5. Vector Calculus

최적화 문제를 풀기 위해서는 Gradient Descent 방법으로 최적의 해를 찾음

머신러닝 알고리즘에서 최적의 parameter 값 찾는 문제

최적의 값을 찾아가는 과정을 learning, 필요한 정보가 gradient

Discuss how to compute gradients(소위 기울기. 미분을 통해 얻어짐) of functions

5.1 Differentiation of Univariate Functions

5.1.1 Taylor Series

Representation of a function f as an infinite sum of terms

5.1.2 Differentiation Rules

5.2 Partial Differentiation and Gradients

지금까지 univariate variable 에 대한 미분. 이제는 multivariate variable 에 대한 미분.

Gradient 를 구하는 방법은 여러 개의 variable 중 하나씩에 대한 미분을 진행하고 모두 모아주면 됨. 이 모아진 것을 gradient 라고 함.

특정 변수에 대한 gradient 를 partial derivative 라고 함.

Multivariate variables 에 대한 partial derivatives

Example 5.6 (Partial Derivatives Using the Chain Rule)
For $f(x, y) = (x + 2y^3)^2$, we obtain the partial derivatives

$$\frac{\partial f(x,y)}{\partial x} = 2(x + 2y^3)\frac{\partial}{\partial x}(x + 2y^3) = 2(x + 2y^3)\,, \tag{5.41}$$

$$\frac{\partial f(x,y)}{\partial y} = 2(x + 2y^3)\frac{\partial}{\partial y}(x + 2y^3) = 12(x + 2y^3)y^2\,. \tag{5.42}$$

where we used the chain rule (5.32) to compute the partial derivatives.

Multivariate variables 의 gradient

Partial derivatives 를 구한 후, 로우 벡터 형태로 모아주면 됨

Example 5.7 (Gradient)
For $f(x_1, x_2) = x_1^2 x_2 + x_1 x_2^3 \in \mathbb{R}$, the partial derivatives (i.e., the derivatives of f with respect to x_1 and x_2) are

$$\frac{\partial f(x_1, x_2)}{\partial x_1} = 2x_1 x_2 + x_2^3 \quad (5.43)$$

$$\frac{\partial f(x_1, x_2)}{\partial x_2} = x_1^2 + 3x_1 x_2^2 \quad (5.44)$$

and the gradient is then

$$\frac{\mathrm{d}f}{\mathrm{d}\boldsymbol{x}} = \begin{bmatrix} \frac{\partial f(x_1, x_2)}{\partial x_1} & \frac{\partial f(x_1, x_2)}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2x_1 x_2 + x_2^3 & x_1^2 + 3x_1 x_2^2 \end{bmatrix} \in \mathbb{R}^{1 \times 2}. \quad (5.45)$$

5.2.1 Basic Rules of Partial Differentiation

5.2.2. Chain Rule

5.3 gradients of Vector-Valued Functions

R^n 을 real value 가 아닌 R^m 으로 매핑하는 함수를 미분함

즉, 미분 값이 vector 형태로 나오는 함수를 미분함

f 가 총 m 개 있어서 각각의 gradient 값을 뱉어냄

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

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

함수 f 의 개수만큼 열벡터로 쭉 나열해준 것.

$$\frac{\mathrm{d}\boldsymbol{f}(\boldsymbol{x})}{\mathrm{d}\boldsymbol{x}} = \left[\boxed{\frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial x_1}} \cdots \boxed{\frac{\partial \boldsymbol{f}(\boldsymbol{x})}{\partial x_n}} \right] \tag{5.56a}$$

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

네모칸이 의미하는 것은 m 개의 함수를 하나의 변수 x1 으로 미분한 값이 되고, 행렬 로우는 각 함수의 gradient 값임

Row 로 나열된 것이 함수의 아웃풋, gradient

Column 방향으로나열된 것이 인풋이 됨

Jacobian 매트릭스(자코비안 행렬)

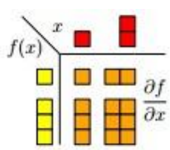
Linear mapping 으로 transform 했을 때 얼마나 scaling 되는지 나타낼 때 linear trans 에서는 가능한데 non-linear trans 일 때는 아니므로, 이 때 어떤지 알려주는 것이 Jacobian 임.

볼륨은 approximation

Partial derivatives 후 어떤 shape

1 차원 -> 1 차원 함수 미분은 real value gradient

n 차원에서 m 차원으로 매핑 함수 미분은 mxn 차원의 gradient 값



In this chapter, we encountered derivatives of functions. Figure 5.6 summarizes the dimensions of those derivatives. If $f : \mathbb{R} \to \mathbb{R}$ the gradient is simply a scalar (top-left entry). For $f : \mathbb{R}^D \to \mathbb{R}$ the gradient is a $1 \times D$ row vector (top-right entry). For $\boldsymbol{f} : \mathbb{R} \to \mathbb{R}^E$, the gradient is an $E \times 1$ column vector, and for $\boldsymbol{f} : \mathbb{R}^D \to \mathbb{R}^E$ the gradient is an $E \times D$ matrix.

