TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC

---o0o---

BÁO CÁO GIẢI TÍCH SỐ

ĐỀ TÀI:

TÌM MIN MAX CỦA HÀM MỘT BIẾN TRÊN KHOẢNG ĐÓNG $\mathbb{R}$

Giảng viên hướng dẫn: Cô Hà Thị Ngọc Yến

Nhóm sinh viên thực hiện:

Họ và tên	MSSV
Trần Viết Tài	20185402
Nguyễn Hoàng Minh	20195902
Phạm Anh Đức	20195859
Phan Tiến Đạt	20195854
Nguyễn Thành Long	20195898

Hà Nội, ngày 15 tháng 4 năm 2021

LỜI NÓI ĐẦU

Giải tích số là môn học nghiên cứu về các phương pháp, các thuật toán sử dụng xấp xỉ cho các bài toán giải tích (phân biệt với bài toán rời rạc). Lớp bài toán đầu tiên được đề cập đến là các phương pháp để giải phương trình phi tuyến f(x) = 0 mà trước tiên, ta phải đi giải một bài toán phụ, đó là tìm giá trị Min-Max của hàm một biến trong khoảng đóng cho trước.

Sau một thời gian tìm hiểu về đề tài "Thuật toán tìm giá trị lớn nhất, nhỏ nhất của hàm số một biến trên một đoạn con đóng của $\mathbb{R}$" bọn em làm bản báo cáo này để trình bày về nội dung phương pháp, hệ thống ví dụ, cùng với thuật toán, chương trình cụ thể để giải quyết đề tài này. Trong quá trình làm báo cáo cũng như xây dựng chương trình, thuật toán… không tránh khỏi sai sót vậy chúng em mong nhận được những lời nhận xét bổ ích từ cô cũng như mọi người để đề tài này của chúng em hoàn thiện hơn. Bọn em cũng gửi lời cảm ơn cô Hà Thị Ngọc Yến đã giúp chúng em trong quá trình tìm hiểu đề tài này. Hi vọng những phương pháp sau đây sẽ là một công cụ hữu hiệu có thể sử dụng mỗi khi gặp bài toán tìm nghiệm của phương trình và ứng dụng vào các bài toán khác.

MỤC LỤC

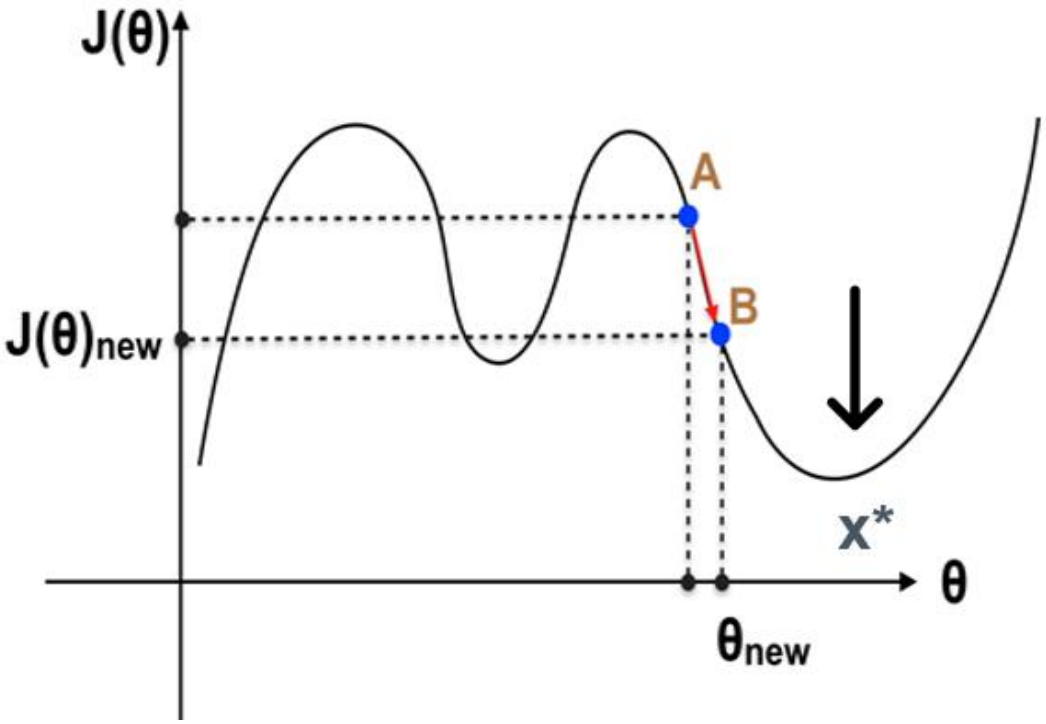
1. Giới thiệu bài toán

Trong kiến thức toán phổ thông chúng ta đã biết muốn tìm cực trị một hàm số $y = f(x)$ chúng ta sẽ giải phương trình đạo hàm của hàm số:

$$f'(x) = 0$$

Tuy nhiên phương trình trên không phải lúc nào cũng giải được dễ dàng vì có những trường hợp việc giải phương trình trên là bất khả thi. Vậy khi gặp những tình huống này chúng ta phải làm gì? May thay thuật toán Gradient Descent cho chúng ta cách thức tìm các điểm cực tiểu cục bộ này một cách xấp xỉ sau một số vòng lặp. Trong thực tế các giá trị dữ liệu không đúng 100% mà đôi khi chúng ta chỉ có những con số gần đúng.

2. Xây dựng thuật toán Gradient Descent:



Giả sử ta bắt đầu tìm cực trị của hàm số này từ điểm A. Vậy ta cần đưa điểm A gần về điểm cực trị x^* nhất. Như chúng ta đã biết, khi đạo hàm tại 1 điểm x nào đó có tính chất $f'(x) < 0 \rightarrow$ hàm số này đang giảm và có khả năng tiến đến cực trị, nếu có cực trị thì chắc chắn sẽ là cực tiểu. Còn khi đạo hàm tại một điểm $f'(x) > 0$ thì hàm số này đang tăng và có khả năng tiến tới cực trị, nếu có thì sẽ là cực đại.

