5.1 Differentiation of Univariate Functions

① difference quotient

$$\frac{dy}{dx} = \frac{f(x+dx) - f(x)}{x + dx - x}$$

② derivative

$$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{x+h-x}$$

③ Taylor polynomial $\longrightarrow$ Taylor series $\supset$ McClaren series

$$T^{(n)}(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x_0 - x)^k \implies T^{\infty}(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(x_0 - x)^k$$

$f \in C^{\infty}$ 모든 구간에서 연속 + 미분 가능

$\Downarrow$

McClaren series $=$ $x_0 = 0$ 인 때의 Taylor series

Remark. A Taylor series is a special case of a power series 멱급수

$$f(x) = \sum_{k=0}^{\infty} a_k (x - c)^k \tag{5.28}$$

where a_k are coefficients and c is a constant, which has the special form in Definition 5.4. ◇

5.2 Partial Differentiation and Gradients

- Gradients = Collection of partial derivatives.

  "Jacobian 이라고 부르기도 함"

- $\nabla_x f = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial x_3}, \cdots \frac{\partial f}{\partial x_n} \right]$

  f : Scalar-valued function

5.3 Gradients of Vector-Valued Functions

- Jacobian = all collection of first-order derivatives of vector-valued functions $f : R^n \to R^m$.

- $J = \nabla_x f = \begin{bmatrix} \frac{\partial f_1}{\partial \vec{x}} \\ \frac{\partial f_2}{\partial \vec{x}} \\ \vdots \\ \frac{\partial f_m}{\partial \vec{x}} \end{bmatrix}$ m×n matrix

  f : Vector-valued function

- Jacobian (partial derivatives) ⟹ 좌표 변환행렬 !!

- $|J|$ = scaling factor!

J ⟹ 좌표변환행렬

$|J|$ ⟹ 변환시 평행사변형 넓이 scaling factor

- $\vec{c} \to \vec{b}$

  ⇓

  평행사변형 넓이 (Det) 변화시키는 것!

  (이것을 변환행렬 찾기)

  ⟹ J

Why J?

c가 변할 때, b가 변하는 정도를 보는 것!

$f(c) = b$.

$\frac{\partial f}{\partial c}$ ⟹ J

- Chain-rule 에서의 Jacobian의 역할!

ex) Least-square Loss 계산에서 활용!

$\Longrightarrow$

$L(e) = \|e\|^2$

$e(\theta) = (y - \Phi\theta)$

$\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial e} \frac{\partial e}{\partial \theta}$

gradient $1 \times N$ Jacobian $N \times D$

5.4 Gradients of Matrices

5.5 Useful Identities for Computing Gradients

5.6 Backpropagation and Automatic Differentiation

- gradient descent $\Longrightarrow$ find parameter.

Chain-rule로 partial derivative 해 구하다!

↳ Backpropagation 으로 쉽게 구할 수 있음!

- In Neural Network, to compute the loss L, Forward pass layer

$f_K(x_{K-1}) = \sigma_K(A_{K-1} x_{K-1} + b_{K-1})$

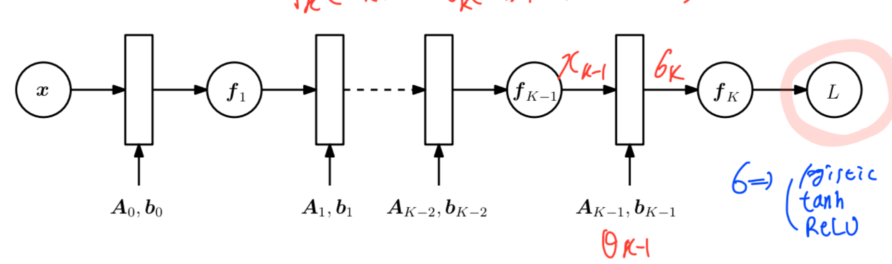

Figure 5.8 Forward pass in a multi-layer neural network to compute the loss L as a function of the inputs $\boldsymbol{x}$ and the parameters $\boldsymbol{A}_i$, $\boldsymbol{b}_i$.

- to compute the gradient of the loss functions, Backward pass layer

es as

⟨reverse mode 이기 때문에⟩

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

ⓐ는 안다면, ⓑ 계산시 추가로 계산하는 부분

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

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

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

Backpropagation 하면

K-1 layer 의 미분 결과가

K-2 layer(이전 단계) 의 미분에 재활용됨

· 사실 Back propagation은 automatic differentiation의 한 종류이다.

* automatic differentiation은 numerical analysis technique

automatic differentiation
- forward mode
- reverse mode = backpropagation

can be v
matrices

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \left(\frac{\mathrm{d}y}{\mathrm{d}b}\frac{\mathrm{d}b}{\mathrm{d}a}\right)\frac{\mathrm{d}a}{\mathrm{d}x}, \quad (5.120)$$

reverse mode

데이터 흐름과 reverse

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{\mathrm{d}y}{\mathrm{d}b}\left(\frac{\mathrm{d}b}{\mathrm{d}a}\frac{\mathrm{d}a}{\mathrm{d}x}\right). \quad (5.121)$$

forward mode

* 데이터의 흐름 $x \to a, b \to y$

reverse mode

intermediate variable

· 왜 forward mode가 아닌 reverse mode인가?

⇒ input 차원이 label의 차원보다 큰 neural network에선 reverse mode가 훨씬 효율적으로 계산!

5.7 Higher-Order Derivatives

- Gradients ⇒ collection of the first-order derivatives
- Hessian ⇒ 〃 second-order derivatives
  - ↳ Newton's method optimization 등에 활용.

- Hessian Matrix ? Tensor ?

$f : \mathbb{R}^n \to \mathbb{R}$

$\Rightarrow H$ matrix $(n \times n)$

$f : \mathbb{R}^n \to \mathbb{R}^m$

$\Rightarrow H$ tensor $[(n \times n) \times m]$

Remark (Hessian of a Vector Field). If $f : \mathbb{R}^n \to \mathbb{R}^m$ is a vector field, the Hessian is an $(m \times n \times n)$-tensor. ◊

$\frac{\partial^2 f_2}{\partial x \partial y}$

$\frac{\partial^2 f_1}{\partial x \partial y}$

↳ $n \times n$

↳ m개

$\therefore$ $m \times (n \times n)$ tensor

헤시안 Tensor