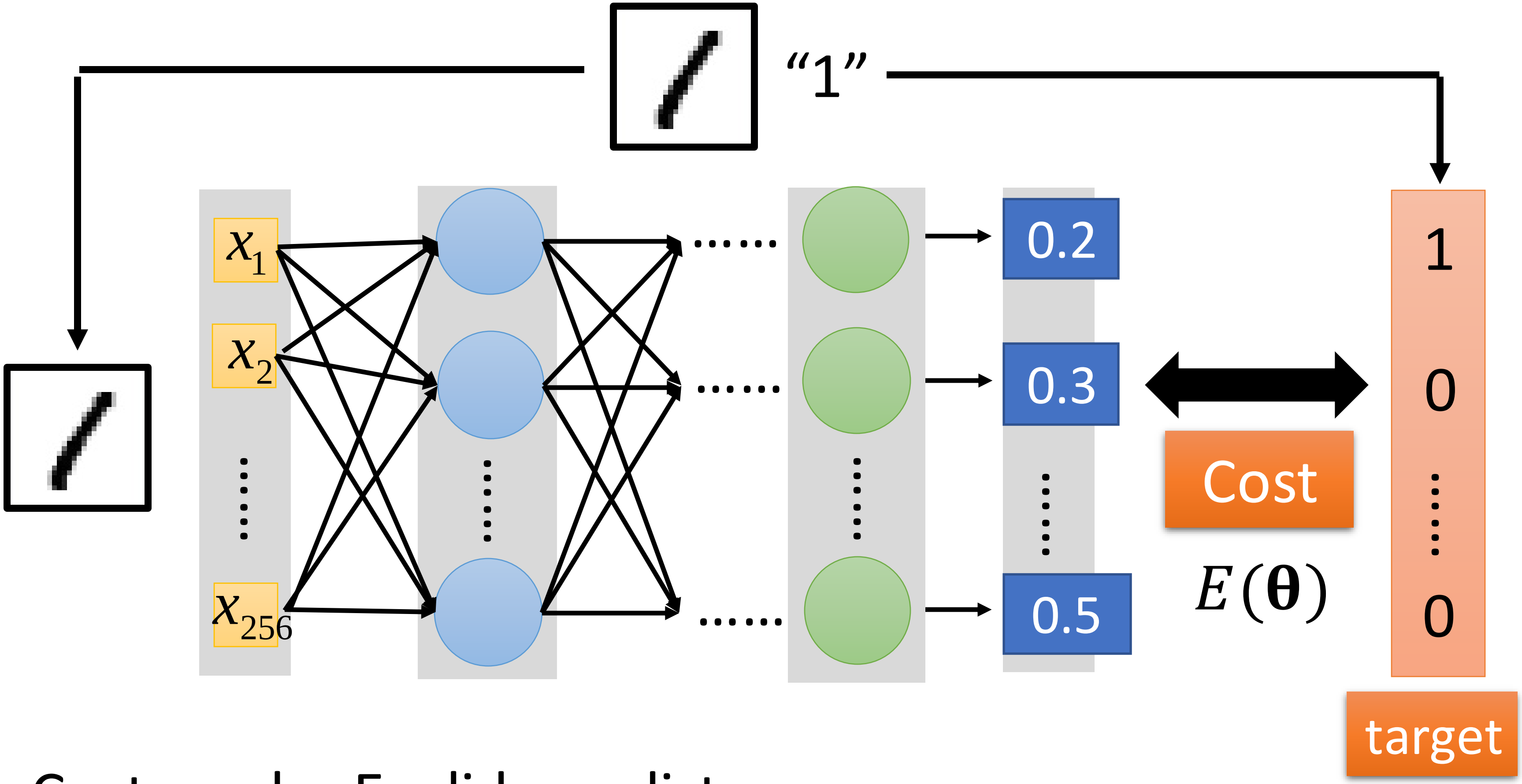
CSE463: Neural Networks

Backpropagation

by:
Hossam Abd El Munim
Computer & Systems Engineering Dept.,
Ain Shams University,
1 El-Sarayat Street, Abbassia, Cairo 11517

Cost

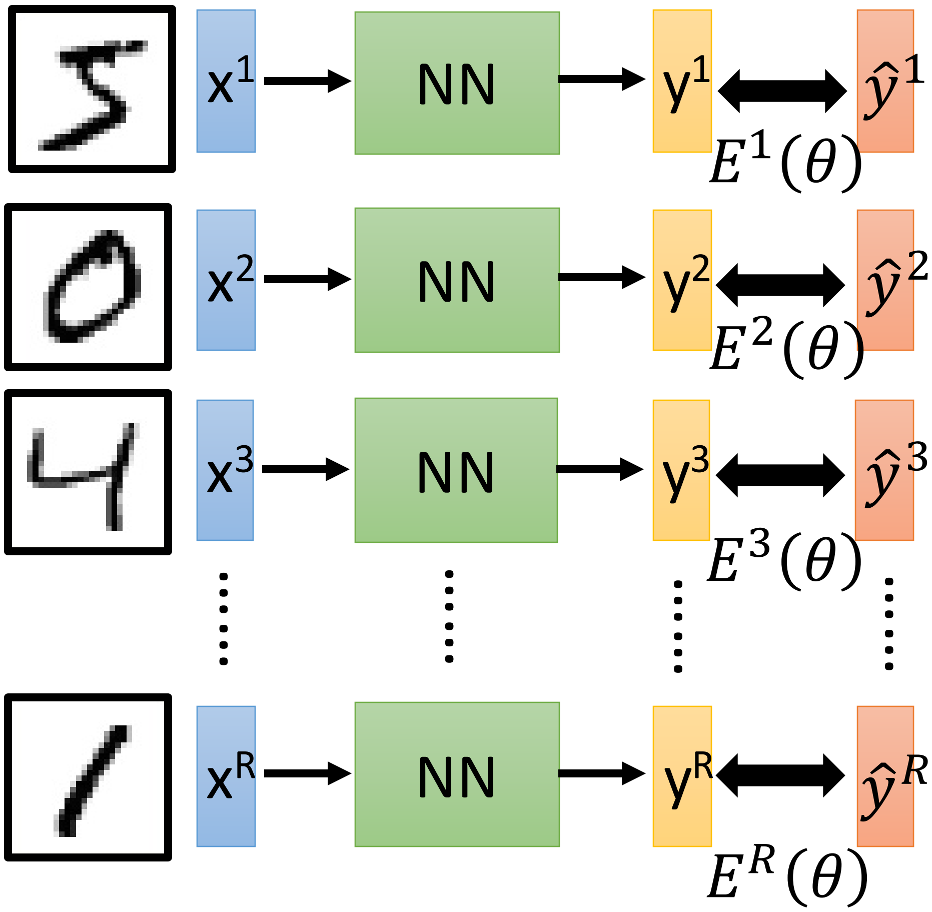
Given a set of network parameters θ, each example has a cost value.



