Lecture 5: Vector Calculus
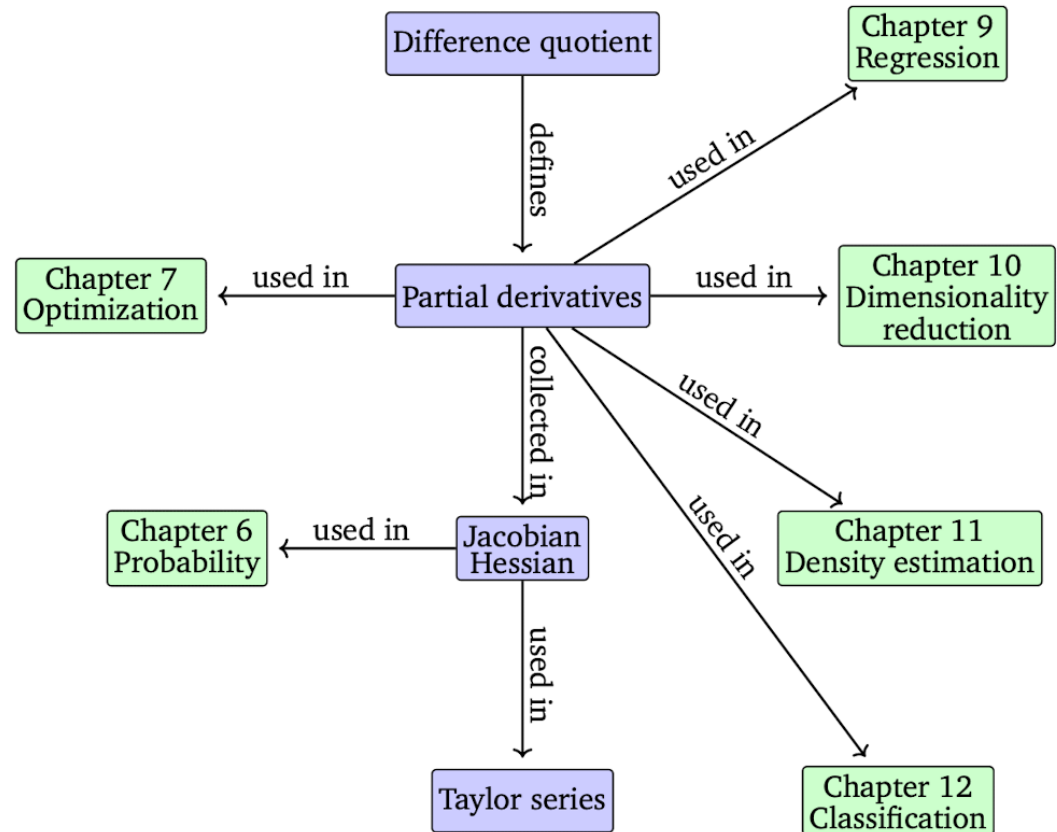
CSE4130: 기초머신러닝

Junsuk Choe (최준석)

Roadmap

(1) Differentiation of Univariate Functions

(2) Partial Differentiation and Gradients

(3) Gradients of Vector-Valued Functions

(4) Gradients of Matrices

(5) Useful Identities for Computing Gradients

(6) Backpropagation and Automatic Differentiation

(7) Higher-Order Derivatives

(8) Linearization and Multivariate Taylor Series

Summary

- Machine learning is about solving an optimization problem whose variables are the parameters of a given model.

- Solving optimization problems require gradient information.

- Central to this chapter is the concept of the function, which we often write

$$f : \mathbb{R}^D \mapsto \mathbb{R}$$
$$\boldsymbol{x} \mapsto f(\boldsymbol{x})$$

Summary

Roadmap

(1) Differentiation of Univariate Functions

(2) Partial Differentiation and Gradients

(3) Gradients of Vector-Valued Functions

(4) Gradients of Matrices

(5) Useful Identities for Computing Gradients

(6) Backpropagation and Automatic Differentiation

(7) Higher-Order Derivatives

(8) Linearization and Multivariate Taylor Series

Difference Quotient and Derivative

- **Difference Quotient.** The average slope of f between x and $x + \partial x$

$$\frac{\partial y}{\partial x} := \frac{f(x + \partial x) - f(x)}{\partial x}$$

- **Derivative.** Pointing in the direction of steepest ascent of f.

$$\frac{\mathrm{d}f}{\mathrm{d}x} := \lim_{h \to 0} \frac{f(x + h) - f(x)}{h}$$

- Unless confusion arises, we often use $f' = \frac{\mathrm{d}f}{\mathrm{d}x}$.