Để dễ minh họa, không mất tính tổng quát, ta xét trường hợp đi tìm cực tiểu. Vậy ta sẽ di chuyển từ điểm A một đoạn Δx, nếu A nằm bên trái x^* thì Δx sẽ mang dấu dương còn A mà nằm bên phải x^* thì Δx sẽ mang dấu âm. Do đó ta có nhận xét rằng dấu của Δx sẽ ngược dấu với $f'(x)$.
Ngược lại, điểm A càng xa về phía bên phải thì giá trị f'(x_A) sẽ càng lớn hơn 0 và ngược lại. Vậy lượng dịch chuyển Δx, một cách trực quan nhất chính là tỉ lệ nghịch với $f'(x_A)$.

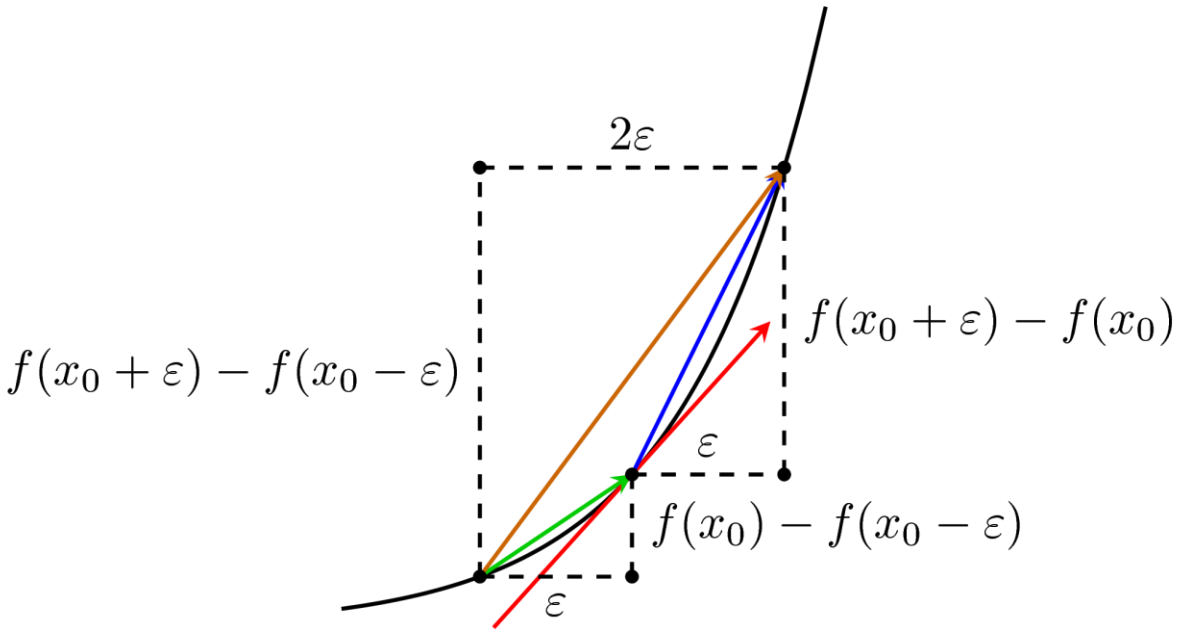
Với nhận xét phía trên, ta rút ra được công thức sau:

$$x_B = x_A + \mathbf{sign} * \eta * f'(x_A) \ (*)$$

Ở đây biến **sign** thể hiện dấu (âm hoặc dương) của đạo hàm tại điểm x_A. Nếu đạo hàm dương thì ta đang đi tìm cực đại. Ngược lại, nếu đạo hàm âm, ta đang đi tìm cực tiểu. Còn $\boldsymbol{\eta}$ (eta) là một số dương được gọi là **learning rate** (tốc độ học).

- Xây dựng công thức đạo hàm

Trong công thức trên, việc tính đạo hàm tại một điểm là không thể tránh khỏi. Do đó, chúng ta phải đi xây dựng công thức tính xấp xỉ đạo hàm cho máy tính. Hình vẽ dưới đây minh họa phương pháp tính xấp xỉ đạo hàm.



Chúng ta cùng quay lại một chút với Giải tích I: Khai triển Taylor.

Với ε rất nhỏ, ta có hai xấp xỉ sau:

$$f(x+\varepsilon) \approx f(x) + f'(x)\varepsilon + \frac{f''(x)}{2}\varepsilon^2 + \dots$$

và:

$$f(x-\varepsilon) \approx f(x) - f'(x)\varepsilon + \frac{f''(x)}{2}\varepsilon^2 - \dots$$

Từ đó ta có:

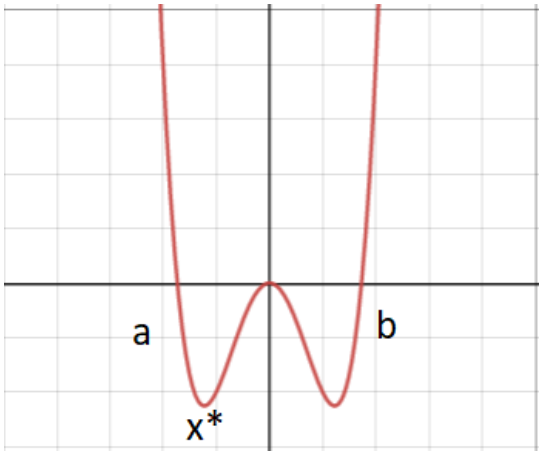
$$\frac{f(x+\varepsilon)-f(x)}{\varepsilon} \approx f'(x)+\frac{f''(x)}{2}\varepsilon+\cdots=f'(x)+O(\varepsilon) \quad (3)$$

$$\frac{f(x+\varepsilon)-f(x-\varepsilon)}{2\varepsilon} \approx f'(x)+\frac{f^{(3)}(x)}{6}\varepsilon^2+\cdots=f'(x)+O(\varepsilon^2) \quad (4)$$

Vậy, nếu xấp xỉ đạo hàm bằng công thức (3) (xấp xỉ đạo hàm phải), sai số sẽ là O(ε). Trong khi đó, nếu xấp xỉ đạo hàm bằng công thức (4) (xấp xỉ đạo hàm hai phía), sai số sẽ là O(ε^2) << O(ε) với rất nhỏ.