벡터를 다루는 함수에 대한 gradient

F(x)를 x 에 대해 미분

Output 이 M 개이므로 로우가 M, input 이 N 개니까 칼럼이 N 개

Fi(x)값은 A 의 i 번째 element 와 xj 와의 dot product 라는 사실

최종적으로 MxN 행렬로 gradient 가 나타남

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

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

최종적으로 f 에 대해 x 를 미분: Chain rule 로 대입하여 gradient 구하기

5.4 Gradients of Matrices

행렬이 주어졌을 때 행렬을 벡터/다른 행렬로 미분한 gradient 들은 multidimensional tensor 로 계산됨. 아웃풋의 dimension 과 크기만 신경쓰면 됨.

이 gradient 를 구하는 방법

예) A(mxn) 행렬이 B(pxq) 행렬로 미분

4 dimensional tensor J

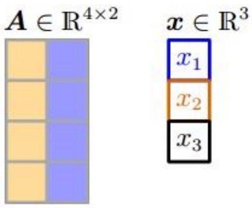
Jijkl 엔트리에는 각각 partial derivatives 값들이 들어감

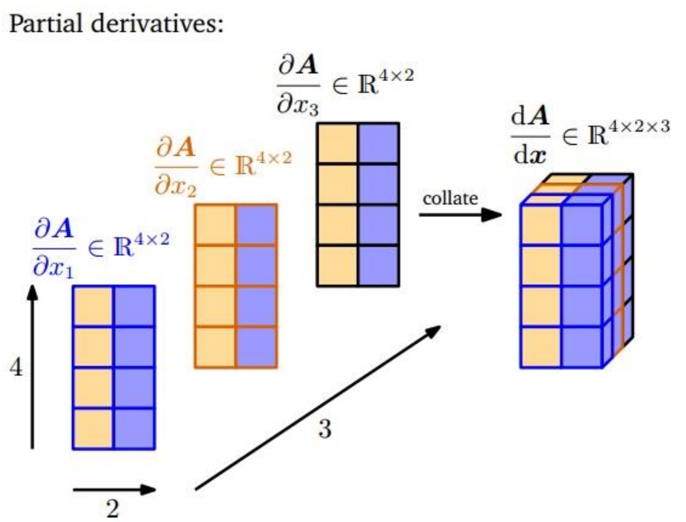
A 라는 행렬을 x 벡터로 미분

최종적으로 4x2x3

각 셀에는 A 와 x 의 1 dimension 곱으로 표현됨

A 를 x1, x2, x3 로 미분한 partial derivatives 들이 있고, 최종적으로 4x3x4 결과가 되는 개념만 파악





(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.

M dimension 의 f 를 A 행렬로 미분하면 M x (M x N) dimension 의 결과를 얻음

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

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

Partial derivatives 의 벡터들을 모아 gradient 를 이루게 됨

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

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

A 의 하나의 element 에 대해서 어떻게 표현되는지 보여줌

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

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

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

$$\frac{\partial f_i}{\partial A_{i,:}} = \boldsymbol{x}^\top \in \mathbb{R}^{1\times1\times N}, \tag{5.90}$$

행렬 A 의 element 들과 j 번째 x 벡터와의 곱으로 표현됨

특정 fi 를 Aiq 로 미분하여 xq 라 하고, 이는 하나의 행벡터로 partial derivatives 가 됨

하나의 partial derivatives

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

행렬을 행렬로 미분

아웃풋 shape: (NxN) x (MxN)의 4 dimension tensor

K 의 각 element 들은 r 의 내적으로 표현됨

Example 5.13 (Gradient of Matrices with Respect to Matrices)
Consider a matrix $\boldsymbol{R} \in \mathbb{R}^{M\times N}$ and $\boldsymbol{f} : \mathbb{R}^{M\times N} \to \mathbb{R}^{N\times N}$ with

$$\boldsymbol{f}(\boldsymbol{R}) = \boldsymbol{R}^\top \boldsymbol{R} =: \boldsymbol{K} \in \mathbb{R}^{N\times N}, \tag{5.93}$$

where we seek the gradient $\mathrm{d}\boldsymbol{K}/\mathrm{d}\boldsymbol{R}$.