Example 5.2 (Derivative of a Polynomial)

Taylor Series

- Representation of a function as an infinite sum of terms, using derivatives of evaluated at x_0.

- **Taylor polynomial.** The Taylor polynomial of degree n of $f : \mathbb{R} \mapsto \mathbb{R}$ at x_0 is:

$$T_n(x) := \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k, \text{ where } f^{(k)}(x_0) \text{ is the } k\text{th derivative of } f \text{ at } x_0.$$

- **Taylor Series.** For a smooth function $f \in \mathcal{C}^\infty$, the Taylor series of f at x_0 is:

$$T_\infty(x) := \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k.$$

- If $f(x) = T_\infty(x)$, f is called analytic.

Example 5.4 (Taylor Series)

Differentiation Rules

- Product rule. $(f(x)g(x))' = f'(x)g(x) + f(x)g'(x)$

- Quotient rule. $\left(\dfrac{f(x)}{g(x)}\right)' = \dfrac{f'(x)g(x) - f(x)g'(x)}{(g(x))^2}$

- Sum rule. $(f(x) + g(x))' = f'(x) + g'(x)$

- Chain rule. $(g(f(x)))' = g'(f(x))f'(x)$

Roadmap

Gradient

- Now, $f : \mathbb{R}^n \mapsto \mathbb{R}$.

- Gradient of f w.r.t. $\boldsymbol{x}$ $\nabla_{\boldsymbol{x}} f$: Varying one variable at a time and keeping the others constant.

Partial Derivative. For $f : \mathbb{R}^n \mapsto \mathbb{R}$,

$$\frac{\partial f}{\partial x_1} = \lim_{h \to 0} \frac{f(x_1 + h, x_2, \ldots, x_n) - f(\boldsymbol{x})}{h}$$

$$\vdots$$

$$\frac{\partial f}{\partial x_n} = \lim_{h \to 0} \frac{f(x_1, x_2, \ldots, x_n + h) - f(\boldsymbol{x})}{h}$$

Gradient. Get the partial derivatives and collect them in the row vector.

$$\nabla_{\boldsymbol{x}} f = \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \left(\frac{\partial f(\boldsymbol{x})}{\partial x_1} \quad \cdots \quad \frac{\partial f(\boldsymbol{x})}{\partial x_n} \right) \in \mathbb{R}^{1 \times n}$$

Examples

- Example. $f(x, y) = (x + 2y^3)^2$

$$\frac{\partial f(x, y)}{\partial x} = 2(x + 2y^3)\frac{\partial x + 2y^3}{\partial x} = 2(x + 2y^3)$$

$$\frac{\partial f(x, y)}{\partial y} = 2(x + 2y^3)\frac{\partial x + 2y^3}{\partial y} = 12(x + 2y^3)y^2$$

- Example. $f(x_1, x_2) = x_1^2 x_2 + x_1 x_2^3$

$$\nabla_{(x_1, x_2)} f = \frac{\mathrm{d}f}{\mathrm{d}x} = \left(\frac{\partial f(x_1, x_2)}{\partial x_1} \quad \frac{\partial f(x_1, x_2)}{\partial x_2} \right) = \left(2x_1 x_2 + x_2^3 \quad x_1^2 + 3x_1 x_2^2 \right)$$

Rules for Partial Differentiation

- Product rule

$$\frac{\partial}{\partial \boldsymbol{x}}\big(f(\boldsymbol{x})g(\boldsymbol{x})\big) = \frac{\partial f}{\partial \boldsymbol{x}}g(\boldsymbol{x}) + f(\boldsymbol{x})\frac{\partial g}{\partial \boldsymbol{x}}$$

- Sum rule

$$\frac{\partial}{\partial \boldsymbol{x}}\big(f(\boldsymbol{x}) + g(\boldsymbol{x})\big) = \frac{\partial f}{\partial \boldsymbol{x}} + \frac{\partial g}{\partial \boldsymbol{x}}$$

- Chain rule

$$\frac{\partial}{\partial \boldsymbol{x}}g\big(f(\boldsymbol{x})\big) = \frac{\partial g}{\partial f}\frac{\partial f}{\partial \boldsymbol{x}}$$

More about Chain Rule

- $f : \mathbb{R}^2 \mapsto \mathbb{R}$ of two variables x_1 and x_2. $x_1(t)$ and $x_2(t)$ are functions of t.

