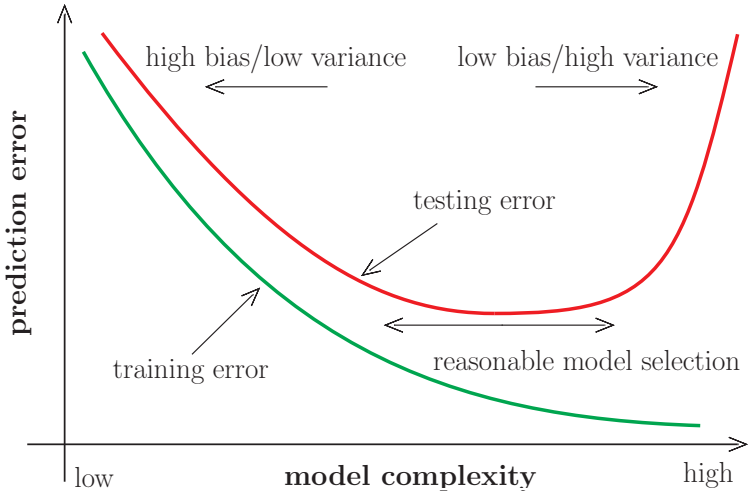

FIG. 5.9: Sketch of the bias-variance decomposition.

The bias-variance decomposition of the learning process provides an explanation for the following observation: if the bias of the learning model is small, then the variance is typically large, and vice versa, since the noise var[a] is generally a constant once the physical model is fixed. A schematic illustration of this decomposition is shown in FIG. 5.9. In fact, there is no prior reason that the total error must be decomposed into variance and bias. A natural question then arises: Can both the bias and the variance be small simultaneously? Questions of this type lie at the core of modern deep learning theory. For example, in certain model studies, a "double descent" behavior of the prediction error has been observed, indicating that the testing error can become very small even in the over-parameterized regime [∗5-5∗]. A deeper understanding of modern neural networks remains an important and exciting open problem. We do not attempt to review the current status here. For instance, some approaches draw inspiration from physics, such as the renormalization group and effective field theory, to analyze neural networks [∗5-6∗].

§5.4 Training Error and Testing Error

In order to characterize the bias-variance decomposition in a more qualitative manner, we define two types of errors, both based on Eq. (5.11). We already have $m = 10$ data points (denoted as the training data), and we define the training error per data sample as

$$e_{\text{train}} \equiv \frac{1}{2m} \sum_{i=1}^{m} \left[f_{\mathbf{w}^*}(x^{(i)}) - y^{(i)} \right]^2, \tag{5.26}$$

where the optimized parameters $\mathbf{w}^*$ are used. For $n = 9$, the training error e_{train} becomes zero, since the learning model can exactly pass through all the training data points.

In addition to the existing training data, one can randomly generate another m' data points (distinct from the training set) according to $f_{\text{phys}}(x^{(i')}) + a^{(i')}$, where both $x^{(i')}$ and $a^{(i')}$ are random vari-

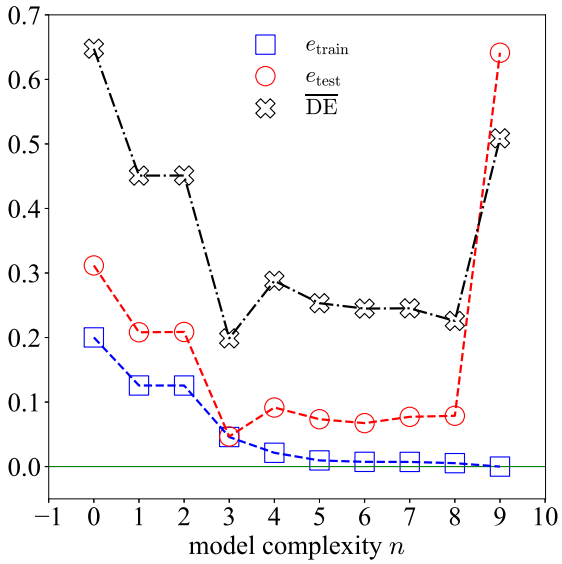

FIG. 5.10: Training error e_{train}, testing error e_{test} and the integration error $\overline{\text{DE}}$.

ables. We then define the testing error per data sample as

$$e_{\text{test}} \equiv \frac{1}{2m'} \sum_{i=1}^{m'} \left[f_{\mathbf{w}^*}(x^{(i')}) - y^{(i')} \right]^2 . \tag{5.27}$$

It is reasonable to expect that the testing error for the case $n = 9$ is larger than that for $n = 3$. Selecting a model with a given n is referred to as model selection. A simple strategy for choosing a reasonable n is to select the learning model with the smallest training error, the smallest testing error, or a total error defined as a weighted sum of the two, namely,

$$e_{\text{total}} = h\, e_{\text{train}} + (1-h) e_{\text{test}}, \tag{5.28}$$

where $0 \le h \le 1$. Taking $h = 1/2$ assigns equal weights to the training and testing errors; otherwise, different weights are assigned.

Another approach to characterize the predictive performance of the learning model is to compute the average (overall) deviation between the physical model $f_{\text{phys}}(x)$ and the learning model $f_{\mathbf{w}}(x)$. In our example, this leads to the following integration error,

$$\overline{\text{DE}} = \int_0^1 \left| f_{\text{phys}}(x) - f_{\mathbf{w}^*}(x) \right| \mathrm{d}x. \tag{5.29}$$

Although $\overline{\text{DE}}$ correctly measures the deviation of $f_{\mathbf{w}^*}(x)$ from the physical model, it is of limited practical use since the exact form of $f_{\text{phys}}(x)$ is generally unknown. In FIG. 5.10, the three types of errors are plotted as functions of the model complexity, where $m' = 5$ is used. For the present problem, we find that $n \approx 3 \sim 6$ provides a reasonable choice.

EXERCISE 5-5. Define the generalized integration error as $\overline{\text{DE}}(\nu) = \int |f_{\text{phys}}(x) - f_{\mathbf{w}^*}(x)|^\nu \mathrm{d}x$, and investigate

its dependence on n for the polynomial model with $\nu = 2, 3$.