To solve this hard problem, let us first write down what we already know: The gradient has the dimensions

$$\frac{\mathrm{d}\boldsymbol{K}}{\mathrm{d}\boldsymbol{R}} \in \mathbb{R}^{(N\times N)\times(M\times N)}, \tag{5.94}$$

which is a tensor. Moreover,

$$\frac{\mathrm{d}K_{pq}}{\mathrm{d}\boldsymbol{R}} \in \mathbb{R}^{1\times M\times N} \tag{5.95}$$

for $p, q = 1, \ldots, N$, where K_{pq} is the (p, q)th entry of $\boldsymbol{K} = \boldsymbol{f}(\boldsymbol{R})$. Denoting the ith column of $\boldsymbol{R}$ by $\boldsymbol{r}_i$, every entry of $\boldsymbol{K}$ is given by the dot product of two columns of $\boldsymbol{R}$, i.e.,

$$K_{pq} = \boldsymbol{r}_p^\top \boldsymbol{r}_q = \sum_{m=1}^{M} R_{mp} R_{mq}\,. \tag{5.96}$$

When we now compute the partial derivative $\frac{\partial K_{pq}}{\partial R_{ij}}$ we obtain

$$\frac{\partial K_{pq}}{\partial R_{ij}} = \sum_{m=1}^{M} \frac{\partial}{\partial R_{ij}} R_{mp} R_{mq} = \partial_{pqij}\,, \tag{5.97}$$

5.5 Useful Identities for Computing Gradients

5.5 Useful Identities for Computing Gradients

In the following, we list some useful gradients that are frequently required in a machine learning context (Petersen and Pedersen, 2012). Here, we use $\mathrm{tr}(\cdot)$ as the trace (see Definition 4.4), $\det(\cdot)$ as the determinant (see Section 4.1) and $\boldsymbol{f}(\boldsymbol{X})^{-1}$ as the inverse of $\boldsymbol{f}(\boldsymbol{X})$, assuming it exists.

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

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

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

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

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

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

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

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

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

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

5.6 Backpropagation and Automatic Differentiation

좋은 파라미터는 gradient descent 로 찾게 됨

Train 데이터를 잘 설명하는 objective function 을 정의하고 이를 minimize 하는 방법으로 parameter 를 찾는 방식

Automatic Differentiation: Gradient 를 구하는 함수가 매부 복잡할 경우, 효율적으로 계산할 수 있게 하는 알고리즘

특별한 경우 backpropagation 알고리즘

Backpropagation 은 gradient 를 계산하기 위해 활용하는 최적화 알고리즘

딥러닝 모델은 최종적인 output 이 간단한 연산들의 sequential 한 chain 으로 표현됨

많은 function 이 결합되어 있는 모델

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

X 가 f1 의 입력이 되어 output 을 내고, 이 output 이 다음의 input 이 되는 걸 반목하는 모델

L (loss)를 최소화할 수 있는 parameter 를 gradient descent 에서 찾음

Gradient 정보를 미분으로 찾고 L 을 줄일 수 있는 방향을 찾아 반대방향으로 이동

L 이 줄어들고,

또 다시 gradient 를 통ㅎ서 방향을 찾아서 loss 를 줄여나가는 방식

주황색: Output 을 함수의 input 으로 편미분

파란색: 함수의 output 을 그 함수가 지닌 parameter 들로 편미분

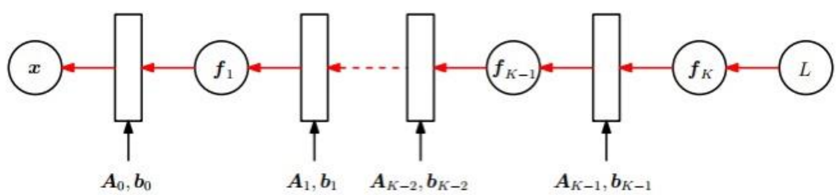
$$\frac{\partial L}{\partial \boldsymbol{\theta}_{K-1}} = \frac{\partial L}{\partial \boldsymbol{f}_K} \frac{\partial \boldsymbol{f}_K}{\partial \boldsymbol{\theta}_{K-1}} \tag{5.115}$$

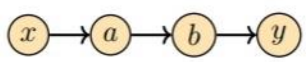
$$\frac{\partial L}{\partial \boldsymbol{\theta}_{K-2}} = \frac{\partial L}{\partial \boldsymbol{f}_K} \boxed{\frac{\partial \boldsymbol{f}_K}{\partial \boldsymbol{f}_{K-1}} \frac{\partial \boldsymbol{f}_{K-1}}{\partial \boldsymbol{\theta}_{K-2}}} \tag{5.116}$$

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

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

연산에 사용된 부분이 재사용가능하다는 것을 토대로 뒤로부터 gradient 가 propagation 된다고 하여 backpropagation 이라고 부름





간단한 함수들의 복잡한 결합을 미분할 때, 단위 함수들의 미분값을 이용한 chain rule 로 gradient 를 계산하는 것이 explicit 하게 gradient 를 구하는 것보다 간편함

Backpropagation 은 하나의 계산 알고리즘으로 gradient 를 구하는 방법 중 하나.