Cost can be Euclidean distance or cross entropy of the network output and target

Total Cost

For all training data …



Total Cost:

$$E(\boldsymbol{\theta}) = \sum_{r=1}^{R} E^r(\boldsymbol{\theta})$$

How bad the network parameters $\boldsymbol{\theta}$ is on this task

Find the network parameters $\boldsymbol{\theta}^*$ that minimize this value

Problem Formulation

Given a data set, D = {($\mathbf{X}_1$, $\mathbf{Y}_1$), ($\mathbf{X}_2$, $\mathbf{Y}_2$)... ($\mathbf{X}_N$, $\mathbf{Y}_N$)} of labelled feature vectors where **Y** represents the target vectors designed as shown before. We to estimate the vector $\boldsymbol{\theta}$ through the conventional gradient descent:-

$$\boldsymbol{\theta}(\mathrm{t}+1)=\boldsymbol{\theta}(\mathrm{t})-\eta\frac{\partial \mathrm{E}(\boldsymbol{\theta}(\mathrm{t}))}{\partial \boldsymbol{\theta}}$$

Let us simplify the problem:-

$$\mathrm{E}(\boldsymbol{\theta}) = \frac{1}{2}||\widehat{\mathbf{Y}}(\boldsymbol{\theta}) - \mathbf{Y}||^2$$

$$\mathrm{E}(\boldsymbol{\theta}) = \frac{1}{2}(\widehat{\mathbf{Y}}(\boldsymbol{\theta}) - \mathbf{Y})^{\mathrm{T}}(\widehat{\mathbf{Y}}(\boldsymbol{\theta}) - \mathbf{Y})$$

$$\frac{\partial \mathrm{E}}{\partial \boldsymbol{\theta}} = (\widehat{\mathbf{Y}}(\boldsymbol{\theta}) - \mathbf{Y})^{\mathrm{T}} \frac{\partial \widehat{\mathbf{Y}}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$$

Vector Calculus

5.2 Partial Differentiation and Gradients

Differentiation as discussed in Section 5.1 applies to functions f of a scalar variable $x \in \mathbb{R}$. In the following, we consider the general case where the function f depends on one or more variables $\boldsymbol{x} \in \mathbb{R}^n$, e.g., $f(\boldsymbol{x}) = f(x_1, x_2)$. The generalization of the derivative to functions of several variables is the *gradient*.

We find the gradient of the function f with respect to $\boldsymbol{x}$ by *varying one variable at a time* and keeping the others constant. The gradient is then the collection of these *partial derivatives*.

Definition 5.5 (Partial Derivative). For a function $f : \mathbb{R}^n \to \mathbb{R}$, $\boldsymbol{x} \mapsto f(\boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^n$ of n variables $x_1, \ldots, x_n$ we define the *partial derivatives* as

$$\begin{aligned} \frac{\partial f}{\partial x_1} &= \lim_{h \to 0} \frac{f(x_1 + h, x_2, \ldots, x_n) - f(\boldsymbol{x})}{h} \\ &\vdots \\ \frac{\partial f}{\partial x_n} &= \lim_{h \to 0} \frac{f(x_1, \ldots, x_{n-1}, x_n + h) - f(\boldsymbol{x})}{h} \end{aligned} \tag{5.39}$$

and collect them in the row vector

$$\nabla_{\boldsymbol{x}} f = \operatorname{grad} f = \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \frac{\partial f(\boldsymbol{x})}{\partial x_1} & \frac{\partial f(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial f(\boldsymbol{x})}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{1 \times n}, \tag{5.40}$$

5.2.1 *Basic Rules of Partial Differentiation*

In the multivariate case, where $\boldsymbol{x} \in \mathbb{R}^n$, the basic differentiation rules that we know from school (e.g., sum rule, product rule, chain rule; see also Section 5.1.2) still apply. However, when we compute derivatives with respect to vectors $\boldsymbol{x} \in \mathbb{R}^n$ we need to pay attention: Our gradients now involve vectors and matrices, and matrix multiplication is not commutative (Section 2.2.1), i.e., the order matters.

Here are the general product rule, sum rule, and chain rule:

$$\text{Product rule:} \qquad \frac{\partial}{\partial \boldsymbol{x}}\big(f(\boldsymbol{x})g(\boldsymbol{x})\big) = \frac{\partial f}{\partial \boldsymbol{x}}g(\boldsymbol{x}) + f(\boldsymbol{x})\frac{\partial g}{\partial \boldsymbol{x}} \tag{5.46}$$

$$\text{Sum rule:} \qquad \frac{\partial}{\partial \boldsymbol{x}}\big(f(\boldsymbol{x}) + g(\boldsymbol{x})\big) = \frac{\partial f}{\partial \boldsymbol{x}} + \frac{\partial g}{\partial \boldsymbol{x}} \tag{5.47}$$

$$\text{Chain rule:} \qquad \frac{\partial}{\partial \boldsymbol{x}}(g \circ f)(\boldsymbol{x}) = \frac{\partial}{\partial \boldsymbol{x}}\big(g(f(\boldsymbol{x}))\big) = \frac{\partial g}{\partial f}\frac{\partial f}{\partial \boldsymbol{x}} \tag{5.48}$$

5.2.2 *Chain Rule*

Consider a function $f : \mathbb{R}^2 \to \mathbb{R}$ of two variables x_1, x_2. Furthermore, $x_1(t)$ and $x_2(t)$ are themselves functions of t. To compute the gradient of f with respect to t, we need to apply the chain rule (5.48) for multivariate functions as

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \begin{bmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{bmatrix} \begin{bmatrix} \frac{\partial x_1(t)}{\partial t} \\ \frac{\partial x_2(t)}{\partial t} \end{bmatrix} = \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial t}\,, \tag{5.49}$$

where d denotes the gradient and ∂ partial derivatives.

If $f(x_1, x_2)$ is a function of x_1 and x_2, where $x_1(s,t)$ and $x_2(s,t)$ are themselves functions of two variables s and t, the chain rule yields the partial derivatives

$$\frac{\partial f}{\partial s} = \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial s} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial s}\,, \tag{5.51}$$

$$\frac{\partial f}{\partial t} = \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial t}\,, \tag{5.52}$$

If $f(x_1, x_2)$ is a function of x_1 and x_2, where $x_1(s,t)$ and $x_2(s,t)$ are themselves functions of two variables s and t, the chain rule yields the partial derivatives

$$\frac{\partial f}{\partial s} = \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial s} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial s}, \tag{5.51}$$

$$\frac{\partial f}{\partial t} = \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2}\frac{\partial x_2}{\partial t}, \tag{5.52}$$

and the gradient is obtained by the matrix multiplication

$$\frac{\mathrm{d}f}{\mathrm{d}(s,t)} = \frac{\partial f}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}}{\partial (s,t)} = \underbrace{\begin{bmatrix}\frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2}\end{bmatrix}}_{=\frac{\partial f}{\partial \boldsymbol{x}}} \underbrace{\begin{bmatrix}\frac{\partial x_1}{\partial s} & \frac{\partial x_1}{\partial t} \\ \frac{\partial x_2}{\partial s} & \frac{\partial x_2}{\partial t}\end{bmatrix}}_{=\frac{\partial \boldsymbol{x}}{\partial (s,t)}}. \tag{5.53}$$

5.3 Gradients of Vector-Valued Functions

Thus far, we discussed partial derivatives and gradients of functions $f : \mathbb{R}^n \to \mathbb{R}$ mapping to the real numbers. In the following, we will generalize the concept of the gradient to vector-valued functions (vector fields) $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$, where $n \geqslant 1$ and $m > 1$.

For a function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ and a vector $\boldsymbol{x} = [x_1, \ldots, x_n]^\top \in \mathbb{R}^n$, the corresponding vector of function values is given as

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} f_1(\boldsymbol{x}) \\ \vdots \\ f_m(\boldsymbol{x}) \end{bmatrix} \in \mathbb{R}^m . \tag{5.54}$$

Writing the vector-valued function in this way allows us to view a vector-valued function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ as a vector of functions $[f_1, \ldots, f_m]^\top$, $f_i : \mathbb{R}^n \to \mathbb{R}$ that map onto $\mathbb{R}$. The differentiation rules for every f_i are exactly the ones we discussed in Section 5.2.

