3-4. 고계미분(Higher Order Derivatives)

고계미분은 함수의 기본 미분을 한 번 이상 반복하여 얻는 미분입니다. 1차 미분(First Derivative)은 함수의 기울기를 나타내며, 2차 미분(Second Derivative)은 1차 미분의 기울기를 나타내는 방식으로 계속해서 미분할 수 있습니다. 고계 미분은 딥러닝, 물리학, 최적화 문제 등 다양한 분야에서 중요한 역할을 합니다.

1. 고계 미분의 정의

   1차 미분

   기본 미분인 1차 미분은 함수의 기울기(변화율)를 나타냅니다. 함수 $f(x)$에 대해 1차 미분은 다음과 같습니다.

   $$f'(x) = \frac{d}{dx}f(x)$$

   2차 미분

   2차 미분은 1차 미분을 다시 미분한 것입니다. 즉, 1차 미분의 변화율을 나타냅니다. 2차 미분은 다음과 같이 표현 됩니다.

   $$f''(x) = \frac{d^2}{dx^2}f(x)$$

   2차 미분은 함수의 굽힘 또는 곡률을 설명합니다. 예를 들어, 곡선이 얼마나 오목하거나 볼록한지를 알 수 있습니다.

   n차 미분

   n차 미분은 (n-1)차 미분을 다시 미분한 것입니다. n차 미분은 다음과 같이 표현됩니다.

   $$f^{(n)}(x) = \frac{d^m}{dx^n}f(x)$$

   여기서 $f^{(n)}(x)$는 $f(x)$의 n번째 미분을 의미합니다. 예를 들어, 3차 미분은 다음과 같이 표현됩니다.

$$f^{(3)}(x) = \frac{d^3}{dx^3}f(x)$$

2. 고계 미분의 의미와 용도

(1) 1차 미분 : 기울기(Gradient)

1차 미분은 함수의 기울기를 나타내며, 이는 함수의 증가나 감소 속도를 알려줍니다. 1차 미분이 0 일때, 함수는 극점(극대 또는 극소) 또는 변곡점일 수 있습니다.

(2) 2차 미분: 곡률 (Curvature)

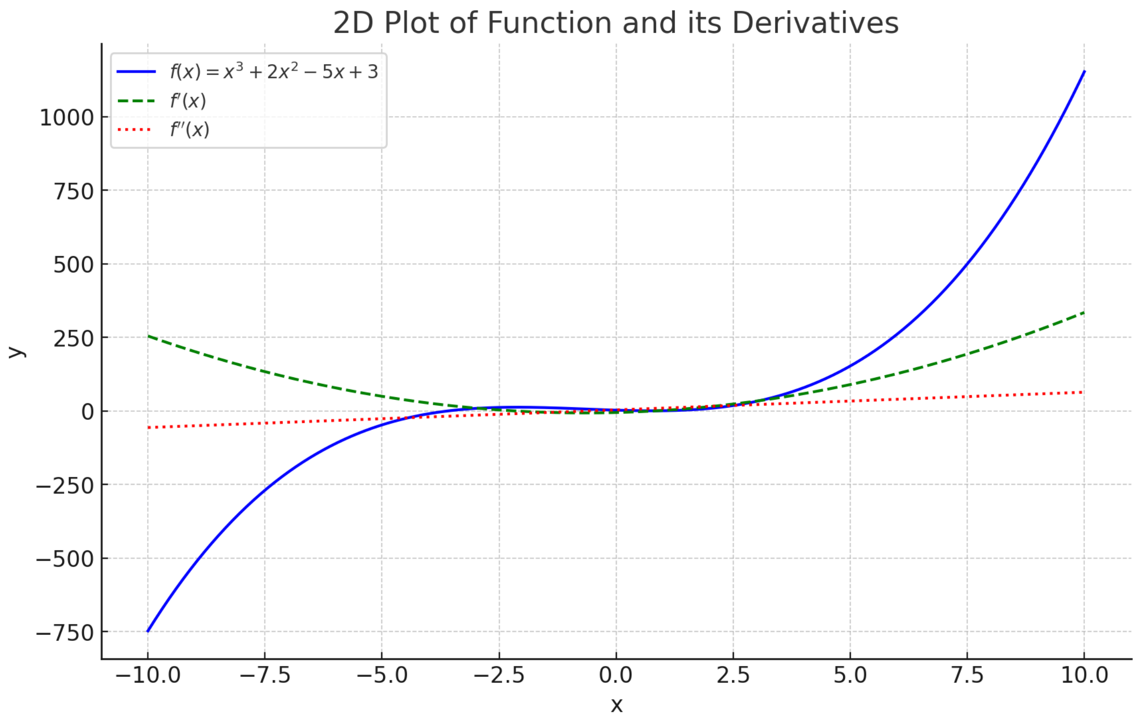
2차 미분은 함수의 굽힘 또는 곡률을 나타냅니다. 구체적으로는 함수의 볼록성과 오목성을 결정합니다.