Trong thuật toán Gradient Descent này, chúng ta sử dụng công thức xấp xỉ hai phía

3. Xây dựng thuật toán tìm các cực trị trên [a, b]

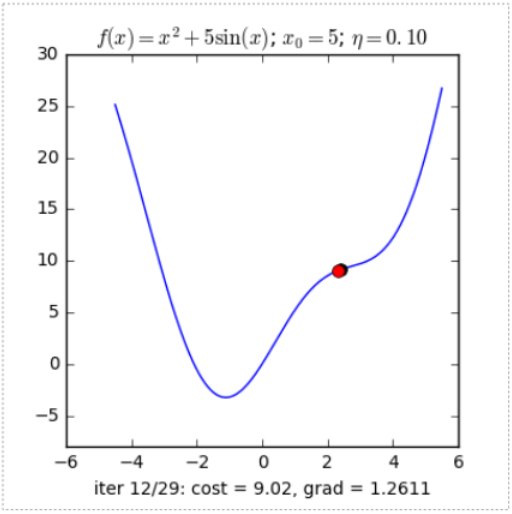


B1: Kiểm tra xem $f'(x_0) < 0$, nếu đúng cực trị tiếp theo (nếu có) là cực tiểu.

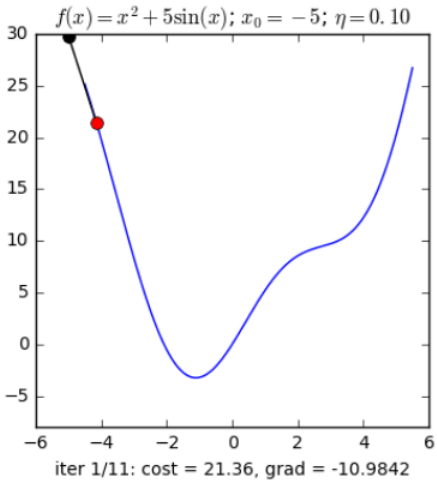
B2: Dùng công thức (*) với **sign** = ± 1 lặp $x_0 \rightarrow x^*$. Tăng 1 khoảng step để $x_0 > x^*$

B3: Quay lại B1 với x_0 mới. Lặp lại đến khi x_0 = b.

4. Sự hội tụ của thuật toán Gradient Descent



$x_0 = 5$

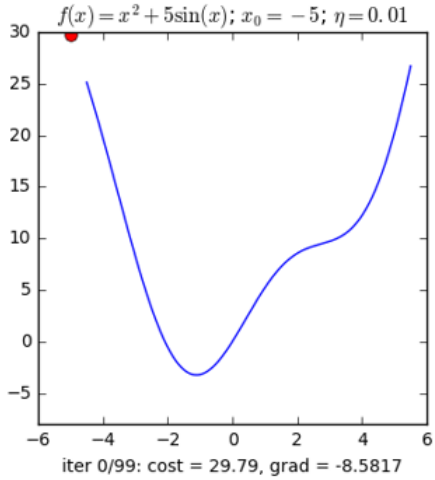


$x_0 = -5$

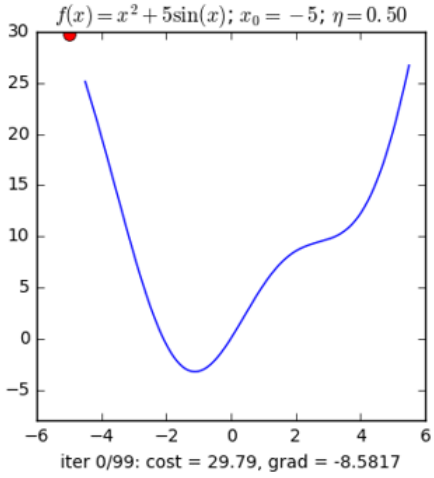
Sự hội tụ của thuật toán phụ thuộc vào 2 yếu tố chính. Thứ nhất phải đề cập đến là điểm khởi tạo đầu. Ta bắt đầu từ x_0 = 5 (hình trái), đường đi của điểm A có chứa một khu vực có đạo hàm khá nhỏ gần điểm có hoành độ bằng 2, ở đoạn này độ dốc của đồ thị khá nhỏ nên thuật toán duyệt ở đây khá lâu tương tự như việc đi xe xuống đốc

Ứng với x_0 = −5 (hình phải), nghiệm hội tụ nhanh hơn vì điểm

x_0 = −5 gần với cực trị hơn và dáng đồ thị trên khoảng [-5, 0] cũng thoải hơn so với hình bên trái.



η = 0.01



η =0.5

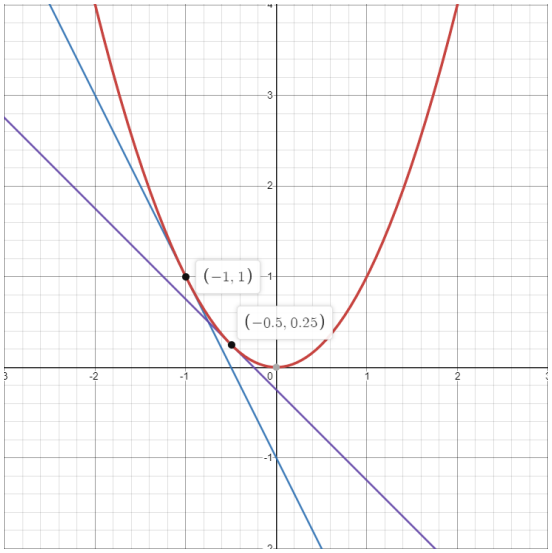
Sự hội tụ còn phụ thuộc vào yếu tố thứ 2 là Learning Rate hay còn gọi là η (eta). Với η = 0.01, tốc độ hội tụ rất chậm, thuật toán vẫn chạy tiếp dù đã rất gần với đích nhưng cuối cùng phải dừng lại do đã được giới hạn số lần lặp tối đa: 100 lần lặp. Do đó η quá thấp sẽ ảnh hưởng tới tốc độ của thuật toán, thậm chí có thể không bao giờ tới được đích.