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \begin{pmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{pmatrix} \begin{pmatrix} \frac{\partial x_1(t)}{\partial t} \\ \frac{\partial x_2(t)}{\partial t} \end{pmatrix} = \frac{\partial f}{\partial x_1} \frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2} \frac{\partial x_2}{\partial t}$$

- Example. $f(x_1, x_2) = x_1^2 + 2x_2$, where $x_1(t) = \sin(t), \ x_2(t) = \cos(t)$

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial x_1} \frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_2} \frac{\partial x_2}{\partial t} = 2\sin(t)\cos(t) - 2\sin t = 2\sin(t)(\cos(t) - 1)$$

- $f : \mathbb{R}^2 \mapsto \mathbb{R}$ of two variables x_1 and x_2. $x_1(s, t)$ and $x_2(s, t)$ are functions of s, t.

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

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

$$\frac{\mathrm{d}f}{\mathrm{d}(s, t)} = \frac{\partial f}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial(s, t)} = \begin{pmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{pmatrix} \begin{pmatrix} \frac{\partial x_1}{\partial s} & \frac{\partial x_1}{\partial t} \\ \frac{\partial x_2}{\partial s} & \frac{\partial x_2}{\partial t} \end{pmatrix}$$

Roadmap

(1) Differentiation of Univariate Functions

(2) Partial Differentiation and Gradients

(3) Gradients of Vector-Valued Functions

(4) Gradients of Matrices

(5) Useful Identities for Computing Gradients

(6) Backpropagation and Automatic Differentiation

(7) Higher-Order Derivatives

(8) Linearization and Multivariate Taylor Series

- For a function $f : \mathbb{R}^n \mapsto \mathbb{R}^m$ and vector $x = \begin{pmatrix} x_1 & \ldots & x_n \end{pmatrix}^{\mathsf{T}} \in \mathbb{R}^n$, the vector-valued function is:

$$f(x) = \begin{pmatrix} f_1(x) \\ \vdots \\ f_m(x) \end{pmatrix}$$

- Partial derivative w.r.t. x_i is a column vector: $\dfrac{\partial f}{\partial x_i} = \begin{pmatrix} \frac{\partial f_1}{\partial x_i} \\ \vdots \\ \frac{\partial f_m}{\partial x_i} \end{pmatrix}$

- Gradient (or Jacobian): $\dfrac{\mathrm{d}f(x)}{\mathrm{d}x} = \begin{pmatrix} \frac{\partial f(x)}{\partial x_1} & \ldots & \frac{\partial f(x)}{\partial x_n} \end{pmatrix}$

Jacobian

$$J = \nabla_x f = \frac{df(x)}{dx} = \left(\frac{\partial f(x)}{\partial x_1} \quad \cdots \quad \frac{\partial f(x)}{\partial x_n} \right)$$

$$= \begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1} & \cdots & \frac{\partial f_1(x)}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_m(x)}{\partial x_1} & \cdots & \frac{\partial f_m(x)}{\partial x_n} \end{pmatrix}$$

- For a $\mathbb{R}^n \mapsto \mathbb{R}^m$ function, its Jacobian is a $m \times n$ matrix.

Example 5.9 (Gradient of Vector-Valued Function)

- $f(x) = Ax$, $f : \mathbb{R}^n \mapsto \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$

- Partial derivatives: $f_i(x) = \sum_{j=1}^{n} A_{ij} x_j \implies \dfrac{\partial f_i}{\partial x_j} = A_{ij}$

- Graident

$$\frac{\mathrm{d}f}{\mathrm{d}x} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix} = \begin{pmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & & \vdots \\ A_{m1} & \cdots & A_{mn} \end{pmatrix} = A$$

Example 5.10 (Chain Rule)

