Thuật toán Newton-Raphson(func, derivFunc, initial\_guess)

Đầu vào:

func - Hàm cần tìm nghiệm

derivFunc - Đạo hàm của hàm func

initial\_guess - Giá trị khởi tạo cho nghiệm

Hằng số:

EPSILON - Sai số cho phép

MAX\_ITERATIONS - Số lần lặp tối đa

Biến:

x - Giá trị hiện tại của nghiệm

h - Sai số của lần lặp hiện tại

iterations - Bộ đếm số lần lặp

1. Đặt x = initial\_guess

2. Tính h = func(x) / derivFunc(x)

3. Đặt iterations = 0

4. Trong khi |h| >= EPSILON và iterations < MAX\_ITERATIONS, làm:

a. Tính h = func(x) / derivFunc(x)

b. Cập nhật x = x - h

c. Tăng iterations lên 1

d. In thông tin debug: iterations, x, func(x), derivFunc(x), h

5. Nếu iterations < MAX\_ITERATIONS thì:

a. In "Nghiệm của hàm là x"

6. Ngược lại:

a. In "Phương pháp không hội tụ sau MAX\_ITERATIONS lần lặp"

Kết thúc thuật toán

Thuật toán SGD(x\_init, learning\_rate, epochs, gradFunc, objFunc)

Đầu vào:

x\_init - Giá trị khởi tạo của tham số x

learning\_rate - Tốc độ học (learning rate)

epochs - Số lần lặp tối đa

gradFunc - Hàm tính gradient của hàm mục tiêu

objFunc - Hàm mục tiêu cần tối ưu hóa

Biến:

x - Giá trị hiện tại của tham số x

x\_prev - Giá trị của x từ lần lặp trước

1. Đặt x = x\_init

2. Lặp qua các epoch từ 1 đến epochs:

a. Lưu x\_prev = x

b. Cập nhật x bằng quy tắc gradient:

x = x - learning\_rate \* gradFunc(x)

c. In thông tin trạng thái hiện tại:

"Iteration i: x = x, function value = objFunc(x)"

d. Kiểm tra hội tụ:

Nếu |x - x\_prev| < 1e-5 thì:

i. In "Hội tụ sau i lần lặp."

ii. Thoát khỏi vòng lặp

e. Nếu số lần lặp đã đạt giới hạn epochs thì:

i. In "Đã đạt số lần lặp tối đa."

ii. Thoát khỏi vòng lặp

3. Trả về objFunc(x) (giá trị của hàm mục tiêu tại x)

Kết thúc thuật toán

Thuật toán optimize(alpha, beta, x\_init, objFunc, gradFunc)

Đầu vào:

alpha - Hệ số điều chỉnh (learning rate)

beta - Hệ số giảm chấn (momentum factor)

x\_init - Giá trị khởi tạo của x

objFunc - Hàm mục tiêu cần tối ưu hóa

gradFunc - Hàm tính gradient của hàm mục tiêu

Biến:

x - Giá trị hiện tại của tham số x

v - Biến lưu động lượng (velocity)

x\_prev - Giá trị của x từ lần lặp trước

iterations - Bộ đếm số lần lặp

1. Đặt x = x\_init

2. Đặt v = 0.0

3. Đặt iterations = 0

4. Trong khi iterations < 1000, làm:

a. Tăng iterations lên 1

b. Lưu x\_prev = x

c. Cập nhật động lượng:

v = beta \* v + (1 - beta) \* gradFunc(x)

d. Cập nhật tham số:

x = x - alpha \* v

e. In thông tin:

"Giá trị hàm mục tiêu tại lần lặp iterations: x = x, function value = objFunc(x)"

f. Kiểm tra hội tụ:

Nếu |x - x\_prev| < 1e-5 thì:

i. In "Tối ưu hóa hàm mục tiêu hoàn tất."

ii. Thoát khỏi vòng lặp

5. Nếu iterations == 1000, thì:

a. In "Đạt số lần lặp tối đa mà không hội tụ."

6. Kết thúc thuật toán