§5.5 Regularization/Penalty Term

As discussed above and shown in FIG. 5.10, if the model complexity n is chosen appropriately, e.g., $n = 3$ or $n = 4$, the learning curve exhibits both low training and testing errors. On the other hand, if n is very small, e.g., $n = 0$, the learning model $f_{\mathbf{w}*}(x) = w_0^*$ yields both large training and testing errors. If n is too large, such as $n = 9$, the learning model achieves zero training error. However, the testing error becomes extremely large, indicating that the model has very poor predictive power, even though it fits the training data perfectly. In addition, in the small-n regime (e.g., $n = 0$ or $n = 1$), the learning model fails to capture the structure of the training data, a situation referred to as under-fitting. In contrast, in the large-n regime (e.g., $n = 9$), the learning model fits the training data extremely well but performs poorly on new samples; this situation is known as over-fitting. Both under-fitting and over-fitting are undesirable. The overall performance of the training and testing errors is characterized by the prediction error.

We now introduce the regularization method to address the over-fitting problem. In addition to the original loss function $2^{-1}\sum_{i=1}^{m}[f_{\mathbf{w}}(x^{(i)}) - y^{(i)}]^2$, we include an extra term $\lambda g(\mathbf{w})$, namely,

$$\boxed{J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m}\left[f_{\mathbf{w}}(x^{(i)}) - y^{(i)}\right]^2 + \lambda g(\mathbf{w}),} \tag{5.30}$$

where $\lambda > 0$ is a parameter introduced by hand, and $g(\mathbf{w})$ is a positive function of the learning parameters $\mathbf{w}$. We refer to $\lambda g(\mathbf{w})$ as the regularization term in the loss function $J(\mathbf{w})$. There exist many different forms of regularization terms, each with its own practical interpretation. Here we adopt the following form:

$$\boxed{\lambda g(\mathbf{w}) = \frac{1}{2}\lambda \mathbf{w}^\top \mathbf{w} = \frac{1}{2}\lambda\left(w_0^2 + w_1^2 + \cdots + w_n^2\right),} \tag{5.31}$$

which is known as the ℓ-2 regularization term [∗5-7∗]. Another important choice is the ℓ-1 regularization, which is often used to promote sparsity in the learning model.

The equations determining the parameters $w_0, w_1, \cdots, w_n$ in the presence of the regularization term can be derived in a similar manner:

$$\begin{pmatrix} \langle 1\rangle & \langle x\rangle & \cdots & \langle x^n\rangle \\ \langle x\rangle & \langle x^2\rangle & \cdots & \langle x^{n+1}\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle x^n\rangle & \langle x^{n+1}\rangle & \cdots & \langle x^{2n}\rangle \end{pmatrix}\begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} \langle y\rangle \\ \langle xy\rangle \\ \vdots \\ \langle x^n y\rangle \end{pmatrix} - \lambda\begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{pmatrix}, \tag{5.32}$$

or equivalently,

$$\left(\mathbf{F} + \lambda\vec{\mathbf{I}}\right)\mathbf{w} = \mathbf{G}, \tag{5.33}$$

where $\vec{\mathbf{I}}$ is the $(n+1)\times(n+1)$ identity matrix. In the left panel of FIG. 5.11, the effect of the regularization term with $\lambda = 10^{-12}$ is illustrated for the 9th-order polynomial learning model. Even such a tiny value of λ significantly regularizes the learning curve, making its overall behavior smoother. As λ increases, the curve becomes increasingly smooth. It should be emphasized once again that the regularization term $\lambda g(\mathbf{w})$ is not encoded in the data itself; rather, it is introduced by hand. In this

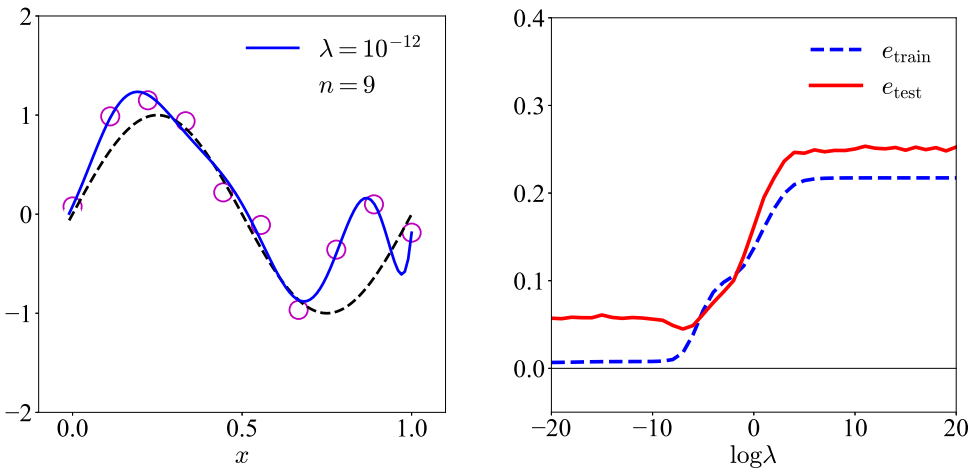

FIG. 5.11: Left: effects of the regularization term with different strengths; right: training/testing error as a function of λ, here $n = 9$.

sense, the parameter λ reflects one's belief in the data: a small λ implies that the data are trusted more, whereas a large λ indicates less confidence in the data.

A natural question concerns the behavior in the large-λ limit. For example, when $\lambda = 1$, the optimal parameters w_j^* in the $n = 9$ model are found to be 0.44, −0.28, −0.31, −0.20,−0.10, −0.02, 0.03, 0.06, 0.08 and 0.09, respectively. The limit $\lambda = \infty$ corresponds to the situation where the original data are completely disregarded, since

$$\lambda g(\mathbf{w}) \gg \frac{1}{2}\sum_{i=1}^{m}\left[f_{\mathbf{w}}(x^{(i)}) - y^{(i)}\right]^2, \tag{5.34}$$

and the learning model is approximately reduced to $f_{\mathbf{w}}(x) = \lambda g(\mathbf{w})$. If $g(\mathbf{w}) = \mathbf{w}^{\top}\mathbf{w}$, the optimal solution is simply $\mathbf{w}^* \approx \vec{0}$.