- $h : \mathbb{R} \mapsto \mathbb{R}$, $h(t) = (f \circ g)(t)$ with

$$f : \mathbb{R}^2 \mapsto \mathbb{R}, \ f(\boldsymbol{x}) = \exp(x_1 x_2^2), \quad g : \mathbb{R} \mapsto \mathbb{R}^2, \ \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = g(t) = \begin{pmatrix} t \cos(t) \\ t \sin(t) \end{pmatrix}$$

- (Note) $\frac{\partial f}{\partial \boldsymbol{x}} \in \mathbb{R}^{1 \times 2}$ and $\frac{\partial g}{\partial t} \in \mathbb{R}^{2 \times 1}$

- Using the chain rule,

$$\frac{\mathrm{d}h}{\mathrm{d}t} = \frac{\partial f}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}}{\partial t} = \begin{pmatrix} \frac{\partial f}{\partial x_1} & \frac{\partial f}{\partial x_2} \end{pmatrix} \begin{pmatrix} \frac{\partial x_1}{\partial t} \\ \frac{\partial x_2}{\partial t} \end{pmatrix}$$

$$= \begin{pmatrix} \exp(x_1 x_2^2) x_2^2 & 2 \exp(x_1 x_2^2) x_1 x_2 \end{pmatrix} \begin{pmatrix} \cos(t) - t \sin(t) \\ \sin(t) + t \cos(t) \end{pmatrix}$$

Example 5.11 (Least-Square Loss (1))

- A linear model: $\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{\theta}$

- $\boldsymbol{\theta} \in \mathbb{R}^D$: parameter vector

- $\boldsymbol{\Phi} \in \mathbb{R}^{N \times D}$: input features

- $\boldsymbol{y} \in \mathbb{R}^N$: observations

- Goal: Find a good parameter vector that provides the best-fit, formulated by minimizing the following loss $L : \mathbb{R}^D \mapsto \mathbb{R}$ over the parameter vector $\boldsymbol{\theta}$.

$$L(\boldsymbol{e}) := \|\boldsymbol{e}\|^2, \quad \text{where } \boldsymbol{e}(\boldsymbol{\theta}) = \boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{\theta}$$

Example 5.11 (Least-Square Loss (2))

- $\dfrac{\partial L}{\partial \boldsymbol{\theta}} = \dfrac{\partial L}{\partial \boldsymbol{e}} \dfrac{\partial \boldsymbol{e}}{\partial \boldsymbol{\theta}}$

- Note. $\dfrac{\partial L}{\partial \boldsymbol{\theta}} \in \mathbb{R}^{1 \times D}, \ \dfrac{\partial L}{\partial \boldsymbol{e}} \in \mathbb{R}^{1 \times N}, \ \dfrac{\partial \boldsymbol{e}}{\partial \boldsymbol{\theta}} \in \mathbb{R}^{N \times D}$