Therefore, the partial derivative of a vector-valued function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ with respect to $x_i \in \mathbb{R}$, $i = 1, \ldots n$, is given as the vector

$$\frac{\partial \boldsymbol{f}}{\partial x_i} = \begin{bmatrix} \frac{\partial f_1}{\partial x_i} \\ \vdots \\ \frac{\partial f_m}{\partial x_i} \end{bmatrix} = \begin{bmatrix} \lim_{h \to 0} \frac{f_1(x_1, \ldots, x_{i-1}, x_i + h, x_{i+1}, \ldots x_n) - f_1(\boldsymbol{x})}{h} \\ \vdots \\ \lim_{h \to 0} \frac{f_m(x_1, \ldots, x_{i-1}, x_i + h, x_{i+1}, \ldots x_n) - f_m(\boldsymbol{x})}{h} \end{bmatrix} \in \mathbb{R}^m . \tag{5.55}$$

From (5.40), we know that the gradient of f with respect to a vector is the row vector of the partial derivatives. In (5.55), every partial derivative $\partial \boldsymbol{f}/\partial x_i$ is a column vector. Therefore, we obtain the gradient of $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ with respect to $\boldsymbol{x} \in \mathbb{R}^n$ by collecting these partial derivatives:

$$\frac{\mathrm{d}\boldsymbol{f}(\boldsymbol{x})}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \boxed{\frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial x_1}} & \cdots & \boxed{\frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial x_n}} \end{bmatrix} \tag{5.56a}$$

$$= \begin{bmatrix} \boxed{\begin{matrix} \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} \\ \vdots \\ \frac{\partial f_m(\boldsymbol{x})}{\partial x_1} \end{matrix}} & \begin{matrix} \cdots \\ \\ \cdots \end{matrix} & \boxed{\begin{matrix} \frac{\partial f_1(\boldsymbol{x})}{\partial x_n} \\ \vdots \\ \frac{\partial f_m(\boldsymbol{x})}{\partial x_n} \end{matrix}} \end{bmatrix} \in \mathbb{R}^{m \times n} . \tag{5.56b}$$

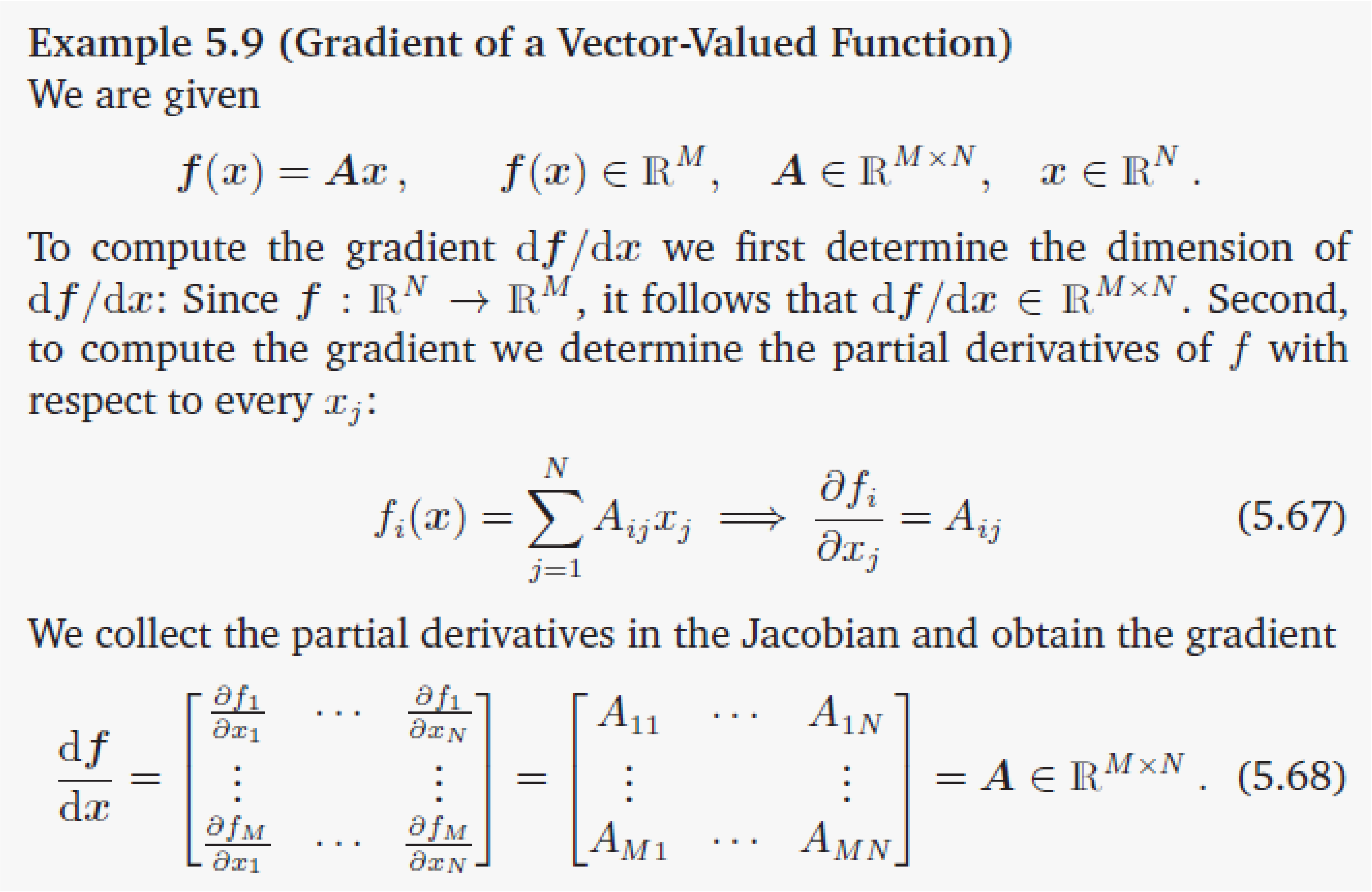
Definition 5.6 (Jacobian). The collection of all first-order partial derivatives of a vector-valued function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ is called the *Jacobian*. The Jacobian $\boldsymbol{J}$ is an $m \times n$ matrix, which we define and arrange as follows:

$$\boldsymbol{J} = \nabla_{\boldsymbol{x}} \boldsymbol{f} = \frac{\mathrm{d}\boldsymbol{f}(\boldsymbol{x})}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \dfrac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial x_1} & \cdots & \dfrac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial x_n} \end{bmatrix} \tag{5.57}$$

$$= \begin{bmatrix} \dfrac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \cdots & \dfrac{\partial f_1(\boldsymbol{x})}{\partial x_n} \\ \vdots & & \vdots \\ \dfrac{\partial f_m(\boldsymbol{x})}{\partial x_1} & \cdots & \dfrac{\partial f_m(\boldsymbol{x})}{\partial x_n} \end{bmatrix}, \tag{5.58}$$

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad J(i,j) = \frac{\partial f_i}{\partial x_j}. \tag{5.59}$$

As a special case of (5.58), a function $f : \mathbb{R}^n \to \mathbb{R}^1$, which maps a vector $\boldsymbol{x} \in \mathbb{R}^n$ onto a scalar (e.g., $f(\boldsymbol{x}) = \sum_{i=1}^{n} x_i$), possesses a Jacobian that is a row vector (matrix of dimension $1 \times n$); see (5.40).



Example 5.9 (Gradient of a Vector-Valued Function)