Với η = 0.5, thuật toán tiến rất nhanh tới gần đích. Ở đây có thể thấy, điểm xét nhảy liên tục từ dốc này sang dốc kia của đồ thị. Tuy nhiên, thuật toán không hội tụ được vì bước nhảy quá lớn.

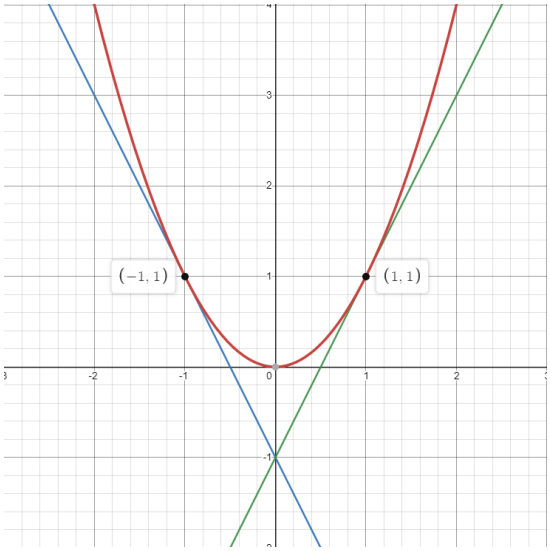
Do đó, việc lựa chọn η rất quan trọng. Ta có thể η khác nhau ở mỗi vòng lặp. Từ đó ta phát triển thêm thuật toán η động.

5. Lựa chọn learning rate (η) động

Eta(η) động là giải pháp chọn η qua mỗi vòng lặp để tránh việc thuật toán chạy quá chậm hoặc bỏ qua cực trị như hai trường hợp trên đã đề cập.



Khi đạo hàm của hai giá trị x liên tiếp có cùng dấu, tức là x chưa vượt qua điểm cực trị x*, ta phải tăng eta lên (ở đây tăng eta lên 3 lần) nhằm giúp tăng tốc độ x đi đến x*.



Khi đạo hàm của hai giá trị x liên tiếp trái dấu nhau, tức là chúng nằm ở hai phía so với điểm cực trị x*, ta phải giảm eta đi (ở đây giảm eta đi một nửa) nhằm giúp hàm hội tụ tốt hơn, không bị phân kỳ.

6. Điều kiện dừng của thuật toán Gradient Descent

Vì phụ thuộc vào hàm f(x), thuật toán có thể chạy mãi và rơi vào vòng lặp vô hạn, do đó, ta phải xây dựng điều kiện dừng cho thuật toán. Dưới đây là hai phương pháp:

- Giới hạn số vòng lặp: đảm bảo rằng chương trình không quá lâu và không làm đơ máy tính. Dù kết quả có thể chưa đạt đến giá trị sai số mong muốn nhưng kết quả này vẫn tạm chấp nhận được.

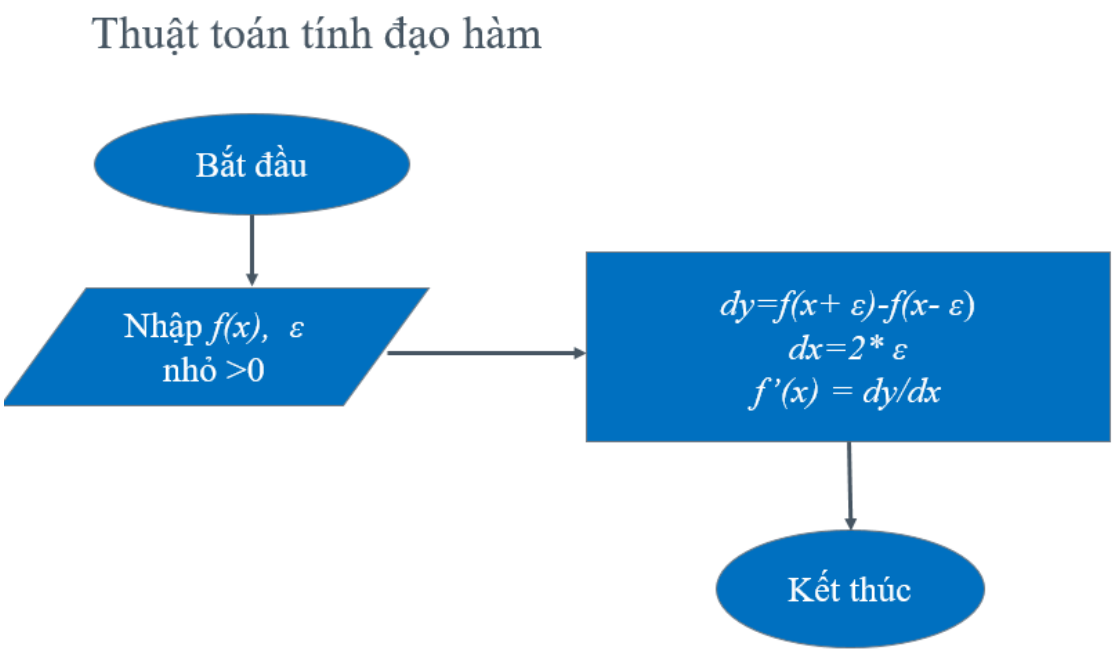
-Kiểm tra giá trị đạo hàm: nếu gradient tại x_0 sau n lần lặp nhỏ hơn một số ε cho trước thì dừng lại.