이 방법은 최적화를 하는 것이 목적이므로 딥러닝의 학습과정에서 최적의 파라미터를 찾는데 잘 활용될 수 있음

5.7 Higher-Order Derivatives

5.8 Linearization and Multivariate Taylor Series

Gradient 는 함수를 한 번 미분

함수 여러 번 미분한 higher-order derivatives

주어진 함수에 대한 특점 지점에서 풍부한 정보 얻고 싶을 때의 정보를 알려 줌

Gradient 는 local 영역에서의 slope 의미.

그 지점에서의 함수 형태가 얼마나 곡선을 띄고 있는지 등에 대한 정보 알 수 있음

F 를 x 에 대해 2 번 미분하거나 n 번 미분

두개의 입력 -> x 에 대해 먼저 미분 후 y 를 미분

- $\frac{\partial^2 f}{\partial x^2}$ is the second partial derivative of f with respect to x.
- $\frac{\partial^n f}{\partial x^n}$ is the nth partial derivative of f with respect to x.
- $\frac{\partial^2 f}{\partial y \partial x} = \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial x}\right)$ is the partial derivative obtained by first partial differentiating with respect to x and then with respect to y.
- $\frac{\partial^2 f}{\partial x \partial y}$ is the partial derivative obtained by first partial differentiating by y and then x.

대표적으로 second derivatives 많이 씀

Hessian matrix: 모든 second-order partial derivatives 를 모아놓은 것

$$\boldsymbol{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix} \tag{5.147}$$

Gradient 가 주는 정보: 주어진 지점에서 tangent lane 에 대한 정보 줌

H 가 주는 정보: 특정 지점에서 local 하게 얼마나 공유를 이루고 있는가에 대한 curvature 정보(2 차함수 한 번 미분하면 접선의 기울기. 한 번 더 미분하면 위로 아래로 볼록인지 여부)

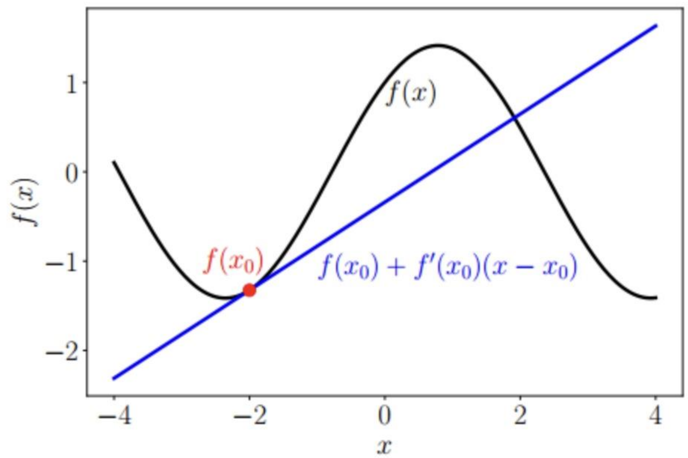
둘 다 이용: 주어진 함수의 특점 지점에서 locally approxiamation 하는 함수를 얻을 수 있음

주어진 함수의 gradient 정보는 특점 지점 x0 에서 이 함수를 locally linear approximation 하는 데 활용이 됨

그림의 검은색 실선이 fx 이고 빨간점이 x0 일 때 이 지점에서 테일러 전개, locally linear approximation 을 수행하면 파란 실선처럼 나옴

그러면 x0 지점에서 원래 함수를 근사시킬 수 있게 되는 것

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}_0) + (\nabla_{\boldsymbol{x}} f)(\boldsymbol{x}_0)(\boldsymbol{x} - \boldsymbol{x}_0)\,. \tag{5.148}$$



주어진 함수가 multivariate function 일 때 이렇게 테일러 전개 가능

$$f : \mathbb{R}^D \to \mathbb{R} \tag{5.149}$$

$$\boldsymbol{x} \mapsto f(\boldsymbol{x})\,, \quad \boldsymbol{x} \in \mathbb{R}^D\,, \tag{5.150}$$

$$f(\boldsymbol{x}) = \sum_{k=0}^{\infty} \frac{D_{\boldsymbol{x}}^k f(\boldsymbol{x}_0)}{k!} \boldsymbol{\delta}^k\,, \tag{5.151}$$

그런데 주로 second-order 까지만 전개가 됨

$$f(x) \approx f(x_0) + \Delta f(x_0)(x - x_0) + \frac{1}{2}(x - x_0)^T \Delta^2 f(x_0)(x - x_0)$$