We are given

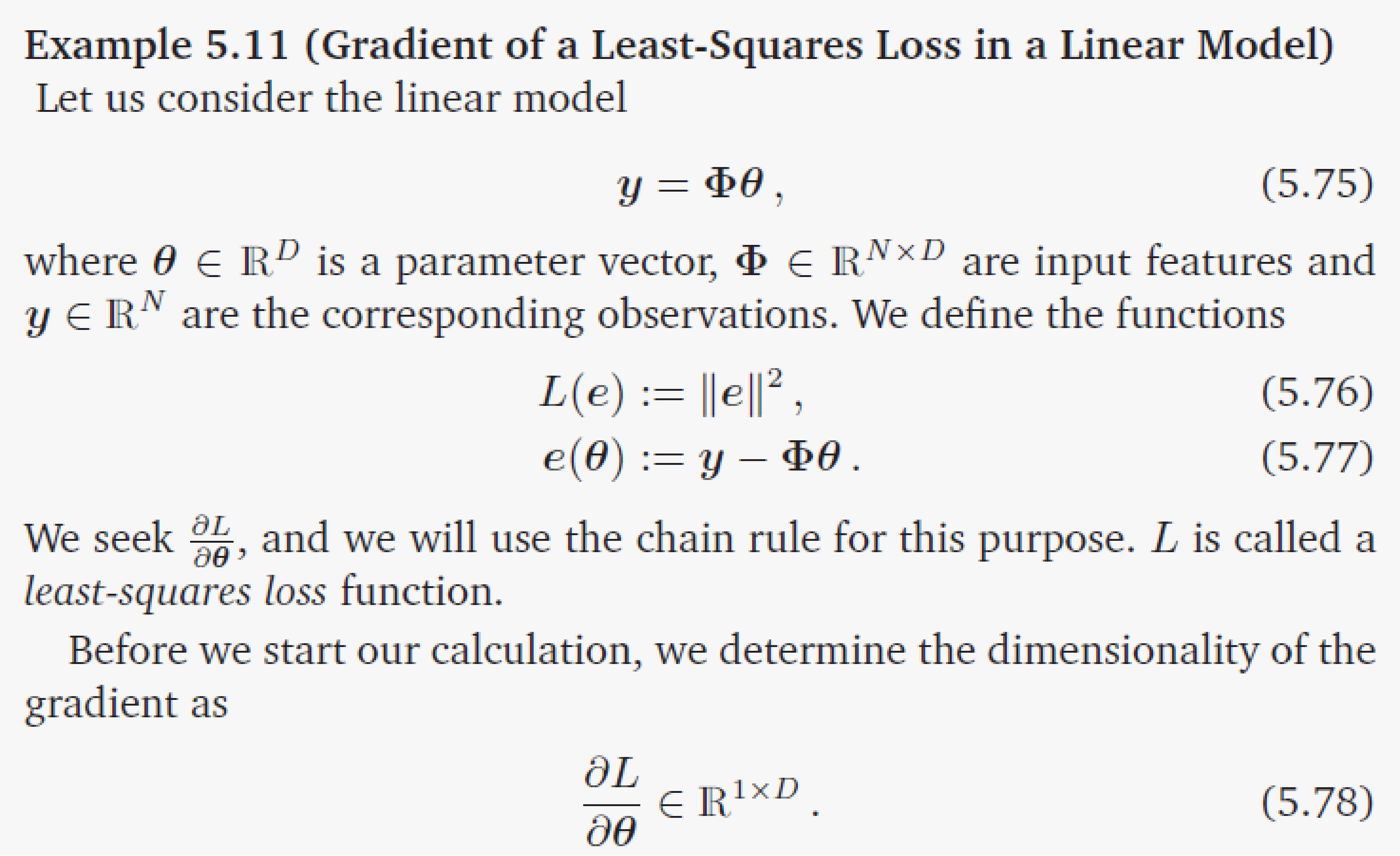
$$f(x) = Ax\,, \qquad f(x) \in \mathbb{R}^M, \quad A \in \mathbb{R}^{M\times N}, \quad x \in \mathbb{R}^N\,.$$

To compute the gradient $\mathrm{d}f/\mathrm{d}x$ we first determine the dimension of $\mathrm{d}f/\mathrm{d}x$: Since $f : \mathbb{R}^N \to \mathbb{R}^M$, it follows that $\mathrm{d}f/\mathrm{d}x \in \mathbb{R}^{M\times N}$. Second, to compute the gradient we determine the partial derivatives of f with respect to every x_j:

$$f_i(x) = \sum_{j=1}^{N} A_{ij}x_j \implies \frac{\partial f_i}{\partial x_j} = A_{ij} \tag{5.67}$$

We collect the partial derivatives in the Jacobian and obtain the gradient

$$\frac{\mathrm{d}f}{\mathrm{d}x} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_N} \\ \vdots & & \vdots \\ \frac{\partial f_M}{\partial x_1} & \cdots & \frac{\partial f_M}{\partial x_N} \end{bmatrix} = \begin{bmatrix} A_{11} & \cdots & A_{1N} \\ \vdots & & \vdots \\ A_{M1} & \cdots & A_{MN} \end{bmatrix} = A \in \mathbb{R}^{M\times N}\,. \tag{5.68}$$



Example 5.11 (Gradient of a Least-Squares Loss in a Linear Model) Let us consider the linear model

$$\boldsymbol{y} = \boldsymbol{\Phi\theta}\,, \tag{5.75}$$

where $\boldsymbol{\theta} \in \mathbb{R}^D$ is a parameter vector, $\boldsymbol{\Phi} \in \mathbb{R}^{N \times D}$ are input features and $\boldsymbol{y} \in \mathbb{R}^N$ are the corresponding observations. We define the functions

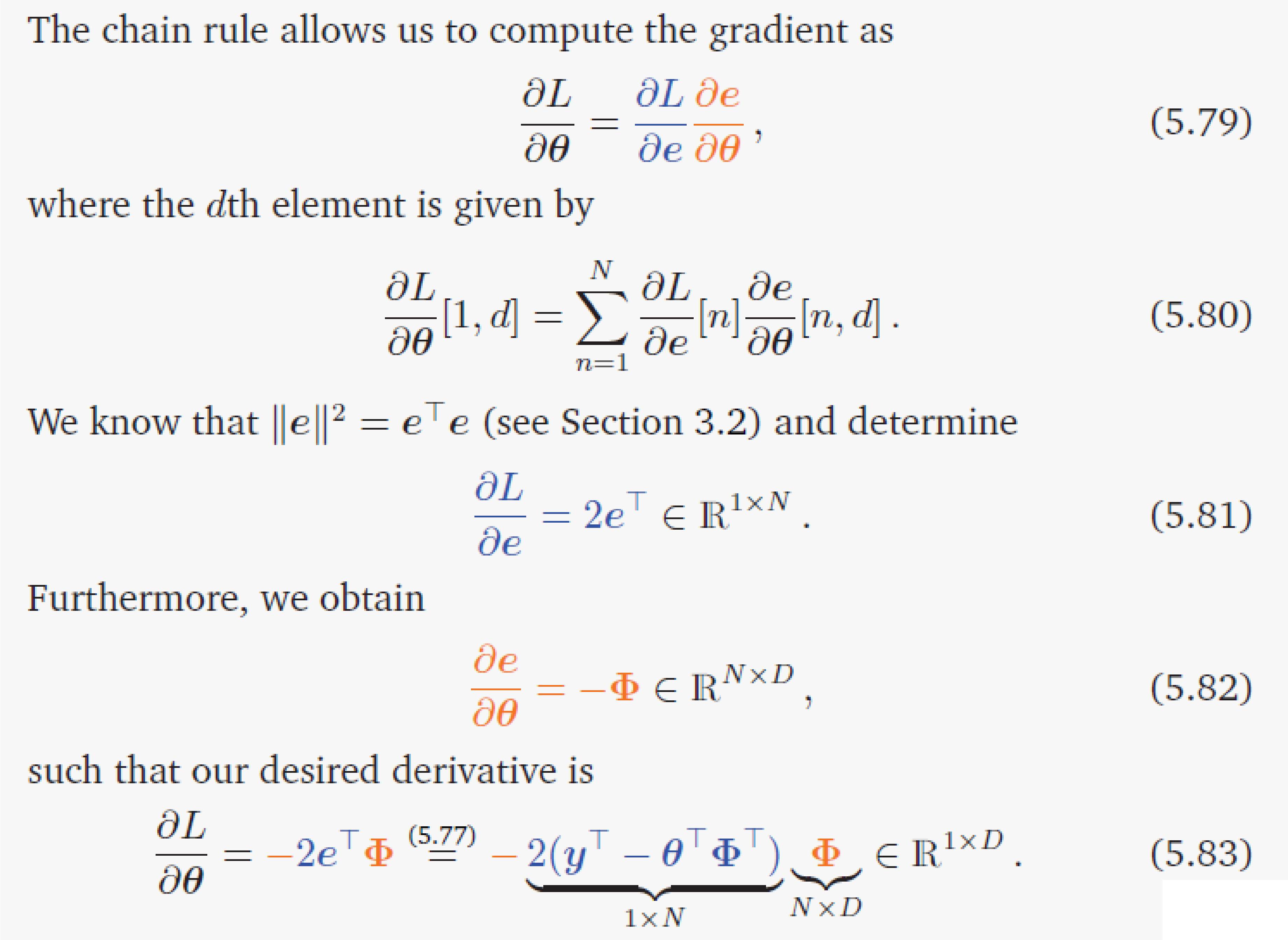
$$L(\boldsymbol{e}) := \|\boldsymbol{e}\|^2\,, \tag{5.76}$$

$$\boldsymbol{e}(\boldsymbol{\theta}) := \boldsymbol{y} - \boldsymbol{\Phi\theta}\,. \tag{5.77}$$

We seek $\frac{\partial L}{\partial \boldsymbol{\theta}}$, and we will use the chain rule for this purpose. L is called a *least-squares loss* function.