In the right panel of FIG. 5.11, the training and testing losses for the model with $n = 9$ are plotted as functions of $\ln\lambda$, where the number of testing samples is $m' = 10^4$. As $\ln\lambda \to -\infty$ (i.e., in the absence of regularization), the training loss approaches zero, since the model perfectly fits the data for $n = 9$, while the testing loss remains finite. In the opposite limit $\ln\lambda \to \infty$, the learning model tends to zero, and both the training and testing losses approach constants determined by the data.

EXERCISE 5-6. Change the noise $a^{(i)}$ to be sampled from the uniform distribution Unif$[-\ell, \ell]$ with ℓ a large number (e.g., 10) in the polynomial learning model, investigate the learning properties. In this case the noise $a^{(i)}$ exceeds the physical model $\sin(2\pi x^{(i)})$ and consequently the measurements $y^{(i)}$ are almost characterized by the noise. It tells us that noise has no regular behavior, and thus can not be studied via learning algorithms.

EXERCISE 5-7. The ℓ-1 norm is defined as $|\mathbf{w}|_1 = \sum_{j=0}^{n} |w_j|$. Explore its geometrical meaning.

EXERCISE 5-8. Write down the equation for $\mathbf{w}^*$ if the regularization term is taken as $\lambda(\mathbf{w}^2)^2$, where $(\mathbf{w}^2)^2 = (w_0^2 + w_1^2 + \cdots + w_n^2)^2$. What's about $\mathbf{w}^4 = w_0^4 + w_1^4 + \cdots + w_n^4$? Moreover, what's about if the regularization term is $\lambda\mathbf{w}^{\top}\mathbf{T}\mathbf{w}$ with $\mathbf{T}$ a positive-definite matrix? Investigate the relevant learning properties.

§5.6 Normal Equation, Weighted Least Squares

Let us discuss the polynomial learning model in more detail. In our polynomial learning model, $f_{\mathbf{w}}(x)=w_0+w_1x+w_2x^2+\cdots+w_nx^n$, which can be rewritten in the form

$$\boxed{f_{\mathbf{w}}(x)=\mathbf{w}^{\top}\vec{\phi}(x)=\vec{\phi}^{\top}(x)\mathbf{w},\ \ \mathbf{w}=(w_0,w_1,w_2,\cdots,w_n)^{\top},\ \ \vec{\phi}(x)=\left(1,x,x^2,\cdots,x^n\right)^{\top},} \tag{5.35}$$

i.e., $\mathbf{w}\in \mathrm{R}^{(n+1)\times 1}\equiv \mathrm{R}^{n+1}$ is a column vector (a column vector is thin and tall) and its transpose $\mathbf{w}^{\top}\in \mathrm{R}^{1\times(n+1)}$ is a row vector (a row vector is fat and short). The jth component of the vector $\vec{\phi}$ is x^j with $j=0\sim n$. Denoting $\phi_j(x)=x^j$, then

$$\boxed{\vec{\phi}(x)=\left(\phi_0(x),\phi_1(x),\cdots,\phi_n(x)\right)^{\top},\ \ \phi_j(x)=x^j,\ \ j=0\sim n.} \tag{5.36}$$

In the polynomial model, each component $\phi_j(x)$ takes the form x^j. More generally, however, the components $\phi_j(x)$ are not restricted to this form; for instance, one may choose $\phi_j(x)=\sin(j\,t\,x)$ with t being a constant. We refer to the functions $\phi_j(x)$ as basis functions.

Adopting the basis function representation, the loss function without a regularization term is given by

$$\boxed{J(\mathbf{w})=\frac{1}{2}\sum_{i=1}^{m}\left[f_{\mathbf{w}}(x^{(i)})-y^{(i)}\right]^2=\frac{1}{2}\left(\vec{\Phi}\mathbf{w}-\mathbf{y}\right)^{\top}\left(\vec{\Phi}\mathbf{w}-\mathbf{y}\right),\ \ f_{\mathbf{w}}(x^{(i)})=\mathbf{w}^{\top}\vec{\phi}(x^{(i)}),} \tag{5.37}$$

where

$$\boxed{\vec{\Phi}=\begin{pmatrix}\phi_0(x^{(1)}) & \phi_1(x^{(1)}) & \cdots & \phi_n(x^{(1)})\\ \phi_0(x^{(2)}) & \phi_1(x^{(2)}) & \cdots & \phi_n(x^{(2)})\\ \vdots & \vdots & \ddots & \vdots\\ \phi_0(x^{(m)}) & \phi_1(x^{(m)}) & \cdots & \phi_n(x^{(m)})\end{pmatrix}=\begin{pmatrix}1 & \phi_1(x^{(1)}) & \cdots & \phi_n(x^{(1)})\\ 1 & \phi_1(x^{(2)}) & \cdots & \phi_n(x^{(2)})\\ \vdots & \vdots & \ddots & \vdots\\ 1 & \phi_1(x^{(m)}) & \cdots & \phi_n(x^{(m)})\end{pmatrix}\in \mathrm{R}^{m\times(n+1)},} \tag{5.38}$$

and

$$\boxed{\mathbf{y}=\left(y^{(1)},y^{(2)},\cdots,y^{(m)}\right)^{\top}\in \mathrm{R}^m.} \tag{5.39}$$