7. Sơ đồ khối thuật toán Gradient Descent

Như đã nói ở trên, việc đầu tiên ta cần đi tìm đạo hàm. Dưới đây là thuật toán để đi tìm đạo hàm.

Input: hàm số f(x), sai số ε nhỏ tùy ý.

Output: giá trị đạo hàm tại f(x).

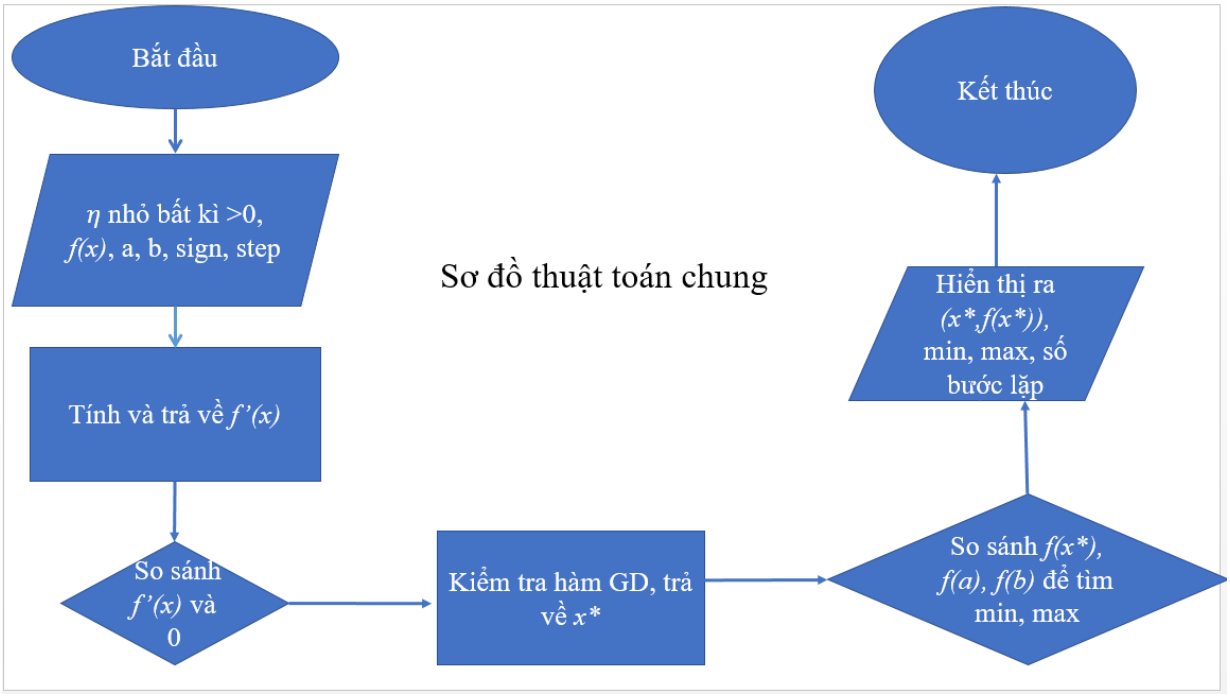


Dưới đây là sơ đồ thuật toán chung của bài toán tìm giá trị lớn nhất, giá trị nhỏ nhất trên khoảng đóng [a, b] ∈ ℝ.

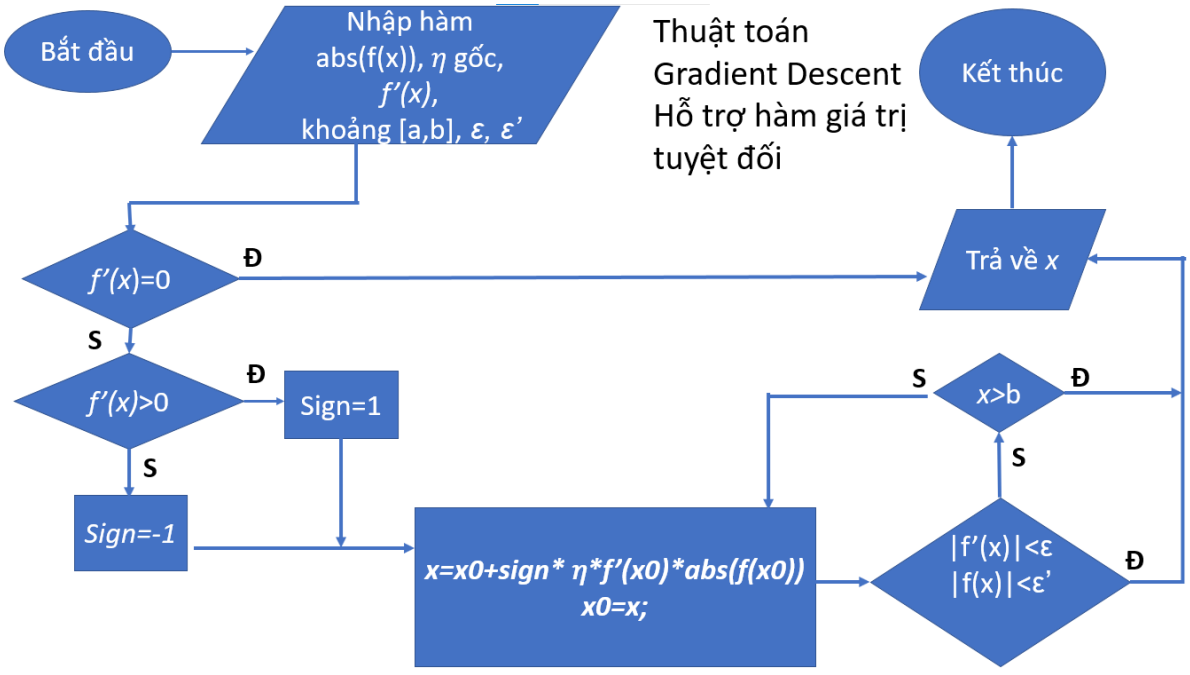
Input: Số η nhỏ bất kì tùy ý, hàm số f(x), bước nhảy khi gặp cực trị (step)

Output: Giá trị lớn nhất, nhỏ nhất của hàm số trên đoạn

[a, b], số lần lặp.