- Using that $\|\boldsymbol{e}\|^2 = \boldsymbol{e}^\mathsf{T}\boldsymbol{e}$, $\dfrac{\partial L}{\partial \boldsymbol{e}} = 2\boldsymbol{e}^\mathsf{T} \in \mathbb{R}^{1 \times N}$ and $\dfrac{\partial \boldsymbol{e}}{\partial \boldsymbol{\theta}} = -\boldsymbol{\Phi} \in \mathbb{R}^{N \times D}$
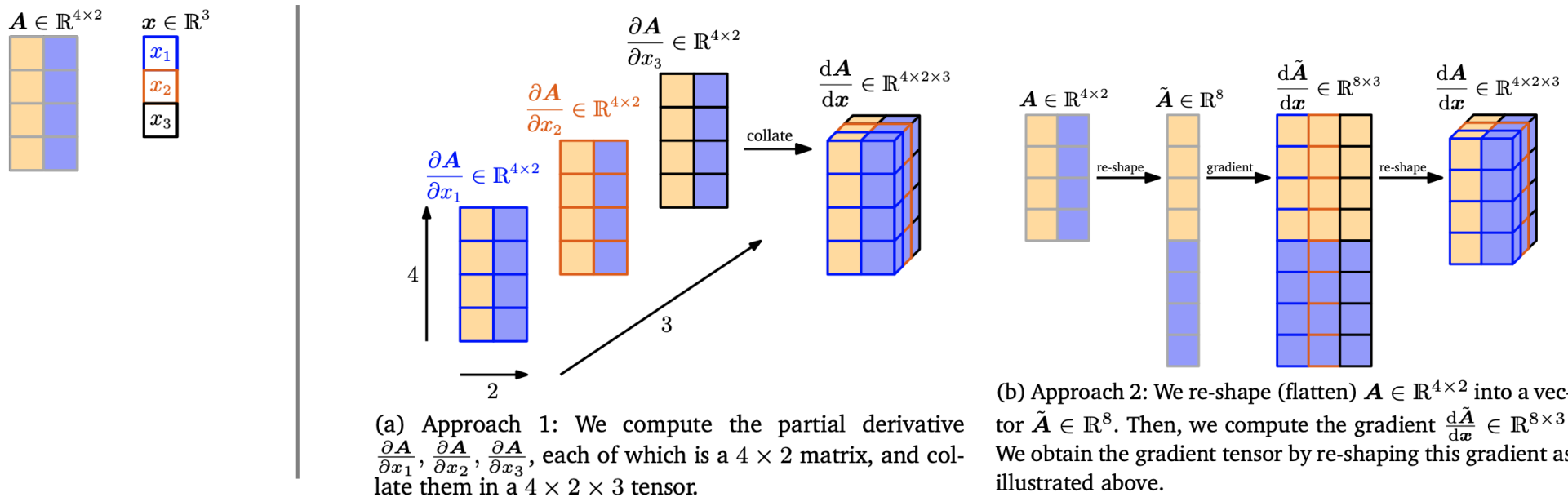
  Finally, we get: $\quad \dfrac{\partial L}{\partial \boldsymbol{\theta}} = 2\boldsymbol{e}^\mathsf{T}(-\boldsymbol{\Phi}) = -\underbrace{2(\boldsymbol{y}^\mathsf{T} - \boldsymbol{\theta}^\mathsf{T}\boldsymbol{\Phi}^\mathsf{T})}_{1 \times N}\underbrace{\boldsymbol{\Phi}}_{N \times D}$

Roadmap

(1) Differentiation of Univariate Functions

(2) Partial Differentiation and Gradients

(3) Gradients of Vector-Valued Functions

(4) Gradients of Matrices

(5) Useful Identities for Computing Gradients

(6) Backpropagation and Automatic Differentiation

(7) Higher-Order Derivatives

(8) Linearization and Multivariate Taylor Series

Gradients of matrices

- Gradient of matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ w.r.t. matrix $\boldsymbol{B} \in \mathbb{R}^{p \times q}$

- Jacobian: A four-dimensional tensor[1] $\boldsymbol{J} = \dfrac{\mathrm{d}\boldsymbol{A}}{\mathrm{d}\boldsymbol{B}} \in \mathbb{R}^{(m \times n) \times (p \times q)}$

(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.

(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.

---

[1]A multidimensional array

Example: Gradient of Vectors for Matrices (1)

- $f(x) = Ax$, $f \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$. What is $\frac{\mathrm{d}f}{\mathrm{d}A}$?

- Dimension: If we consider $f : \mathbb{R}^{m \times n} \mapsto \mathbb{R}^m$, $\frac{\mathrm{d}f}{\mathrm{d}A} \in \mathbb{R}^{m \times (m \times n)}$

- Partial derivatives: $\frac{\partial f_i}{\partial A} \in \mathbb{R}^{1 \times (m \times n)}$, $\quad \frac{\mathrm{d}f}{\mathrm{d}A} = \begin{pmatrix} \frac{\partial f_1}{\partial A} \\ \vdots \\ \frac{\partial f_m}{\partial A} \end{pmatrix}$

$$f_i = \sum_{j=1}^n A_{ij} x_j, \ i = 1, \ldots, m \implies \frac{\partial f_i}{\partial A_{iq}} = x_q,$$

$\dfrac{\partial f_i}{\partial A_{i\cdot}} = x^\mathsf{T} \in \mathbb{R}^{1 \times 1 \times n}$ (for ith row vector)

$\dfrac{\partial f_i}{\partial A_{k \neq i\cdot}} = \mathbf{0}^\mathsf{T} \in \mathbb{R}^{1 \times 1 \times n}$ (for kth row vector, $k \neq i$)

$$\frac{\partial f_i}{\partial A} = \begin{pmatrix} \mathbf{0}^\mathsf{T} \\ \vdots \\ \mathbf{0}^\mathsf{T} \\ x^\mathsf{T} \\ \mathbf{0}^\mathsf{T} \\ \vdots \\ \mathbf{0}^\mathsf{T} \end{pmatrix} \in \mathbb{R}^{1 \times (m \times n)}$$

- $R \in \mathbb{R}^{m \times n}$ and $f : \mathbb{R}^{m \times n} \mapsto \mathbb{R}^{n \times n}$ with $f(R) = K := R^\mathsf{T} R \in \mathbb{R}^{n \times n}$. What is $\frac{\mathrm{d}K}{\mathrm{d}R} \in \mathbb{R}^{(n \times n) \times (m \times n)}$?