Specifically, the derivation proceeds as follows,

$$\begin{aligned}\sum_{i=1}^{m}\left[f_{\mathbf{w}}(x^{(i)})-y^{(i)}\right]^2&=\left[f_{\mathbf{w}}(x^{(1)})-y^{(1)}\right]^2+\cdots+\left[f_{\mathbf{w}}(x^{(m)})-y^{(m)}\right]^2\\ &=\begin{pmatrix}\underbrace{w_0\phi_0(x^{(1)})+\cdots+w_n\phi_n(x^{(1)})}f\mathbf{w}(x^{(1)})-y^{(1)}\\ \vdots\\ \underbrace{w_0\phi_0(x^{(m)})+\cdots+w_n\phi_n(x^{(m)})}f\mathbf{w}(x^{(m)})-y^{(m)}\end{pmatrix}^{\top}\\ &\quad\times\begin{pmatrix}w_0\phi_0(x^{(1)})+\cdots+w_n\phi_n(x^{(1)})-y^{(1)}\\ \vdots\\ w_0\phi_0(x^{(m)})+\cdots+w_n\phi_n(x^{(m)})-y^{(m)}\end{pmatrix}.\end{aligned} \tag{5.40}$$

We refer to the matrix $\vec{\Phi}$ as the design matrix, whose entries are given by $\phi_j(x^{(i)})$. The derivative of the loss $J(\mathbf{w})$ with respect to $\mathbf{w}$ is given by $\partial J(\mathbf{w})/\partial \mathbf{w} = \vec{\Phi}^\top \vec{\Phi}\mathbf{w} - \vec{\Phi}^\top \mathbf{y}$. Setting $\partial J(\mathbf{w})/\partial \mathbf{w} = 0$, namely

$$\boxed{\vec{\Phi}^\top \vec{\Phi}\mathbf{w} = \vec{\Phi}^\top \mathbf{y},} \tag{5.41}$$

one obtains the solution of this normal equation,

$$\boxed{\mathbf{w}^* = \left(\vec{\Phi}^\top \vec{\Phi}\right)^{-1} \vec{\Phi}^\top \mathbf{y}.} \tag{5.42}$$

After including the regularization term $2^{-1}\lambda \mathbf{w}^\top \mathbf{w}$ into the loss function $J(\mathbf{w})$, the optimal solution becomes

$$\boxed{\mathbf{w}^* = \left(\vec{\Phi}^\top \vec{\Phi} + \lambda \vec{1}\right)^{-1} \vec{\Phi}^\top \mathbf{y}.} \tag{5.43}$$

It should be pointed out that the regularization term introduced into the least-squares formulation plays an important role when the matrix $\vec{\Phi}^\top \vec{\Phi}$ is singular, i.e., $\det(\vec{\Phi}^\top \vec{\Phi}) = 0$. Choosing a large λ implies that the influence of the original data is relatively down-weighted. Let us further analyze the normal equation. Assume that we seek the minimum of the quadratic objective function $K(\mathbf{w}) = 2^{-1}\mathbf{w}^\top \vec{\Phi}\mathbf{w} - \mathbf{y}^\top \mathbf{w}$. The gradient of $K(\mathbf{w})$ is given by $\vec{\Phi}\mathbf{w} - \mathbf{y}$. To find the optimal $\mathbf{w}^*$, one needs to solve $\vec{\Phi}\mathbf{w} = \mathbf{y}$. However, in many practical situations, for example, when the number of equations exceeds the number of unknowns (i.e., the number of rows of $\vec{\Phi}$ is larger than the number of columns), this system may have no exact solution. In such cases, the system is over-determined. This is often related to the situation where $\det(\vec{\Phi}^\top \vec{\Phi}) = 0$ discussed above. Therefore, instead of solving $\vec{\Phi}\mathbf{w} = \mathbf{y}$ directly, one typically reformulates the problem as a least-squares optimization problem, $\min_{\mathbf{w}} (\vec{\Phi}\mathbf{w} - \mathbf{y})^\top (\vec{\Phi}\mathbf{w} - \mathbf{y})$. This objective function coincides with $J(\mathbf{w})$, and its solution is obtained from the normal equation.

EXERCISE 5-9. Prove the following identities which are useful in deriving the normal equation

$$\frac{\partial \mathbf{a}^\top \mathbf{b}}{\partial \mathbf{a}} = \mathbf{b}, \quad \frac{\partial \mathbf{a}^\top \mathbf{M}\mathbf{a}}{\partial \mathbf{a}} = \mathbf{a}^\top \left(\mathbf{M} + \mathbf{M}^\top\right), \quad \mathbf{a}, \mathbf{b} \in \mathrm{R}^{d\times 1} \equiv \mathrm{R}^d, \quad \mathbf{M} \in \mathrm{R}^{d\times d}. \tag{5.44}$$

Since $\phi_0 = 1$, without loss of generality the design matrix can also be written as

$$\begin{pmatrix} \phi_1(x^{(1)}) & \phi_2(x^{(1)}) & \cdots & \phi_n(x^{(1)}) \\ \phi_1(x^{(2)}) & \phi_2(x^{(2)}) & \cdots & \phi_n(x^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(x^{(m)}) & \phi_2(x^{(m)}) & \cdots & \phi_n(x^{(m)}) \end{pmatrix}, \tag{5.45}$$

where each column forms a vector $\vec{\varphi}_j = (\phi_j(x^{(1)}), \cdots, \phi_j(x^{(m)}))^\top \in \mathrm{R}^m$ with $j = 1 \sim n$ (the distinction between n and $n+1$ is not essential for the following discussion). Each row $\vec{\phi}(x^{(i)})$ is an n-dimensional vector. Under the assumption that the model complexity n is smaller than the number of data points m, the vectors $\vec{\phi}(x^{(i)})$ span a subspace S of dimension n. Denote $\mathbf{t}$ as an m-dimensional vector whose ith component is given by $f_{\mathbf{w}}(x^{(i)})$, i.e., $\mathbf{t} = (f_{\mathbf{w}}(x^{(1)}), \cdots, f_{\mathbf{w}}(x^{(m)}))^\top$. Since $\mathbf{t}$ is a linear combination of the basis vectors $\vec{\varphi}_j$, it lies in the n-dimensional subspace. Under this perspective, the loss function $J(\mathbf{w}) \sim \sum_{i=1}^m [y^{(i)} - f_{\mathbf{w}}(x^{(i)})]^2 = (y^{(1)} - t_1)^2 + \cdots + (y^{(m)} - t_m)^2$ can be interpreted as the squared Euclidean distance between $\mathbf{y}$ and $\mathbf{t}$. Therefore, solving the least-squares problem amounts to finding a vector $\mathbf{t}$ within the subspace S that minimizes the distance to $\mathbf{y}$, i.e., projecting $\mathbf{y}$ onto S.