Before we start our calculation, we determine the dimensionality of the gradient as

$$\frac{\partial L}{\partial \boldsymbol{\theta}} \in \mathbb{R}^{1 \times D}\,. \tag{5.78}$$



The chain rule allows us to compute the gradient as

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = \frac{\partial L}{\partial \boldsymbol{e}} \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{\theta}}, \tag{5.79}$$

where the dth element is given by

$$\frac{\partial L}{\partial \boldsymbol{\theta}}[1, d] = \sum_{n=1}^{N} \frac{\partial L}{\partial \boldsymbol{e}}[n] \frac{\partial \boldsymbol{e}}{\partial \boldsymbol{\theta}}[n, d]. \tag{5.80}$$

We know that $\|\boldsymbol{e}\|^2 = \boldsymbol{e}^\top \boldsymbol{e}$ (see Section 3.2) and determine

$$\frac{\partial L}{\partial \boldsymbol{e}} = 2\boldsymbol{e}^\top \in \mathbb{R}^{1 \times N}. \tag{5.81}$$

Furthermore, we obtain

$$\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{\theta}} = -\boldsymbol{\Phi} \in \mathbb{R}^{N \times D}, \tag{5.82}$$

such that our desired derivative is

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = -2\boldsymbol{e}^\top \boldsymbol{\Phi} \overset{(5.77)}{=} -\underbrace{2(\boldsymbol{y}^\top - \boldsymbol{\theta}^\top \boldsymbol{\Phi}^\top)}_{1 \times N} \underbrace{\boldsymbol{\Phi}}_{N \times D} \in \mathbb{R}^{1 \times D}. \tag{5.83}$$

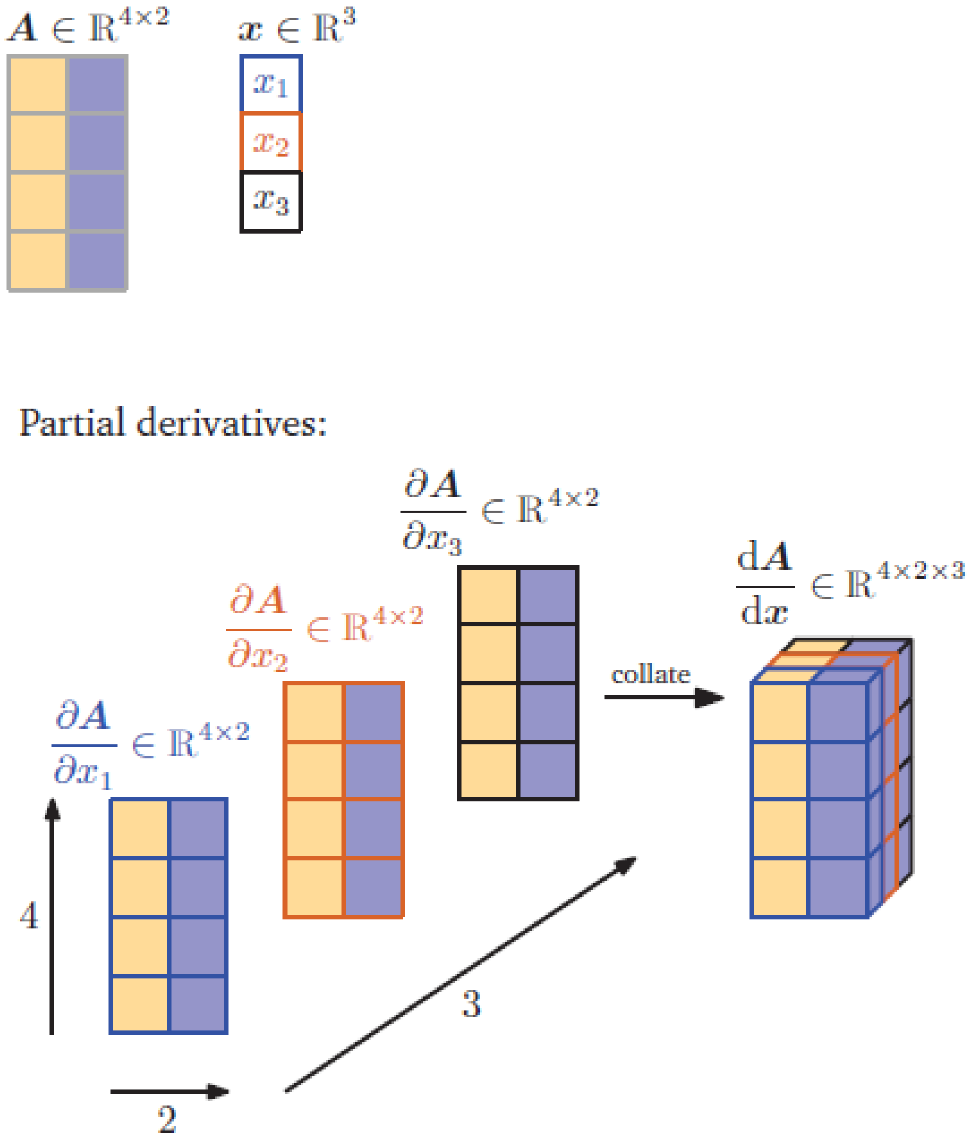
Remark. We would have obtained the same result without using the chain rule by immediately looking at the function

$$L_2(\boldsymbol{\theta}) := \|\boldsymbol{y} - \boldsymbol{\Phi\theta}\|^2 = (\boldsymbol{y} - \boldsymbol{\Phi\theta})^\top(\boldsymbol{y} - \boldsymbol{\Phi\theta})\,. \tag{5.84}$$

