5-2. 라그랑주 승수법(Lagrange Multiplier): 제약이 있는 최적화 문제

라그랑주 승수법은 제약 조건이 있는 최적화 문제를 풀기 위한 방법입니다. 딥러닝에서 최적화 문제는 매우 중요한데, 특히 라그랑주 승수법은 여러 제약을 만족시키면서 최적의 해를 구할 때 유용합니다. 이 방법은 미적분의 극값을 찾는 문제를 제약 조건이 추가된경우로 확장한 것으로 볼 수 있습니다.

1. 제약 조건이 있는 최적화 문제

   우리가 최적화하려는 목적 함수가 $f(x, y)$이고, 그에 대한 제약 조건이 $g(x, y) = 0$일 때, 이 최적화 문제는 단순히 목적 함수의 극값을 찾는 문제가 아닙니다. 이 경우에는 제약 조건을 만족시키면서 최적화를 해야합니다.

   예를 들어, 목적 함수$f(x, y)$를 최소화하는 값 x와 y를 찾으려고 할 때, 이 값들이 제약 조건 $g(x, y) = 0$도 동시에 만족해야 한다면, 라그랑주 승수법을 사용합니다.

2. 라그랑주 승수법 아이디어

   라그랑주 승수법은 제약 조건을 만족시키면서 최적화를 하기 위해 라그랑주 함수를 정의합니다.

   $$L(x, y, \lambda) = f(x, y) + \lambda \cdot g(x, y)$$

   - $f(x, y)$는 우리가 최소화(혹은 최대화)하려는 목적 함수입니다.
   - $g(x, y) = 0$은 제약 조건입니다.
   - λ는 라그랑주 승수라고 불리는 추가 변수로, 제약 조건에 대한 중요도를 조절하는 역할을 합니다.

   이때, 최적화 문제는 이제 라그랑주 함수 $L(x, y, \lambda)$를 최적화하는 문제로 바뀝니다. 이를 풀기 위해서는 라그랑주 함수에 대해 모든 변수에 대한 편미분이 0이 되는 값을 찾습니다.

즉, 다음 조건을 마족하는 값을 찾습니다.

$$\frac{\partial L}{\partial x} = 0,\ \frac{\partial L}{\partial y} = 0,\ \frac{\partial L}{\partial \lambda} = 0$$

3. 예제

   다음과 같은 목적 함수 $f(x, y) = x^2 + y^2$을 원 $g(x, y) = x^2 + y^2 - 1 = 0$위에서 최소화한다고 해봅시다.

   라그랑주 함수

   목적 함수와 제약 조건을 사용하여 라그랑주 함수를 정의합니다.

   $$L(x, y, \lambda) = x2 + y2 + \lambda(x2 + y2 - 1)$$

   이 함수에 대해 각 변수 x, y그리고 λ에 대해 편미분을 하고, 그 미분값을 0으로 만드는 값을 찾습니다.