We now examine the meaning of the bias parameter w_0. By computing the derivative of $J(\mathbf{w})$ with respect to w_0, starting from

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m}\left[y^{(i)} - w_0 - \sum_{j=1}^{n} w_j \phi_j(\mathbf{x}^{(i)})\right]^2, \tag{5.46}$$

one obtains the optimal value

$$\boxed{w_0^* = \langle y\rangle - \sum_{j=1}^{n} w_j \langle \phi_j\rangle,\ \ \langle y\rangle = \frac{1}{m}\sum_{i=1}^{m} y^{(i)},\ \ \langle \phi_j\rangle = \frac{1}{m}\sum_{i=1}^{m}\phi_j(\mathbf{x}^{(i)}).} \tag{5.47}$$

This result indicates that the bias parameter w_0 compensates for the difference between the target output and the weighted average of the basis functions evaluated on the data samples.

For cases where the data samples $\mathbf{x}^{(i)}$ carry different levels of importance, for example, when some samples are considered more reliable than others, we can introduce a weighted loss function,

$$\boxed{J_{\vec{\Theta}}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m}\theta^{(i)}\left[f_{\mathbf{w}}(x^{(i)}) - y^{(i)}\right]^2 + \frac{1}{2}\lambda \mathbf{w}^\top \mathbf{w},} \tag{5.48}$$

which leads to the modified normal equation as well as the solution:

$$\boxed{\left(\vec{\Phi}^\top \vec{\Theta}\vec{\Phi} + \lambda \vec{\mathrm{I}}\right)\mathbf{w} = \vec{\Phi}\vec{\Theta}\mathbf{y} \to \mathbf{w} = \left(\vec{\Phi}^\top \vec{\Theta}\vec{\Phi} + \lambda \vec{\mathrm{I}}\right)^{-1}\vec{\Phi}\vec{\Theta}\mathbf{y},} \tag{5.49}$$

where $\vec{\Theta} = \mathrm{diag}(\theta^{(1)}, \cdots, \theta^{(m)})$, with $\theta^{(i)} > 0$ and $\sum_{i=1}^{m}\theta^{(i)} = 1$.

EXERCISE 5-10. The sigmoid function is often defined as $\sigma(x) = 1/(1+e^{-x})$. Prove that $\tanh(x) = 2\sigma(2x) - 1$. For two learning models,

$$f_{\mathbf{w}}(x) = w_0 + \sum_{j=1}^{n} w_j \sigma\left(\frac{x - v_j}{s}\right),\ \ f_{\mathbf{u}}(x) = u_0 + \sum_{j=1}^{n} u_j \tanh\left(\frac{x - v_j}{2s}\right), \tag{5.50}$$

determine the relationship between the two sets of parameters w_j and u_j.

§5.7 Stochastic Gradient Descent (Conceptual)

It is worth noting that the dimension of the matrix $\vec{\Phi}^\top\vec{\Phi}$ is $(n+1)\times(n+1)$, i.e., $\dim \vec{\Phi}^\top\vec{\Phi} = n+1$, which is often much smaller than the number of data points m, $n+1 \ll m$, making the normal equation solvable in practice. However, when the model complexity n becomes large, directly inverting $\vec{\Phi}^\top\vec{\Phi}$ may become computationally prohibitive. In such cases, one typically relies on iterative optimization methods such as gradient descent to find the optimal parameters. In these situations, the parameters are updated using the gradient of $J(\mathbf{w})$, namely, $\mathbf{w} \leftarrow \mathbf{w} - \epsilon(\vec{\Phi}^\top\vec{\Phi}\mathbf{w} - \vec{\Phi}^\top\mathbf{y})$, where ϵ denotes the learning rate (or step size). Since the learning problems discussed in this lecture are based on polynomial models, closed-form solutions exist via the normal equation. In more general cases where the basis functions take other forms, closed-form solutions may not exist for the parameters. In such situations, optimization algorithms must be employed.

The optimization procedure typically consists of the following steps: (a) initialize the parameter $\mathbf{w}$; (b) randomly select a data sample $(x^{(i)}, y^{(i)})$; (c) update the parameter according to $\mathbf{w} \leftarrow \mathbf{w} - \epsilon_i \nabla J^{(i)}(\mathbf{w})$, where $\nabla J^{(i)}(\mathbf{w})$ is the gradient associated with the selected sample, given by $\nabla J^{(i)}(\mathbf{w}) = -\vec{\phi}(x^{(i)})(y^{(i)} - f_{\mathbf{w}}(x^{(i)}))$; and (d) repeat the procedure until a termination criterion is satisfied. Here, data samples are selected stochastically to reduce the computational cost of evaluating the gradient at each iteration. This procedure is known as stochastic gradient descent (SGD), which plays a central role in modern large-scale optimization.

EXERCISE 5-11. Use SGD instead of the normal equation to solve the nonlinear curve fitting problem.

EXERCISE 5-12. Denote the set for SGD contain m' samples, discuss the relation between the random error produced by the SGD algorithm and the ratio m'/m.

EXERCISE 5-13. If on the other hand, $n+1 \sim n \gg m$, i.e., the number of model parameters in the model is larger than the sample size, are there new features of such problems?

§5.8 Problems for This Lecture

Theoretical

1. In the linear learning model, define the distance from the point $(x^{(i)}, y^{(i)})$ to the fitting line $y = ax + b$ as

$$\boxed{d_{\perp}^{(i)}(a,b) = \left|a x^{(i)} + b - y^{(i)}\right| \Big/ \sqrt{a^2+1}\,, \quad d_{\mathrm{x}}^{(i)}(a,b) = \left|a x^{(i)} + b - y^{(i)}\right| \Big/ a.} \tag{5.51}$$

   Consequently, the loss functions can be defined as $J_{\perp}(a,b) = 2^{-1}\sum_{i=1}^{m} d_{\perp}^{(i),2}(a,b)$ or $J_{\mathrm{x}}(a,b) = 2^{-1}\sum_{i=1}^{m} d_{\mathrm{x}}^{(i),2}(a,b)$. Derive the equations for determining the parameters a and b. See FIG. 5.12 for the geometric interpretation of $d_{\perp}$ and d_{x}. In the regime where a is large, one has $d_{\perp} \approx d_{\mathrm{x}}$. Are the estimators biased or not?