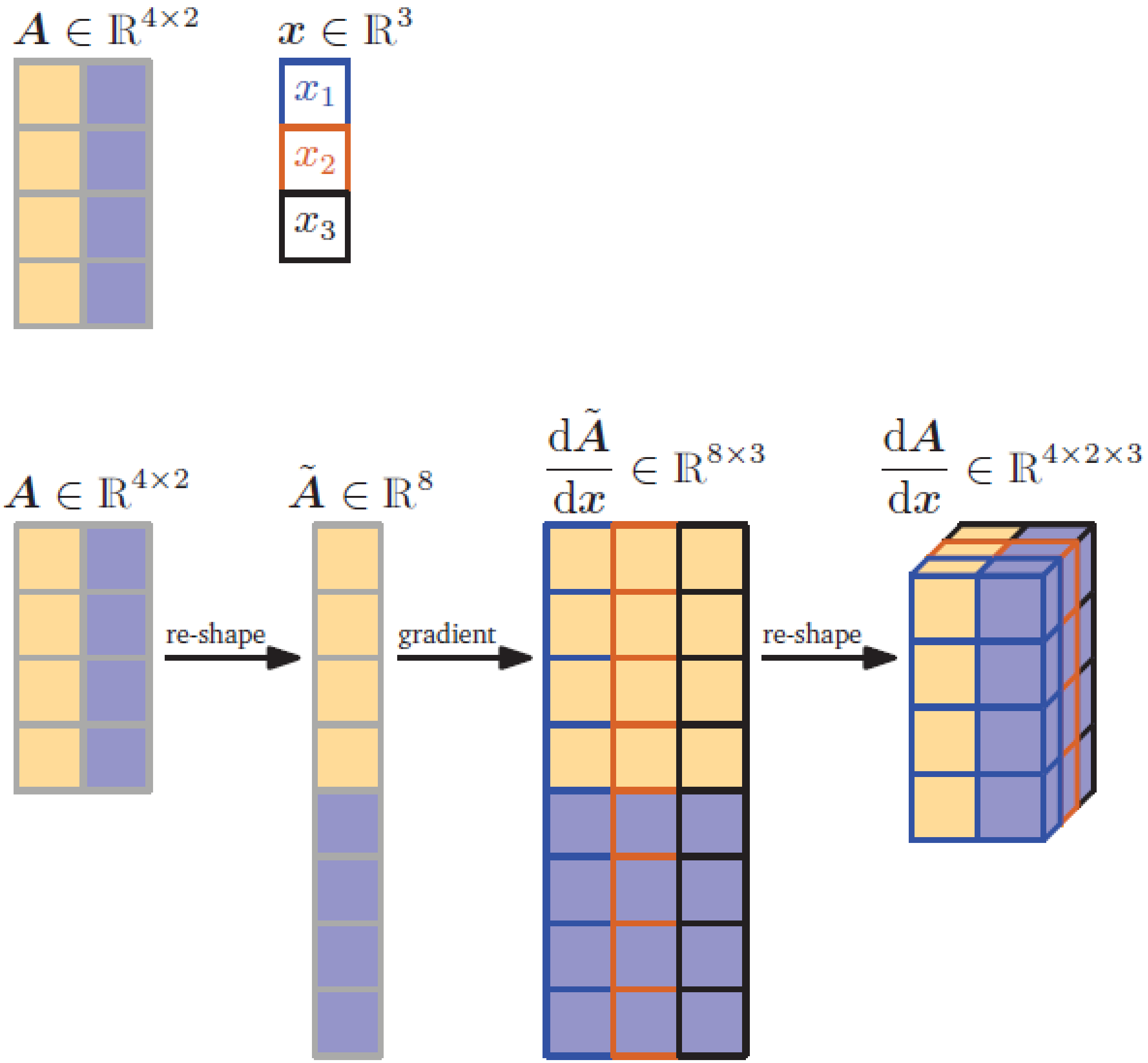
This approach is still practical for simple functions like L_2 but becomes impractical for deep function compositions. ♢

5.4 Gradients of Matrices

We will encounter situations where we need to take gradients of matrices with respect to vectors (or other matrices), which results in a multidimensional tensor. We can think of this tensor as a multidimensional array that collects partial derivatives. For example, if we compute the gradient of an $m \times n$ matrix $\boldsymbol{A}$ with respect to a $p \times q$ matrix $\boldsymbol{B}$, the resulting Jacobian would be $(m\times n)\times(p\times q)$, i.e., a four-dimensional tensor $\boldsymbol{J}$, whose entries are given as $J_{ijkl} = \partial A_{ij}/\partial B_{kl}$.


$\boldsymbol{A} \in \mathbb{R}^{4\times 2}$
$\boldsymbol{x} \in \mathbb{R}^{3}$
x_1
x_2
x_3
Partial derivatives:
$\frac{\partial \boldsymbol{A}}{\partial x_3} \in \mathbb{R}^{4\times 2}$
$\frac{\partial \boldsymbol{A}}{\partial x_2} \in \mathbb{R}^{4\times 2}$
$\frac{\partial \boldsymbol{A}}{\partial x_1} \in \mathbb{R}^{4\times 2}$
$\frac{\mathrm{d}\boldsymbol{A}}{\mathrm{d}\boldsymbol{x}} \in \mathbb{R}^{4\times 2\times 3}$
collate
4
2
3

(a) Approach 1: We compute the partial derivative $\frac{\partial \boldsymbol{A}}{\partial x_1}, \frac{\partial \boldsymbol{A}}{\partial x_2}, \frac{\partial \boldsymbol{A}}{\partial x_3}$, each of which is a 4×2 matrix, and collate them in a $4 \times 2 \times 3$ tensor.


$\boldsymbol{A} \in \mathbb{R}^{4\times 2}$
$\boldsymbol{x} \in \mathbb{R}^{3}$
x_1
x_2
x_3
$\boldsymbol{A} \in \mathbb{R}^{4\times 2}$
re-shape
$\tilde{\boldsymbol{A}} \in \mathbb{R}^{8}$
gradient
$\frac{\mathrm{d}\tilde{\boldsymbol{A}}}{\mathrm{d}\boldsymbol{x}} \in \mathbb{R}^{8\times 3}$
re-shape
$\frac{\mathrm{d}\boldsymbol{A}}{\mathrm{d}\boldsymbol{x}} \in \mathbb{R}^{4\times 2\times 3}$

(b) Approach 2: We re-shape (flatten) $\boldsymbol{A} \in \mathbb{R}^{4\times 2}$ into a vector $\tilde{\boldsymbol{A}} \in \mathbb{R}^{8}$. Then, we compute the gradient $\frac{\mathrm{d}\tilde{\boldsymbol{A}}}{\mathrm{d}\boldsymbol{x}} \in \mathbb{R}^{8\times 3}$. We obtain the gradient tensor by re-shaping this gradient as illustrated above.

Example 5.12 (Gradient of Vectors with Respect to Matrices)

Let us consider the following example, where

$$\boldsymbol{f} = \boldsymbol{A}\boldsymbol{x}\,, \quad \boldsymbol{f} \in \mathbb{R}^M, \quad \boldsymbol{A} \in \mathbb{R}^{M\times N}, \quad \boldsymbol{x} \in \mathbb{R}^N \tag{5.85}$$

and where we seek the gradient $\mathrm{d}\boldsymbol{f}/\mathrm{d}\boldsymbol{A}$. Let us start again by determining the dimension of the gradient as

$$\frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{A}} \in \mathbb{R}^{M\times(M\times N)}\,. \tag{5.86}$$

By definition, the gradient is the collection of the partial derivatives:

$$\frac{\mathrm{d}\boldsymbol{f}}{\mathrm{d}\boldsymbol{A}} = \begin{bmatrix} \frac{\partial f_1}{\partial \boldsymbol{A}} \\ \vdots \\ \frac{\partial f_M}{\partial \boldsymbol{A}} \end{bmatrix}, \quad \frac{\partial f_i}{\partial \boldsymbol{A}} \in \mathbb{R}^{1\times(M\times N)}\,. \tag{5.87}$$

To compute the partial derivatives, it will be helpful to explicitly write out the matrix vector multiplication:

$$f_i = \sum_{j=1}^{N} A_{ij}x_j, \quad i = 1, \dots, M\,, \tag{5.88}$$

and the partial derivatives are then given as

$$\frac{\partial f_i}{\partial A_{iq}} = x_q \, . \tag{5.89}$$

This allows us to compute the partial derivatives of f_i with respect to a row of $\boldsymbol{A}$, which is given as

$$\frac{\partial f_i}{\partial A_{i,:}} = \boldsymbol{x}^\top \in \mathbb{R}^{1\times 1\times N} \, , \tag{5.90}$$

$$\frac{\partial f_i}{\partial A_{k\neq i,:}} = \boldsymbol{0}^\top \in \mathbb{R}^{1\times 1\times N} \tag{5.91}$$

where we have to pay attention to the correct dimensionality. Since f_i maps onto $\mathbb{R}$ and each row of $\boldsymbol{A}$ is of size $1 \times N$, we obtain a $1 \times 1 \times N$-sized tensor as the partial derivative of f_i with respect to a row of $\boldsymbol{A}$.