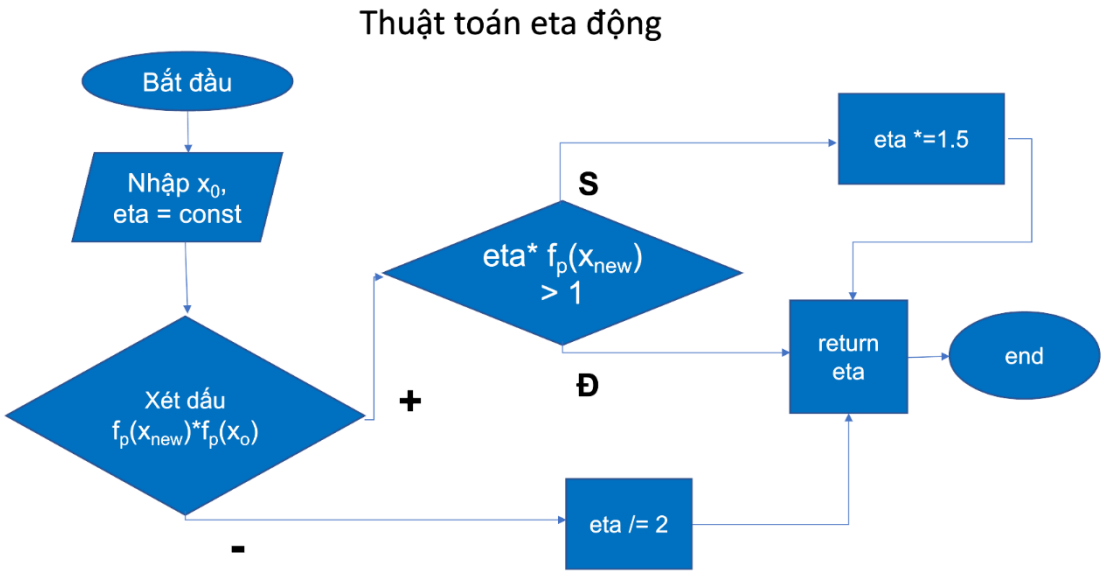
Đây là sơ đồ thuật toán Gradient Descent chính. Ở lần sửa đổi này, nhóm chúng em đã hỗ trợ hàm giá trị tuyệt đối.



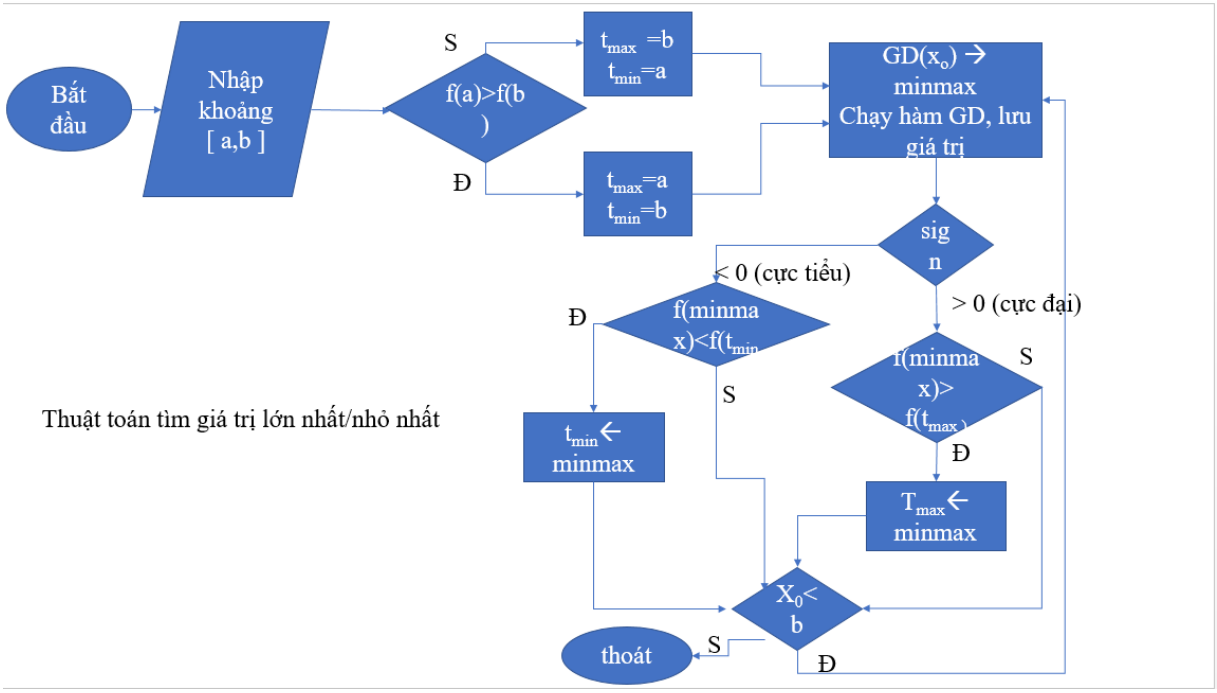
Đây là sơ đồ thuật toán thay đổi eta sau mỗi vòng lặp

Input: giá trị eta cũ.

Output: Giá trị eta mới.



Cuối cùng là sơ đồ thuật toán tìm tất cả các điểm cực trị trên [a, b] bằng cách so sánh giá trị các điểm cực trị tìm được với hai đầu mút.