- $\frac{\mathrm{d}K_{pq}}{\mathrm{d}R} \in \mathbb{R}^{1 \times m \times n}$. Let r_i be the ith column of R. Then $K_{pq} = r_p{}^\mathsf{T} r_q = \sum_{k=1}^{m} R_{kp} R_{kq}$.

- Partial derivative $\frac{\partial K_{pq}}{\partial R_{ij}}$

$$\frac{\partial K_{pq}}{\partial R_{ij}} = \sum_{k=1}^{m} \frac{\partial}{\partial R_{ij}} R_{kp} R_{kq} = \partial_{pqij}, \quad \partial_{pqij} = \begin{cases} R_{iq} & \text{if } j = p, p \neq q \\ R_{ip} & \text{if } j = q, p \neq q \\ 2R_{iq} & \text{if } j = p, p = q \\ 0 & \text{otherwise} \end{cases}$$

Useful Identities

$$\frac{\partial}{\partial \boldsymbol{X}} \boldsymbol{f}(\boldsymbol{X})^\top = \left(\frac{\partial \boldsymbol{f}(\boldsymbol{X})}{\partial \boldsymbol{X}}\right)^\top \tag{5.99}$$

$$\frac{\partial}{\partial \boldsymbol{X}} \operatorname{tr}(\boldsymbol{f}(\boldsymbol{X})) = \operatorname{tr}\left(\frac{\partial \boldsymbol{f}(\boldsymbol{X})}{\partial \boldsymbol{X}}\right) \tag{5.100}$$

$$\frac{\partial}{\partial \boldsymbol{X}} \det(\boldsymbol{f}(\boldsymbol{X})) = \det(\boldsymbol{f}(\boldsymbol{X}))\operatorname{tr}\left(\boldsymbol{f}(\boldsymbol{X})^{-1}\frac{\partial \boldsymbol{f}(\boldsymbol{X})}{\partial \boldsymbol{X}}\right) \tag{5.101}$$

$$\frac{\partial}{\partial \boldsymbol{X}} \boldsymbol{f}(\boldsymbol{X})^{-1} = -\boldsymbol{f}(\boldsymbol{X})^{-1}\frac{\partial \boldsymbol{f}(\boldsymbol{X})}{\partial \boldsymbol{X}}\boldsymbol{f}(\boldsymbol{X})^{-1} \tag{5.102}$$

$$\frac{\partial \boldsymbol{a}^\top \boldsymbol{X}^{-1}\boldsymbol{b}}{\partial \boldsymbol{X}} = -(\boldsymbol{X}^{-1})^\top \boldsymbol{a}\boldsymbol{b}^\top(\boldsymbol{X}^{-1})^\top \tag{5.103}$$

$$\frac{\partial \boldsymbol{x}^\top \boldsymbol{a}}{\partial \boldsymbol{x}} = \boldsymbol{a}^\top \tag{5.104}$$

$$\frac{\partial \boldsymbol{a}^\top \boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{a}^\top \tag{5.105}$$

$$\frac{\partial \boldsymbol{a}^\top \boldsymbol{X}\boldsymbol{b}}{\partial \boldsymbol{X}} = \boldsymbol{a}\boldsymbol{b}^\top \tag{5.106}$$

$$\frac{\partial \boldsymbol{x}^\top \boldsymbol{B}\boldsymbol{x}}{\partial \boldsymbol{x}} = \boldsymbol{x}^\top(\boldsymbol{B} + \boldsymbol{B}^\top) \tag{5.107}$$

$$\frac{\partial}{\partial \boldsymbol{s}}(\boldsymbol{x} - \boldsymbol{A}\boldsymbol{s})^\top \boldsymbol{W}(\boldsymbol{x} - \boldsymbol{A}\boldsymbol{s}) = -2(\boldsymbol{x} - \boldsymbol{A}\boldsymbol{s})^\top \boldsymbol{W}\boldsymbol{A} \quad \text{for symmetric } \boldsymbol{W} \tag{5.108}$$