We stack the partial derivatives (5.91) and get the desired gradient in (5.87) via

$$\frac{\partial f_i}{\partial \boldsymbol{A}} = \begin{bmatrix} \boldsymbol{0}^\top \\ \vdots \\ \boldsymbol{0}^\top \\ \boldsymbol{x}^\top \\ \boldsymbol{0}^\top \\ \vdots \\ \boldsymbol{0}^\top \end{bmatrix} \in \mathbb{R}^{1\times (M\times N)} \, . \tag{5.92}$$

5.6 Backpropagation and Automatic Differentiation

In many machine learning applications, we find good model parameters by performing gradient descent (Section 7.1), which relies on the fact that we can compute the gradient of a learning objective with respect to the parameters of the model. For a given objective function, we can obtain the gradient with respect to the model parameters using calculus and applying the chain rule; see Section 5.2.2. We already had a taste in Section 5.3 when we looked at the gradient of a squared loss with respect to the parameters of a linear regression model.

Consider the function

$$f(x) = \sqrt{x^2 + \exp(x^2)} + \cos\left(x^2 + \exp(x^2)\right) . \tag{5.109}$$

By application of the chain rule, and noting that differentiation is linear, we compute the gradient

$$\begin{aligned}\frac{\mathrm{d}f}{\mathrm{d}x} &= \frac{2x + 2x\exp(x^2)}{2\sqrt{x^2 + \exp(x^2)}} - \sin\left(x^2 + \exp(x^2)\right)\left(2x + 2x\exp(x^2)\right) \\ &= 2x\left(\frac{1}{2\sqrt{x^2 + \exp(x^2)}} - \sin\left(x^2 + \exp(x^2)\right)\right)\left(1 + \exp(x^2)\right) .\end{aligned} \tag{5.110}$$

Writing out the gradient in this explicit way is often impractical since it often results in a very lengthy expression for a derivative. In practice, it means that, if we are not careful, the implementation of the gradient could be significantly more expensive than computing the function, which imposes unnecessary overhead. For training deep neural network models, the *backpropagation* algorithm (Kelley, 1960; Bryson, 1961; Dreyfus, 1962; Rumelhart et al., 1986) is an efficient way to compute the gradient of an error function with respect to the parameters of the model.

5.6.1 Gradients in a Deep Network

An area where the chain rule is used to an extreme is deep learning, where the function value $\boldsymbol{y}$ is computed as a many-level function composition

$$\boldsymbol{y} = (f_K \circ f_{K-1} \circ \cdots \circ f_1)(\boldsymbol{x}) = f_K(f_{K-1}(\cdots(f_1(\boldsymbol{x}))\cdots)), \quad (5.111)$$

where $\boldsymbol{x}$ are the inputs (e.g., images), $\boldsymbol{y}$ are the observations (e.g., class labels), and every function f_i, $i = 1, \ldots, K$, possesses its own parameters.

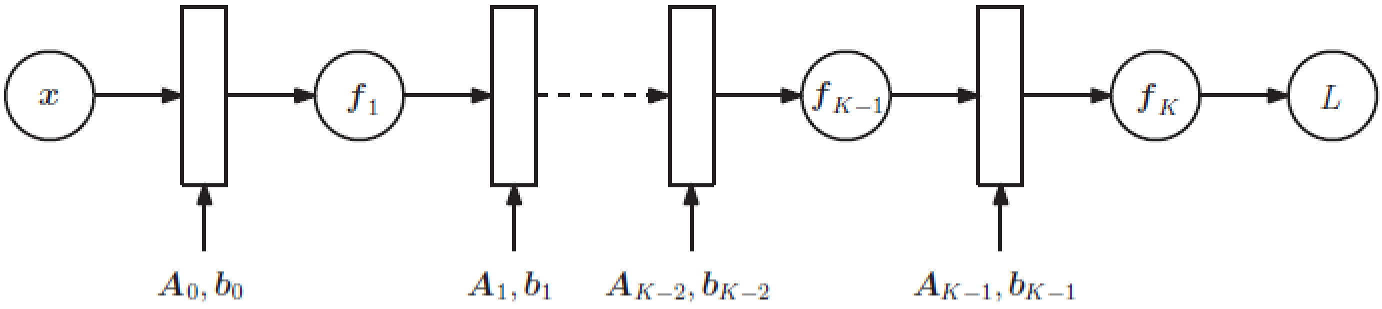


Figure 5.8 Forward pass in a multi-layer neural network to compute the loss L as a function of the inputs x and the parameters $\boldsymbol{A}_i,\ \boldsymbol{b}_i$.

In neural networks with multiple layers, we have functions $f_i(\boldsymbol{x}_{i-1}) = \sigma(\boldsymbol{A}_{i-1}\boldsymbol{x}_{i-1} + \boldsymbol{b}_{i-1})$ in the ith layer. Here $\boldsymbol{x}_{i-1}$ is the output of layer $i-1$ and σ an activation function, such as the logistic sigmoid $\frac{1}{1+e^{-x}}$, tanh or a rectified linear unit (ReLU). In order to train these models, we require the gradient of a loss function L with respect to all model parameters $\boldsymbol{A}_j, \boldsymbol{b}_j$ for $j = 1, \dots, K$. This also requires us to compute the gradient of L with respect to the inputs of each layer. For example, if we have inputs $\boldsymbol{x}$ and observations $\boldsymbol{y}$ and a network structure defined by

$$\boldsymbol{f}_0 := \boldsymbol{x} \tag{5.112}$$

$$\boldsymbol{f}_i := \sigma_i(\boldsymbol{A}_{i-1}\boldsymbol{f}_{i-1} + \boldsymbol{b}_{i-1}), \quad i = 1, \dots, K, \tag{5.113}$$

see also Figure 5.8 for a visualization, we may be interested in finding $\boldsymbol{A}_j, \boldsymbol{b}_j$ for $j = 0, \dots, K-1$, such that the squared loss

$$L(\boldsymbol{\theta}) = \|\boldsymbol{y} - \boldsymbol{f}_K(\boldsymbol{\theta}, \boldsymbol{x})\|^2 \tag{5.114}$$

is minimized, where $\boldsymbol{\theta} = \{\boldsymbol{A}_0, \boldsymbol{b}_0, \dots, \boldsymbol{A}_{K-1}, \boldsymbol{b}_{K-1}\}$.

To obtain the gradients with respect to the parameter set $\boldsymbol{\theta}$, we require the partial derivatives of L with respect to the parameters $\boldsymbol{\theta}_j = \{\boldsymbol{A}_j, \boldsymbol{b}_j\}$ of each layer $j = 0, \ldots, K-1$. The chain rule allows us to determine the partial derivatives as

$$\frac{\partial L}{\partial \boldsymbol{\theta}_{K-1}} = \frac{\partial L}{\partial \boldsymbol{f}_K} \frac{\partial \boldsymbol{f}_K}{\partial \boldsymbol{\theta}_{K-1}} \tag{5.115}$$

$$\frac{\partial L}{\partial \boldsymbol{\theta}_{K-2}} = \frac{\partial L}{\partial \boldsymbol{f}_K} \boxed{\frac{\partial \boldsymbol{f}_K}{\partial \boldsymbol{f}_{K-1}} \frac{\partial \boldsymbol{f}_{K-1}}{\partial \boldsymbol{\theta}_{K-2}}} \tag{5.116}$$

$$\frac{\partial L}{\partial \boldsymbol{\theta}_{K-3}} = \frac{\partial L}{\partial \boldsymbol{f}_K} \frac{\partial \boldsymbol{f}_K}{\partial \boldsymbol{f}_{K-1}} \boxed{\frac{\partial \boldsymbol{f}_{K-1}}{\partial \boldsymbol{f}_{K-2}} \frac{\partial \boldsymbol{f}_{K-2}}{\partial \boldsymbol{\theta}_{K-3}}} \tag{5.117}$$

$$\frac{\partial L}{\partial \boldsymbol{\theta}_{i}} = \frac{\partial L}{\partial \boldsymbol{f}_K} \frac{\partial \boldsymbol{f}_K}{\partial \boldsymbol{f}_{K-1}} \cdots \boxed{\frac{\partial \boldsymbol{f}_{i+2}}{\partial \boldsymbol{f}_{i+1}} \frac{\partial \boldsymbol{f}_{i+1}}{\partial \boldsymbol{\theta}_{i}}} \tag{5.118}$$