8. Code

```
#Một số kí hiệu toán học khi nhập hàm:
    #Dấu mũ (a^b): Nhập a**b
    #Lấy trị tuyệt đối(|f(x)|,f(|x|)):Nhập Abs(f(x)),f(Abs(x))
    #Lấy căn thức (sqrt()): Nhập sqrt()
    #Hàm logarith: log(f(x))/log(a) với a là cơ số


from sympy import *
from sympy.parsing.sympy_parser import parse_expr
from matplotlib.animation import FuncAnimation
from sympy.plotting import plot
import matplotlib.pyplot as plt
import numpy as np
import time
import sys


#Khai báo các hằng số
ETA=0.0001
STEP=0.001
global expr


#vào hàm main
def main():
    Input=int(input("1.Dùng hàm số cho trước.\n2.Tự nhập hàm số.\nLựa chọn của bạn:"))
```

```
if (Input==1):
    print("Lựa chọn hàm số cho trước:")
    print("1.f(x)=sin(x)+x-x^2")
    print("2.f(x)=x^4-4x^2+4")
    print("3.f(x)=|e^x+sin(x^2)|")
    print("4.f(x)=cos(x^2)+sin(x^2)+x^2")
    print("5.f(x)=")
    ham=int(input("Chọn hàm số thứ:"))
    if (ham==1):
        expr="sin(x)+x-x**2"
    if (ham==2):
        expr="x**4-4*x**2+4"
    if(ham==3):
        expr="Abs(exp(x)+sin(x**2))"
    if(ham==4):
        expr="cos(x**2)+sin(x**2)+x**2"
if (Input==2):
   expr=input("Nhap ham so can tim\nf(x)=")

x= Symbol('x'),#sử dụng x làm biến toán học
f=lambdify(x, parse_expr(expr))#tạo hàm số f(x)
a=float(input("Nhập khoảng [a,b]\nNhập a="))
b=float(input("Nhập b="))
if abs(a)>abs(b):Xlim=abs(a)
else:Xlim=abs(b)
```

```
    plot(parse_expr(expr),axis=True,xlim=(-Xlim,Xlim), ylim=(-
2*Xlim,2*Xlim), autoscale = true,title="Đồ thị f(x)="+expr+"\n")


    def fp(x):#Tính đạo hàm
        dy= f(x+0.000001)-f(x-0.000001)
        dx= 2*0.000001
        return dy/dx
    def xetdau(x):
        if fp(x)==0: return 0
        if fp(x)>0: return +1
        if fp(x)<0: return -1
    def GD2(eta,x0):#Thuật toán gradient descent Eta động
        sign=0
        sign=xetdau(x0)
        it=0
        varX=[x0]
        for it in range(300000):
            deltaX=eta*fp(x0)*abs(f(x0)) # nhân thêm abs(f(x0)) khiến nó
chạy chậm lại khi đến gần nghiệm của f(x)=0 (nếu hàm không có trị
tuyệt đối thì sẽ làm GD chậm lại)
            if deltaX>0.2 : deltaX=0.2   #giới hạn tốc độ dịch chuyển của
x(phòng trường hợp x nhảy quá xa)
            if deltaX<-0.2: deltaX=-0.2
            x_new = x0 + sign*deltaX
            if abs(fp(x_new))<0.005:abslim=1e-15
            else: abslim=1e-7
```

```
        if abs(fp(x_new)) < 1e-20 or abs(f(x_new))<abslim or
x_new>b:
            break

        #eta động
        if (fp(x_new)*fp(x0)>0 ):eta*=3
        if fp(x_new)*fp(x0)<0:eta/=5
        x0=x_new
        varX.append(x_new)
      if x_new<=b: #hiển thị các giá trị cực trị sau các lần lặp

        plt.xlabel("Số lần lặp")
        plt.ylabel("Giá trị của x")
        plt.plot(varX,ls='None',marker='.')

      return (x_new, it, sign)

#cài đặt giá trị max, min ban đầu
   if(f(a)>f(b)):
     tmax=a; tmin=b
   else:
     tmin=a; tmax=b

   Sum=0 #tổng số lần lặp
   x_new=a
```

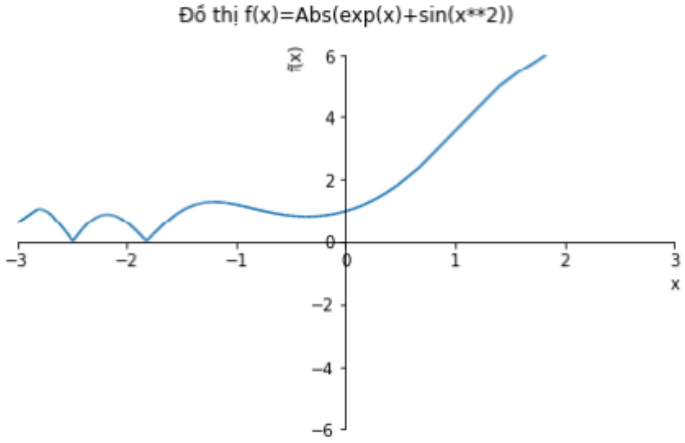
```
    localMin=[]#        Tạo 1 list để lưu cực tiểu
    localMax=[]#        Tạop 1 list để lưu cực đại
    iterationOfEachMinMax=[] #Tạo 1 list để đếm mỗi lần lặp
    i=0

#chương trình chính

    while(i<300):
      (minMax,itera,sign)=GD2(ETA,x_new)
      x_new=minMax
    # Kiểm tra điểm tới hạn có phải cực trị hay không?
      if fp(x_new-0.1)*fp(x_new+0.1)<0 :

        if( sign>0  and minMax <= b ):
          localMax.append(minMax)
          if f(minMax)>f(tmax): tmax=minMax
        if(sign<0   and minMax <= b):
          localMin.append(minMax)
          if f(minMax)<f(tmin): tmin=minMax
        iterationOfEachMinMax.append(itera)
        Sum+=itera
      while (abs(fp(x_new))<1e-9 or abs(f(x_new))<1e-3  and
x_new<b):
        x_new+= STEP
      i+=1
```