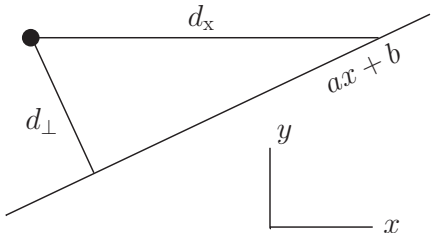

FIG. 5.12: Sketch of the distances $d_{\perp}$ and d_{x}.

2. Prove the relation $\partial \mathbf{a}^{\top}\mathbf{X}^{\top}\mathbf{b}/\partial \mathbf{X} = \mathbf{b}\mathbf{a}^{\top}$, and

$$\frac{\partial \det \mathbf{Y}}{\partial x} = \det \mathbf{Y} \mathrm{tr}\left[\mathbf{Y}^{-1}\frac{\partial \mathbf{Y}}{\partial x}\right], \quad \frac{\partial \det \mathbf{Y}}{\partial \mathbf{Y}} = \det \mathbf{Y}\mathbf{Y}^{-\top}, \quad \frac{\partial \ln \det \mathbf{Y}}{\partial \mathbf{Y}} = \mathbf{Y}^{-\top}, \tag{5.52}$$

   where capital letters denote matrices.
3. If $\mathbf{AB} = \mathbf{BA}$, show $\mathbf{A}^2 - \mathbf{B}^2 = (\mathbf{A}+\mathbf{B})(\mathbf{A}-\mathbf{B})$ and $\mathbf{A}^2 + \mathbf{B}^2 = \mathbf{A}^2 + 2\mathbf{AB} + \mathbf{B}^2$.
4. Let $\mathbf{A}$ be a $d \times d$ symmetric matrix, $\mathbf{C}$ a $k \times k$ symmetric matrix, and $\mathbf{B}$ a $k \times d$ matrix. Show that the following identity, known as the Sherman–Morrison–Woodbury (SMW) relation, holds:

$$\left(\mathbf{A}^{-1} + \mathbf{B}^{\top}\mathbf{C}^{-1}\mathbf{B}\right)^{-1} = \mathbf{A} - \mathbf{A}\mathbf{B}^{\top}\left(\mathbf{B}\mathbf{A}\mathbf{B}^{\top} + \mathbf{C}\right)^{-1}\mathbf{B}\mathbf{A}. \tag{5.53}$$

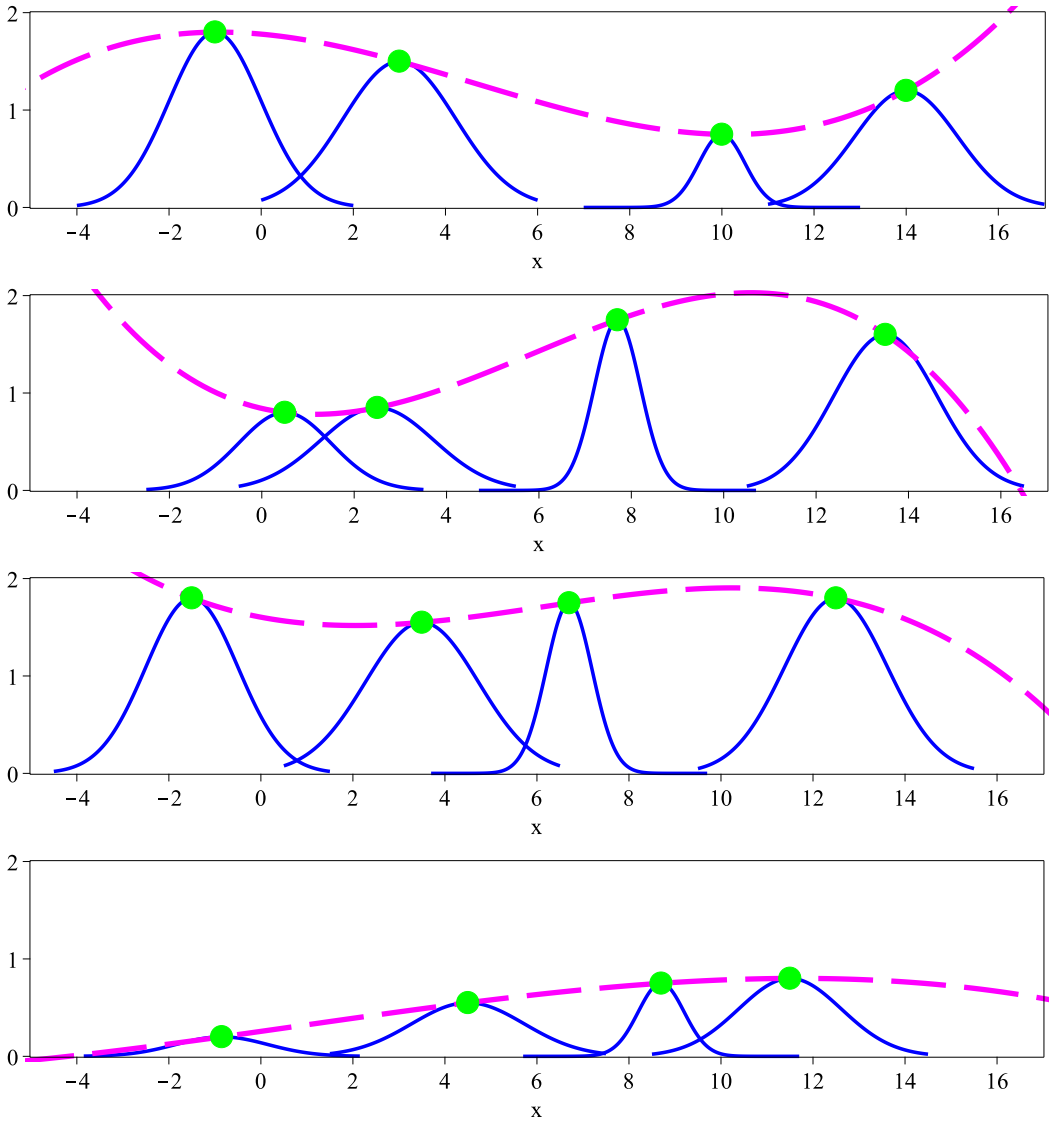

FIG. 5.13: Sketch of the Gauss kernel with different ν for each data sample.