The **orange** terms are partial derivatives of the output of a layer with respect to its inputs, whereas the **blue** terms are partial derivatives of the output of a layer with respect to its parameters. Assuming, we have already computed the partial derivatives $\partial L / \partial \boldsymbol{\theta}_{i+1}$, then most of the computation can be reused to compute $\partial L / \partial \boldsymbol{\theta}_i$. The additional terms that we

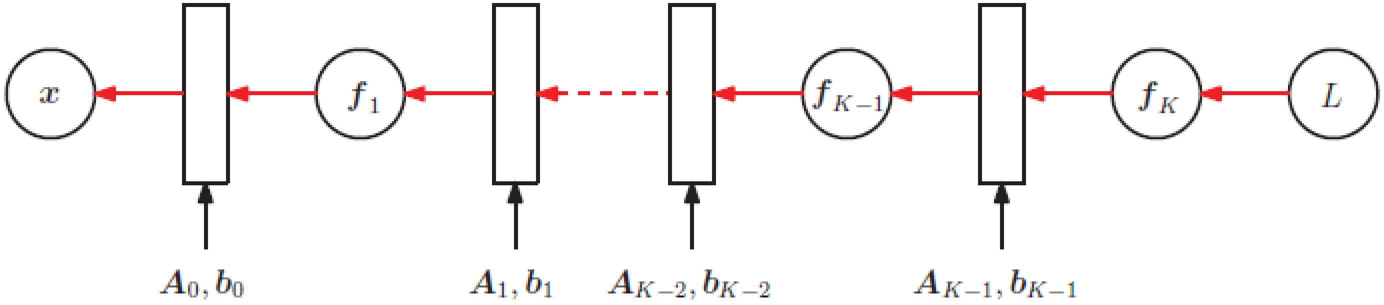
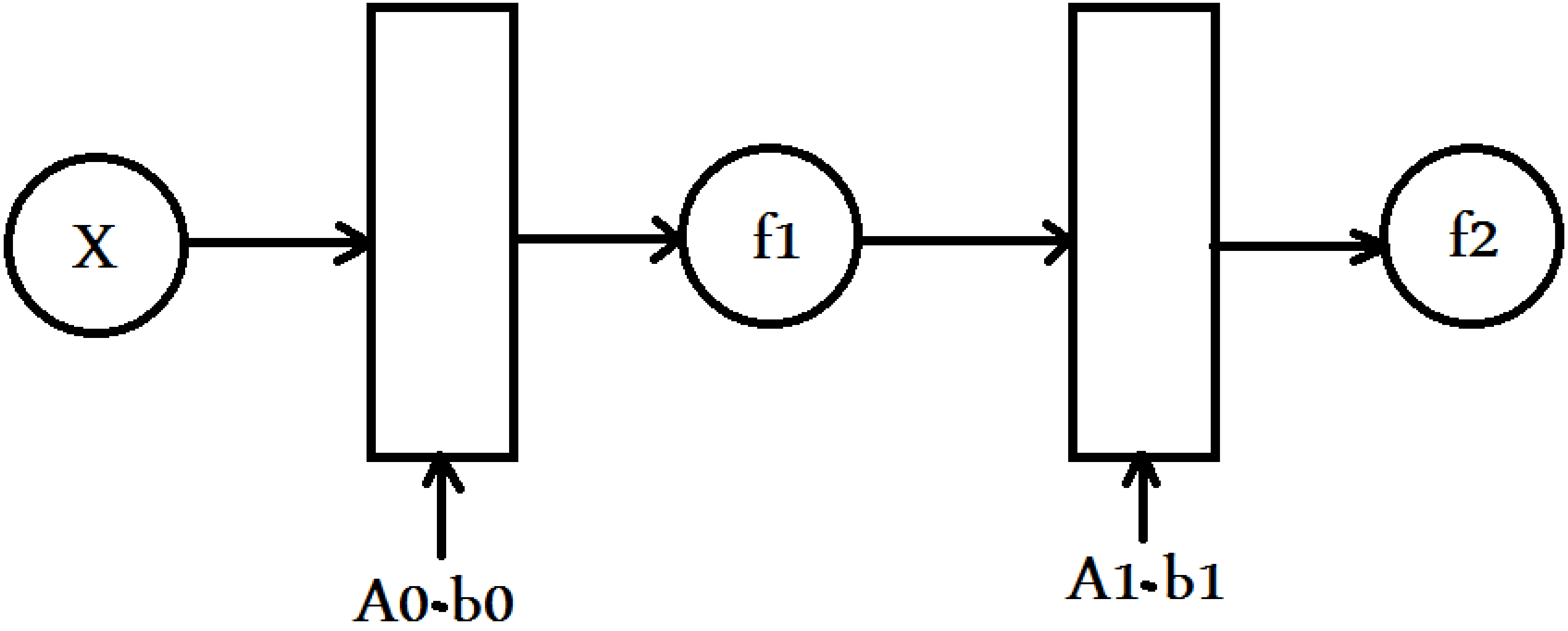


Figure 5.9 Backward pass in a multi-layer neural network to compute the gradients of the loss function.

need to compute are indicated by the boxes. Figure 5.9 visualizes that the gradients are passed backward through the network.

Specific Example



Forward Path

$\mathbf{f_0 = X}$

$\mathbf{Z_1 = A_0 f_0 + b_0}$

$\mathbf{f_1} = \sigma\mathbf{(Z1)}$

$\mathbf{Z_2 = A_1 f_1 + b_1}$

$\mathbf{f_2} = \sigma\mathbf{(Z_2)}$

$E = (1/2)\ (\mathbf{f}_2 - \mathbf{Y})^T(\mathbf{f}_2 - \mathbf{Y})$

Backward Path (1)

$$\partial E/\partial \mathbf{b_1}=(\mathbf{f}_2-\mathbf{Y})^T(\partial \mathbf{f}_2/\partial \mathbf{Z}_2)(\partial \mathbf{Z}_2/\partial \mathbf{b_1})$$

$(\partial \mathbf{Z2}/\partial \mathbf{b_1})$ = Identity Matrix

$(\partial \mathbf{f}_2/\partial \mathbf{Z2})$ = Diagonal matrix of σ'

Backward Path (2)

$$\underbrace{\partial E/\partial \mathbf{b}_0}_{\text{Row Vector}}=(\mathbf{f}_2-\mathbf{Y})^T\underbrace{(\partial \mathbf{f}_2/\partial \mathbf{Z}_2)}_{\text{Diagonal}}\underbrace{(\partial \mathbf{Z}_2/\partial \mathbf{f}_1)}_{\mathbf{A}_1}\underbrace{(\partial \mathbf{f}_1/\partial \mathbf{Z}_1)}_{\text{Diagonal}}\underbrace{(\partial \mathbf{Z}_1/\partial \mathbf{b}_0)}_{\text{Identity}}$$

Backward Path (3): Simplifying the derivative w.r.t a matrix

$$\partial E/\partial \alpha=(\mathbf{f}_2-\mathbf{Y})^T(\partial \mathbf{f}_2/\partial \mathbf{Z}_2)(\partial \mathbf{Z}_2/\partial \alpha)$$

$$\partial \mathbf{Z}_2/\partial \alpha = [\partial(\mathbf{A}_1)/\partial \alpha]\mathbf{f}_1$$

$$\alpha \in \mathbf{A}_1$$

Backward Path (4): Simplifying the derivative w.r.t a matrix

$$\partial E/\partial \beta=(\mathbf{f}_2-\mathbf{Y})^T(\partial \mathbf{f}_2/\partial \mathbf{Z}_2)(\partial \mathbf{Z}_2/\partial \mathbf{f}_1)\ (\partial \mathbf{f}_1/\partial \mathbf{Z}_1)(\partial \mathbf{Z}_1/\partial \beta)$$

$$\partial \mathbf{Z}_1/\partial \beta = [\partial(\mathbf{A}_0)/\partial \beta]\mathbf{f}_0$$

$$\beta \in \mathbf{A}_0$$