- $f''(x) > 0$:함수가 볼록(Upward concave)하고, 그 점에서 기울기가 증가하고 있음을 의미합니다.
- $f''(x) < 0$:함수가 오목(Downward concave)하고, 그 점에서 기울기가 감소하고 있음을 의미합니다.
- $f''(x) = 0$:변곡점이 될 수 있으며, 곡률의 변화가 발생하는 지점일 수 있습니다.

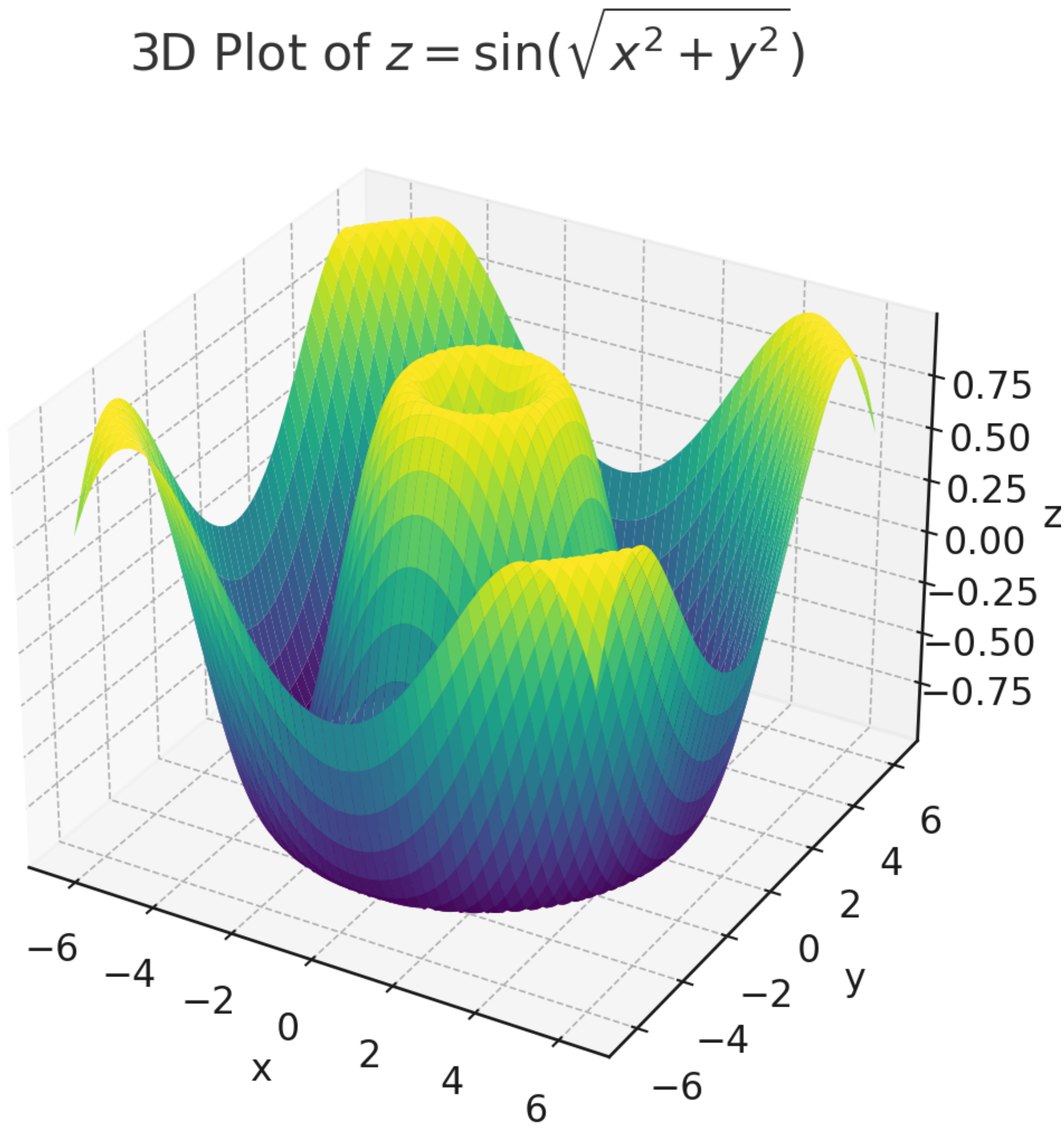
(3) n차 미분 : 더 높은 차원의 변화율

n차 미분은 함수의 더 복잡한 변화율을 설명합니다. 특히 최적화나 딥러닝에서 고계 미분은 다음과 같은 용도로 사용됩니다.

- 뉴턴법 (Newton's Method): 2차 미분을 이용하여 최적화 문제에서 더 빠르게 수렴할 수 있도록 도와줍니다.
- 헤세 행렬 (Hessian Matrix): 2차 미분으로 구성된 행렬로, 다변수 함수의 국소적인 곡률을 분석하는 데 사용됩니다. 이는 최적화에서 중요한 역할을 합니다.
- 테일러 급수 (Taylor Series): 함수의 근사 계산에 고계미분을 활용하여 높은 정확도로 함수를 근사합니다.