Roadmap

Motivation: Neural Networks with Many Layers (1)

- In a neural network with many layers, the function y is a many-level function compositions

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

where, for example,
  - x: images as inputs, y: class labels (e.g., cat or dog) as outputs
  - each f_i has its own parameters
- In neural networks, with the model parameters $\theta = \{A_0, b_0, \ldots, A_{K-1}, b_{K-1}\}$

$$\begin{cases} f_0 & := x \\ f_1 & := \sigma_1(A_0 f_0 + b_0) \\ \vdots \\ f_K & := \sigma_K(A_{K-1} f_{K-1} + b_{K-1}) \end{cases}$$

  - Minimizing the loss function over θ:

$$\min_{\theta} L(\theta),$$

where $L(\theta) = \|y - f_K(\theta, x)\|^2$

  - σ_i is called the activation function at i-th layer

Motivation: Neural Networks with Many Layers (2)

- In neural networks, with the model parameters $\theta = \{A_0, b_0, \ldots, A_{K-1}, b_{K-1}\}$

$$\begin{cases} f_0 & := x \\ f_1 & := \sigma_1(A_0 f_0 + b_0) \\ \vdots \\ f_K & := \sigma_K(A_{K-1} f_{K-1} + b_{K-1}) \end{cases}$$

○ σ_i is called the activation function at i-th layer
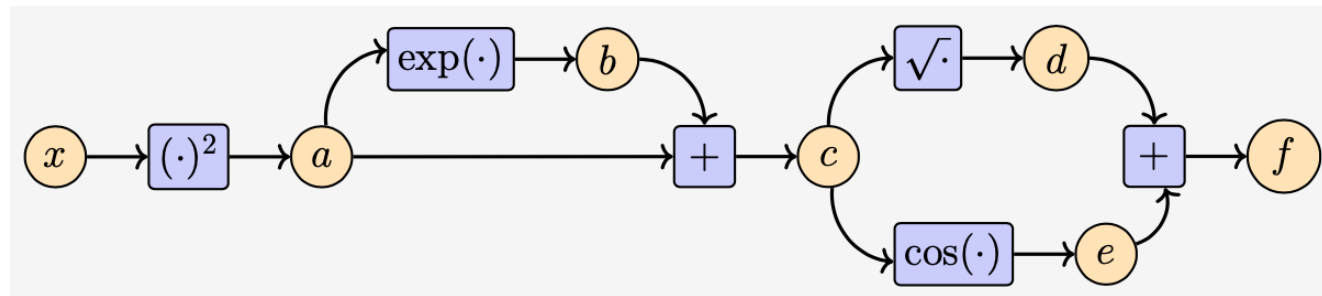
○ Minimizing the loss function over θ:

$$\min_{\theta} L(\theta),$$

where $L(\theta) = \|y - f_K(\theta, x)\|^2$

- Question. How can we efficiently compute $\dfrac{\mathrm{d}L}{\mathrm{d}\theta}$ in computers?

Backpropagatin: Example (1)

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

- Computation graph: Connect via "elementary" operations

$$a = x^2, \; b = \exp(a), \; c = a + b, \; d = \sqrt{c}, \; e = \cos(c), \; f = d + e$$

- Automatic Differentiation
  - A set of techniques to numerically (not symbolically) evaluate the gradient of a function by working with intermediate variables and applying the chain rule.