5. In the example $y = \sin(2\pi x)$, the matrix constructed from eight data points is given by

$$\mathbf{F} = \begin{pmatrix} 8.00 & 4.00 & 2.86 & 2.29 & 1.95 & 1.73 & 1.57 & 1.46 \\ 4.00 & 2.86 & 2.29 & 1.95 & 1.73 & 1.57 & 1.46 & 1.37 \\ 2.86 & 2.29 & 1.95 & 1.73 & 1.57 & 1.46 & 1.37 & 1.31 \\ 2.29 & 1.95 & 1.73 & 1.57 & 1.46 & 1.37 & 1.31 & 1.25 \\ 1.95 & 1.73 & 1.57 & 1.46 & 1.37 & 1.31 & 1.25 & 1.21 \\ 1.73 & 1.57 & 1.46 & 1.37 & 1.31 & 1.25 & 1.21 & 1.18 \\ 1.57 & 1.46 & 1.37 & 1.31 & 1.25 & 1.21 & 1.18 & 1.15 \\ 1.46 & 1.37 & 1.31 & 1.25 & 1.21 & 1.18 & 1.15 & 1.12 \end{pmatrix}. \tag{5.54}$$

What properties does this matrix have? In the general case (i.e., for an arbitrary number of data

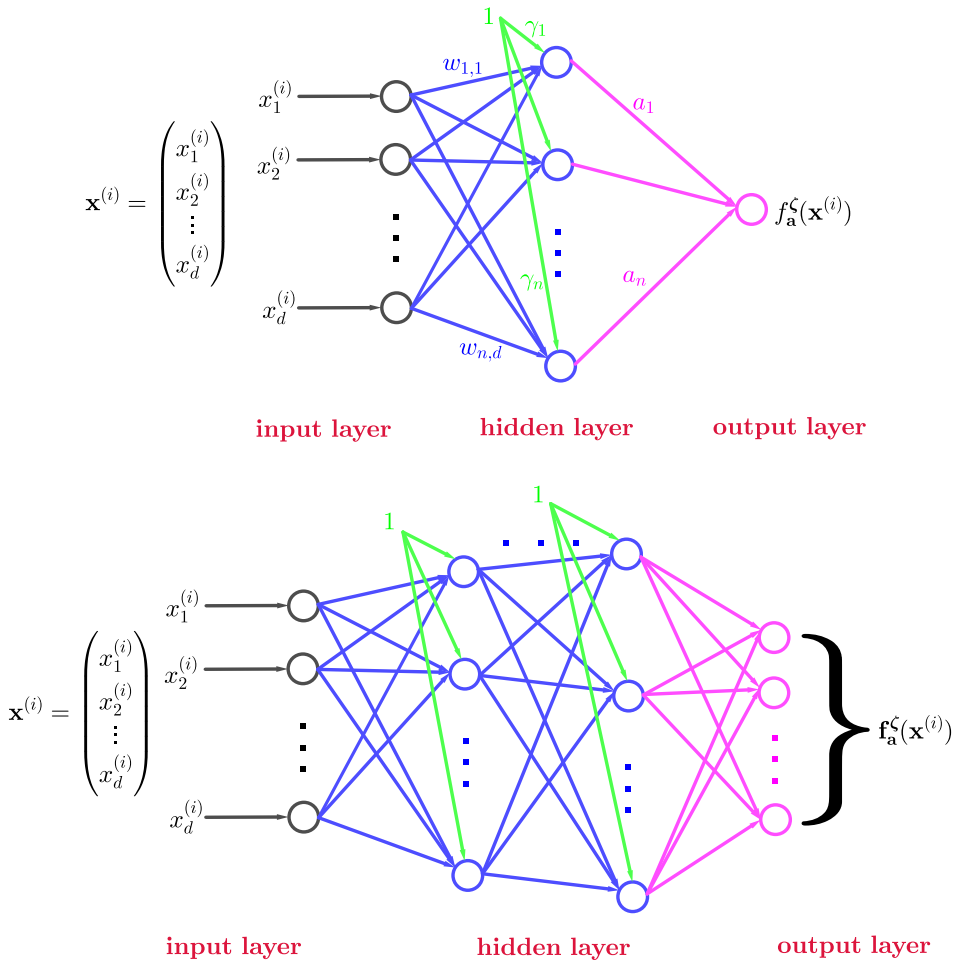

FIG. 5.14: Neural network with three and multiple layers (upper and lower).

points), analyze and discuss the properties of $\mathbf{F}$.

Computational/Programming

6. Apply the gradient descent algorithm to determine the optimal values of the learning parameters for the loss function (5.51), and compare the results with FIG. 5.2. Discuss whether the predictions for a and b are biased or unbiased.
7. Consider the following learning model,

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m}\left[\sum_{j=1}^{m} w_j G(x^{(i)}, x^{(j)}) - y^{(i)}\right]^2 + \frac{\lambda}{2}\mathbf{w}^{\top}\mathbf{w}, \quad G(x, x^{(j)}) = \exp\left[\frac{-(x - x^{(j)})^2}{2\nu^2}\right], \tag{5.55}$$

where no w_0 term is included. The function G is referred to as the Gaussian kernel. See FIG. 5.13 for a schematic illustration of the Gaussian kernel method, where each data sample may have a different ν. The data are generated from the model $y = \sin(\pi x)/\pi x + \alpha x + \beta\delta$, where $\alpha = 0.1$ is fixed and $\delta \sim \text{Unif}[0, 1]$, and β controls the noise strength. Investigate the learning process using

SGD under different choices of ϵ, ν, and λ, with $m = 50$ fixed. Discuss the limiting behavior as $\nu \to 0$. Write down the design matrix $\vec{\Phi}$ for the Gaussian kernel learning model.