- 2차원 그래프에서는 다항식 함수 $f(x) = x^3 + 2x^2 - 5x + 3$와 그 1차, 2차 미분을 시각화했습니다.
  파란색은 원래 함수, 초록색 점섬은 1차 미분, 빨간색 점선은 2차 미분을 나타냅니다.



- 3차원 그래프에서는 삼각 함수의 고계 미분과 관련된 형상으로 $z = sin(\sqrt{x^2 + y^2})$ 함수를 시각화 했습니다. 이는 함수가 x와 y의 복합적인 영향을 받아 변형된 3차원 함수의 모습을 보여줍니다.

3. 고계 미분 예제

예제1 다항식의 고계미분

$$f(x) = x^3 + 2x^2 - 5x + 3$$
$$1\text{차미분} : f'(x) = 3x^2 + 4x - 5$$
$$2\text{차미분} : f''(x) + 6x + 4$$
$$3\text{차미분} : f^{(3)}(x) = 6$$

4차 미분은 0이므로, 더 이상 변화가 없습니다. 이 경우 3차 미분까지 구한 후에는 미분값이 더 이상 변하지 않습니다.

예제2 삼각 함수의 고계 미분

$$
\begin{gathered}
\text{삼각함수} f(x) = sinx \text{의고계미분} \\
1\text{차미분 : } f'(x) = cosx \\
2\text{차미분 : } f''(x) = -sinx \\
3\text{차미분 : } f^{(3)}(x) = -cosx \\
4\text{차미분 : } f^{(4)}(x) = sinx
\end{gathered}
$$

삼각 함수의 경우, 미분이 주기적입니다. 따라서 4차 미분 이후에는 미분 결과가 반복됩니다.

4. 고계 미분의 응용 : 딥러닝과 최적화

고계 미분은 딥러닝과 최적화 알고리즘에서 중요한 역할을 합니다. 다음은 그 응용 분야들입니다.

(1) 헤세 행렬(Hessian Matrix)

헤세 행렬은 2차 미분으로 이루어진 행렬로, 함수의 국소적인 곡률을 나타냅니다. 다변수 함수 $f(x_1, x_2, ..., x_n)$의 헤세 행렬은 다음과 같이 정의됩니다.

$$
\mathbf{H} = \begin{bmatrix}
\frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\
\frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2}
\end{bmatrix}
$$

헤세 행렬은 뉴턴법과 같은 최적화 알고리즘에서 중요한 역할을 하며, 2차 미분을 사용해 최적화 문제의수렴 속도를 높이는데 도움을 줍니다.

(2) 뉴턴법 (Newton's Method)

뉴턴법은 2차 미분을 이용하여 함수의 최소값을 더 빠르게 찾는 방법입니다. 1차 미분만 사용하는 경사하강법에 비해, 2차 미분 정보를 사용하여 함수의 곡률을 고려하기 때문에 더 빠르게 수렴할 수 있습니다. 뉴턴법의 업데이트 공식은 다음과 같습니다.

$$x_{new} = x - \frac{f'(x)}{f''(x)}$$

(3) 테일러 급수 (Taylor Series)

테일러 급수는 함수 $f(x)$를 특정 지점 a주변에서 근사하는 방법으로, 고계 미분을 사용하여 근사정확도를 높일 수 있습니다. 테일러 급수는 다음과 같이 표현됩니다.

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \dots$$

고차 미분이 포함된 테일러 급수를 사용하면 함수의 복잡한 형태를 더 정밀하게 근사할 수 있습니다.

```
import sympy as sp

# 변수 정의
x = sp.symbols('x')

# 함수 정의
f = x**3 + 2*x**2 - 5*x + 3

# 1차 미분
f_prime = sp.diff(f, x)
print(f"1차 미분: {f_prime}")

# 2차 미분
f_double_prime = sp.diff(f_prime, x)
print(f"2차 미분: {f_double_prime}")

# 3차 미분
f_triple_prime = sp.diff(f_double_prime, x)
print(f"3차 미분: {f_triple_prime}")

# 삼각 함수의 고계 미분
g = sp.sin(x)
```

```
# 1차 미분
g_prime = sp.diff(g, x)
print(f"\n삼각 함수 1차 미분: {g_prime}")

# 2차 미분
g_double_prime = sp.diff(g_prime, x)
print(f"삼각 함수 2차 미분: {g_double_prime}")

# 3차 미분
g_triple_prime = sp.diff(g_double_prime, x)
print(f"삼각 함수 3차 미분: {g_triple_prime}")

# 4차 미분
g_quadruple_prime = sp.diff(g_triple_prime, x)
print(f"삼각 함수 4차 미분: {g_quadruple_prime}")
```

- Sympy 라이브러리를 사용하여 변수와 함수를 정의
- sp.diff() 함수를 사용해 1차, 2차, 3차 미분을 계산
- 삼각 함수에 대해 고계 미분을 계산하며, 주기성을 확인할 수 있습니다.