Backpropagation: Example (2)

- $a = x^2, \; b = \exp(a), \; c = a + b, \; d = \sqrt{c}, \; e = \cos(c), \; f = d + e$

- Derivatives of the intermediate variables with their inputs

$$\frac{\partial a}{\partial x} = 2x, \; \frac{\partial b}{\partial a} = \exp(a), \; \frac{\partial c}{\partial a} = 1 = \frac{\partial c}{\partial b}, \; \frac{\partial d}{\partial c} = \frac{1}{2\sqrt{c}}, \; \frac{\partial e}{\partial c} = -\sin(c), \; \frac{\partial f}{\partial d} = 1 = \frac{\partial f}{\partial e}$$

- Compute $\dfrac{\partial f}{\partial x}$ by working backward from the output

$$\frac{\partial f}{\partial c} = \frac{\partial f}{\partial d}\frac{\partial d}{\partial c} + \frac{\partial f}{\partial e}\frac{\partial e}{\partial c}, \; \frac{\partial f}{\partial b} = \frac{\partial f}{\partial c}\frac{\partial c}{\partial b}$$

$$\frac{\partial f}{\partial a} = \frac{\partial f}{\partial b}\frac{\partial b}{\partial a} + \frac{\partial f}{\partial c}\frac{\partial c}{\partial a}, \; \boxed{\frac{\partial f}{\partial x}} = \frac{\partial f}{\partial a}\frac{\partial a}{\partial x}$$

$$\frac{\partial f}{\partial c} = 1 \cdot \frac{1}{2\sqrt{c}} + 1 \cdot (-\sin(c))$$

$$\frac{\partial f}{\partial b} = \frac{\partial f}{\partial c} \cdot 1, \quad \frac{\partial f}{\partial a} = \frac{\partial f}{\partial b}\exp(a) + \frac{\partial f}{\partial c} \cdot 1$$

$$\boxed{\frac{\partial f}{\partial x}} = \frac{\partial f}{\partial a} \cdot 2x$$

Backpropagation

- Implementation of gradients can be very expensive, unless we are careful.

- Using the idea of automatic differentiation, the whole gradient computation is decomposed into a set of gradients of elementary functions and application of the chain rule.

- Why backward?
  - In neural networks, the input dimensionality is often much higher than the dimensionality of labels.

  - In this case, the backward computation (than the forward computation) is much cheaper.

- Works if the target is expressed as a computation graph whose elementary functions are differentiable. If not, some care needs to be taken.

Roadmap

Higher-Order Derivatives

- Some optimization algorithms (e.g., Newton's method) require second-order derivatives, if they exist.

- (Truncated) Taylor series is often used as an approximation of a function.

- For $f : \mathbb{R}^n \mapsto \mathbb{R}$ of variable $\boldsymbol{x} \in \mathbb{R}^n$, $\nabla_{\boldsymbol{x}} f = \frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \left(\frac{\partial f(\boldsymbol{x})}{\partial x_1} \quad \cdots \quad \frac{\partial f(\boldsymbol{x})}{\partial x_n} \right) \in \mathbb{R}^{1 \times n}$
  - If f is twice-differentiable, the order doesn't matter.

$$
H_{\boldsymbol{x}} f = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & & & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n} & \frac{\partial^2 f}{\partial x_2 \partial x_n} & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_n} \end{pmatrix}
$$

- For $f : \mathbb{R}^n \mapsto \mathbb{R}^m$, $\nabla_{\boldsymbol{x}} f \in \mathbb{R}^{m \times n}$
  - Thus, $H_{\boldsymbol{x}} f \in \mathbb{R}^{m \times n \times n}$ (a tensor)

Function Approximation: Linearization and More

- First-order approximation of $f(x)$ (i.e., linearization by taking the first two terms of Taylor Series)

$$f(x) \approx f(x_0) + (\nabla_x f)(x_0)(x - x_0)$$

- Multivariate Talyer Series for $f : \mathbb{R}^D \mapsto \mathbb{R}$ at x_0

$$f(x) = \sum_{k=0}^{\infty} \frac{D_x^k f(x_0)}{k!} \delta^k,$$

where $D_x^k f(x_0)$ is the kth derivative of f w.r.t. x, evaluated at x_0, and $\delta := x - x_0$.

  ◦ Partial sum up to, say n, can be an approximation of $f(x)$.

  ◦ $D_x^k f(x_0)$ and δ^k are kth order tensors, i.e., k-dimensional array.

  ◦ δ^k is a k-fold outer product $\otimes$. For example, $\delta^2 = \delta \otimes \delta = \delta \delta^\mathsf{T}$. $\delta^3 = \delta \otimes \delta \otimes \delta$.

Questions?

References

[1] This lecture slide is mainly based upon
`https://yung-web.github.io/home/courses/mathml.html` (made by Prof. Yung Yi, KAIST EE)

[2] Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong. Mathematics for machine learning. Cambridge University Press, 2020.