8. In the simplest neural network setting, the learning model is given by $f_{\mathbf{a}}^{\vec{\zeta}}(\mathbf{x}) = \sum_{j=1}^{n} a_j \phi(\mathbf{x}, \vec{\zeta}_j)$, which is linear in a_j (analogous to w_j in the main text) and nonlinear in the parameters $\vec{\zeta}_j$. Here the sigmoid basis function is adopted: $\phi(\mathbf{x}, \vec{\zeta}) = [1+\exp(-\gamma - \mathbf{x}^{\top}\mathbf{w})]^{-1}$, where $\vec{\zeta}$ consists of $\mathbf{w}$ and γ. The upper panel of FIG. 5.14 illustrates a three-layer neural network, where $\mathbf{x}^{(i)} \in \mathrm{R}^d$, and each $w_{j,k}$ corresponds to the kth component of the input vector $\mathbf{x}^{(i)}$. Only one hidden layer is considered. The indices are defined as follows: $i = 1 \sim m$ labels data samples, $j = 1 \sim n$ labels the hidden units (parameters $\mathbf{w}_j, \gamma_j$ and coefficients a_j), and $k = 1 \sim d$ labels the input dimension. The lower panel of FIG. 5.14 shows a general multilayer neural network. Using the standard ℓ-2 loss function,

$$J(\mathbf{w}, a, \gamma) = \frac{1}{2}\sum_{i=1}^{m}\Big[y^{(i)} - f_{\mathbf{a}}^{\vec{\zeta}}(\mathbf{x}^{(i)})\Big]^2 = \frac{1}{2}\sum_{i=1}^{m}\Big[y^{(i)} - f_{\mathbf{a}}^{\mathbf{w},\gamma}(\mathbf{x}^{(i)})\Big]^2, \tag{5.56}$$

and defining the error $e^{(i)}$ for the ith sample as $e^{(i)} = y^{(i)} - f_{\mathbf{a}}^{\mathbf{w},\gamma}(\mathbf{x}^{(i)})$, as well as the basis function $b_j^{(i)} = \phi(\mathbf{x}^{(i)}, \mathbf{w}_j, \gamma_j)$, the augmented input vector $\widetilde{\mathbf{x}}^{(i)} = (\mathbf{x}^{(i),\top}, 1)^{\top}$, and the augmented parameter vector $\widetilde{\mathbf{w}}_j = (\mathbf{w}_j^{\top}, \gamma_j)^{\top}$, prove the following relations,

$$\frac{\partial J^{(i)}}{\partial a_j} = -\phi(\mathbf{x}^{(i)}, \mathbf{w}_j, \gamma_j)\Big[y^{(i)} - f_{\mathbf{a}}^{\mathbf{w},\gamma}(\mathbf{x}^{(i)})\Big], \quad \frac{\partial J^{(i)}}{\partial \widetilde{w}_{j,k}} = -a_j b_j^{(i)}\Big[1 - b_j^{(i)}\Big]\widetilde{x}_k^{(i)} e^{(i)}, \tag{5.57}$$

where $k = 1 \sim d+1$ in the second expression. Implement the learning procedure using SGD based on the dataset constructed in the previous problem[∗5-8∗].

References and Notes for Lecture 5

[∗5-1∗] For example, consider two measurements $(1,2)$ and $(2,5)$, i.e., $x^{(1)} = 1$, $y^{(1)} = 2$ and $x^{(2)} = 2$, $y^{(2)} = 5$, with $m = 2$. Then

$$\langle x\rangle = \frac{1}{2}\big(x^{(1)} + x^{(2)}\big) = 3/2, \quad \langle y\rangle = \frac{1}{2}\big(y^{(1)} + y^{(2)}\big) = 7/2,$$
$$\langle x^2\rangle = \frac{1}{2}\big(x^{(1),2} + x^{(2),2}\big) = 5/2, \quad \langle y^2\rangle = \frac{1}{2}\big(y^{(1),2} + y^{(2),2}\big) = 29/2, \quad \langle xy\rangle = \frac{1}{2}\big(x^{(1)}y^{(1)} + x^{(2)}y^{(2)}\big) = 6.$$

[∗5-2∗] See the discussion in, C. Bishop, *Pattern Recognition and Machine Learning*, 2006, Springer, Chap. 1.

[∗5-3∗] D. Wolpert and W. Macready, *No Free Lunch Theorems for Optimization*, IEEE Trans. Evol. Comp. **1**, 67 (1997).

[∗5-4∗] An example of fine-tuning is the quantity

$$u = \sqrt{x+1} - \sqrt{x} = \frac{1}{\sqrt{x+1} + \sqrt{x}},$$

where, for instance, x is a very large number. The first expression for calculating u is numerically unstable when x is large, as rounding errors may arise. In contrast, the second expression is stable and does not suffer from such numerical issues.

[∗5-5∗] See, e.g., M. Belkin *et al.*, *Reconciling Modern Machine Learning Practice and the Classical Bias-variance Trade-off*, PNAS, **116**, 15849 (2019).

[∗5-6∗] See, e.g., H.W. Lin, M. Tegmark, and D. Rolnick, *Why Does Deep and Cheap Learning Work So Well?*, J. Stat. Phys. **168**, 1223 (2017), where ideas from effective field theory are applied; C. Beny, *Deep learning and the Renormalization Group*, arXiv:1301.3124 (2013), where renormalization group techniques are explored.

[∗5-7∗] Sometimes we denote it as ℓ_2.

[∗5-8∗] For an overview of deep learning, see I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.