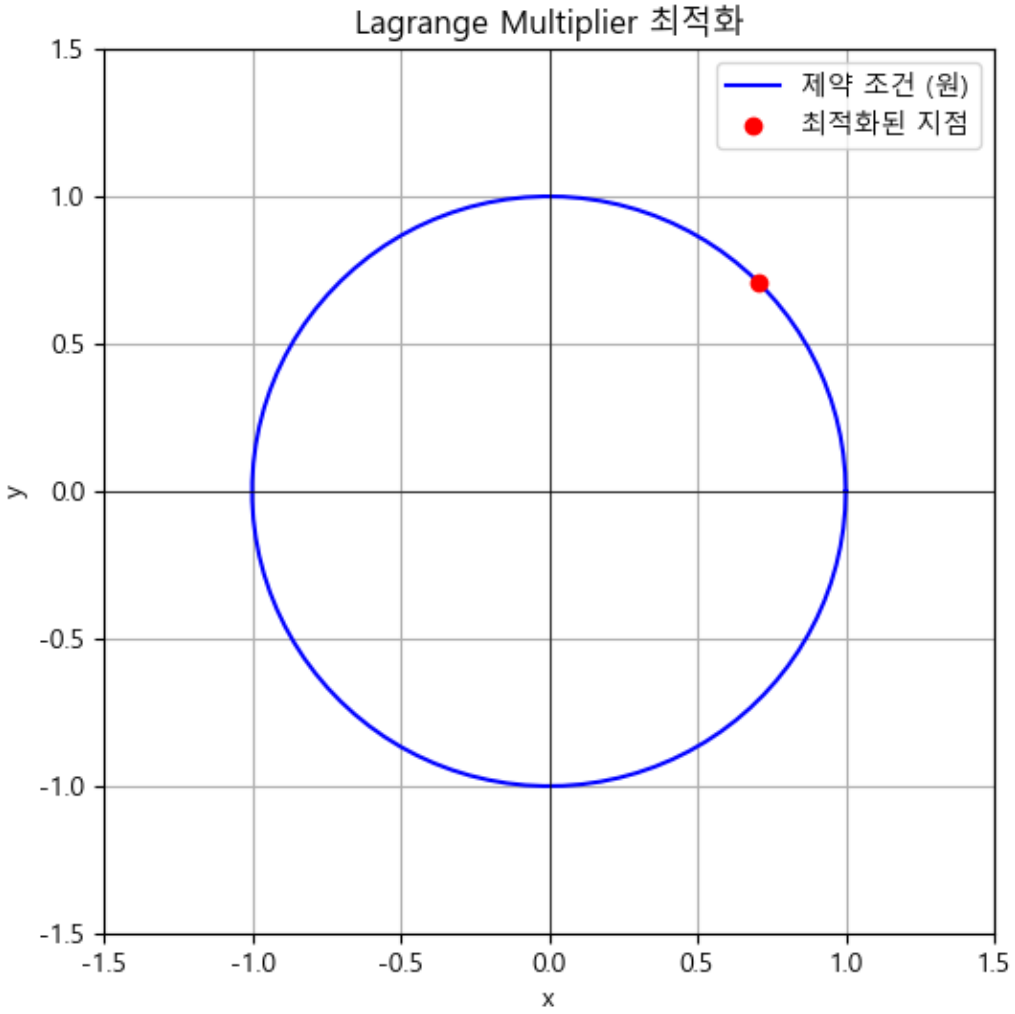
   라그라주 함수 미분

   1. $\frac{\partial L}{\partial x} = 2x + 2\lambda x = 0$
   2. $\frac{\partial L}{\partial y} = 2y + 2\lambda y = 0$
   3. $\frac{\partial L}{\partial \lambda} = x2 + y2 - 1 = 0$

   이 세 개의 방정식을 풀면 최적화 해를 찾을 수 있습니다.

4. 라그랑주 승수법의 딥러닝 활용

   딥러닝에서 라그랑주 승수법은 제약 조건을 만족하는 최적화 문제에 활용될 수 있습니다. 예를 들어, 모델을 학습할 때 특정 변수나 레이어의 가중치를 제한하거나, 정규화 조건을 부과할 때 사용될 수 있습니다. 주로 Regularization(정규화)과 관련된 최적화 문제에서 유용하게 쓰입니다. 또한 특정한 제약 조건 하에서 최적화를 요구하는 강화학습이나 GAN 등의 모델에서 활용되기도 합니다.



- 목적 함수는 $f(x, y) = x^2 + y^2$이며, 이는 원점에서의 거리를 최소화하려고 합니다.
- 제약 조건은 $g(x, y) = x^2 + y^2 - 1 = 0$으로, 이는 반지름이 1인 원을 의미합니다.
- fsolve 함수를 사용하여 방정식을 풀고, 라그랑주 승수를 사용해 최적의 해를 구합니다.
- 그래프에서는 제약 조건으로 주어진 원과, 그 위에서 최적화된 점을 시각화합니다.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import fsolve

# 목적 함수 f(x, y) = x^2 + y^2
def f(x, y):
    return x**2 + y**2
```

```
# 제약 조건 g(x, y) = x^2 + y^2 - 1
def g(x, y):
    return x**2 + y**2 - 1

# 라그랑주 함수의 편미분 방정식
def equations(vars):
    x, y, lam = vars
    eq1 = 2*x + 2*lam*x  # ∂L/∂x = 0
    eq2 = 2*y + 2*lam*y  # ∂L/∂y = 0
    eq3 = x**2 + y**2 - 1  # g(x, y) = 0 (제약 조건)
    return [eq1, eq2, eq3]

# 초기 추정값
initial_guess = [0.5, 0.5, 0.5]

# fsolve를 사용해 방정식 풀기
solution = fsolve(equations, initial_guess)
x_sol, y_sol, lam_sol = solution

print(f"최적화 결과: x = {x_sol}, y = {y_sol}, λ = {lam_sol}")
print(f"최소화된 목적 함수 값: f(x, y) = {f(x_sol, y_sol)}")

# 그래프 그리기 (원과 최적화된 지점)
theta = np.linspace(0, 2*np.pi, 100)
x_circle = np.cos(theta)
y_circle = np.sin(theta)

plt.figure(figsize=(6, 6))
plt.plot(x_circle, y_circle, label="제약 조건 (원)", color="blue
plt.scatter(x_sol, y_sol, color="red", label="최적화된 지점", zo
plt.xlim(-1.5, 1.5)
plt.ylim(-1.5, 1.5)
plt.xlabel('x')
plt.ylabel('y')
plt.axhline(0, color='black',linewidth=0.5)
plt.axvline(0, color='black',linewidth=0.5)
plt.title("Lagrange Multiplier 최적화")
plt.legend()
```

```
plt.grid(True)
plt.show()
```