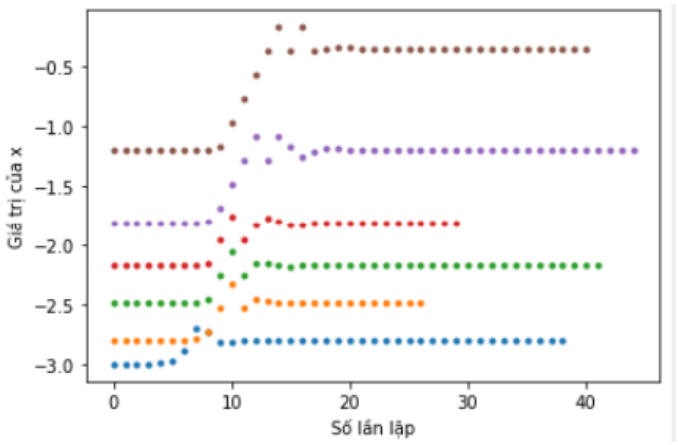
```
        if(x_new>b): break
    plt.show()

    print("Số lần lặp khi đi tìm mỗi cực trị:",iterationOfEachMinMax)
    print("\nTổng số lần lặp:",Sum)
    print("\nLocal min",localMin)
    print("\nLocal max",localMax)
    print("\nGlobal min (",tmin,",",f(tmin),")" )
    print("\nGlobal max (",tmax,",",f(tmax),")")

main()
luachon=str(input("Bạn có muốn chạy lại chường trình không? Yes/No\nLựa chọn của bạn:"))
Luachon=luachon.upper()
while (Luachon=='YES' or Luachon=='Y'):
    main()
    luachon=str(input("Bạn có muốn chạy lại chường trình không? Yes/No\nLựa chọn của bạn:"))
    Luachon=luachon.upper()
if (Luachon=='NO' or Luachon=='N'):
    text="Chương trình đang kết thúc . . .\n"
    for char in text:
        sys.stdout.write (char)
        time.sleep(0.07)
```

```
print("Đã kết thúc!").
```

9. Ví dụ minh họa thuật toán bằng Python

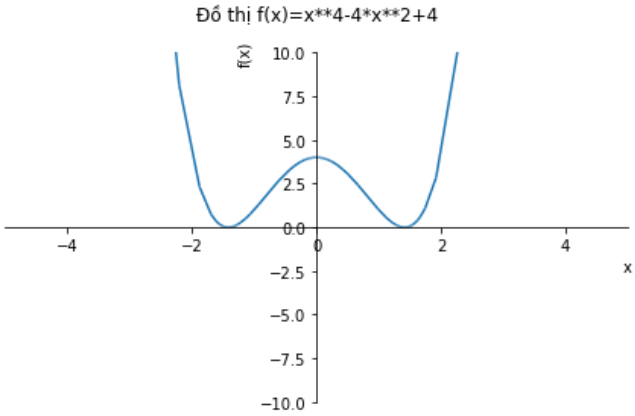
- f(x)=|e^x + sin(x^2)| trong khoảng [-3, 3]

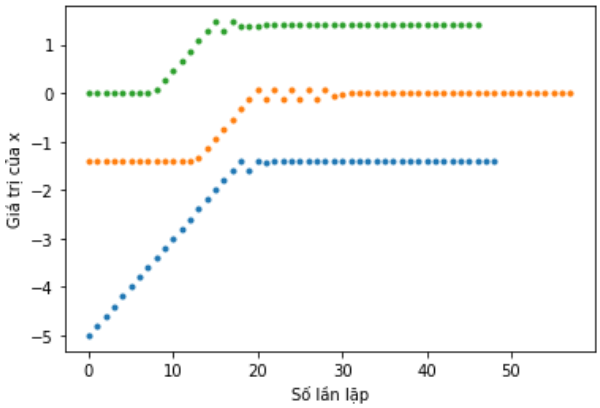





```
Số lần lặp khi đi tìm mỗi cực trị: [38, 26, 41, 29, 44, 40]

Tổng số lần lặp: 218

Local min [-2.490016356855722, -1.8178791015007059, -0.3537815870696506]

Local max [-2.800559000630809, -2.1767958964396814, -1.2022878992148265]

Global min ( -1.8178791015007059 , 3.176230134460667e-08 )

Global max ( 3.0 , 20.497655408429424 )
```

- f(x)=x^4-4x^2+4 trong khoảng [-5, 5]

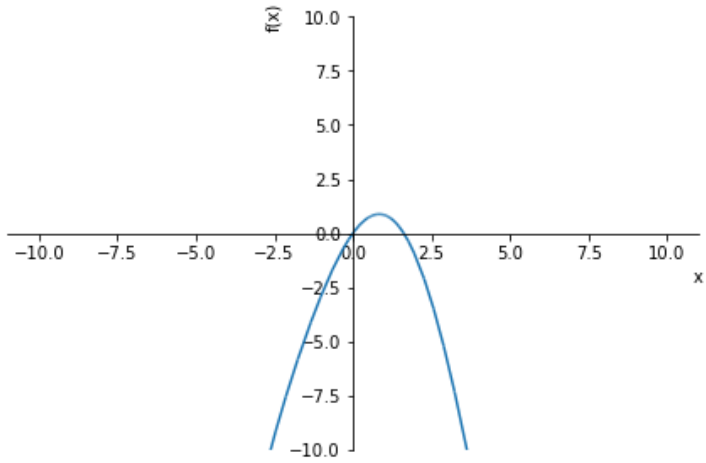





```
Số lần lặp khi đi tìm mỗi cực trị: [48, 57, 46]

Tổng số lần lặp: 151

Local min [-1.4142135512585543, 1.4142135536887273]

Local max [4.115182832688865e-12]

Global min ( 1.4142135536887273 , 4.440892098500626e-16 )

Global max ( 5.0 , 529.0 )
```

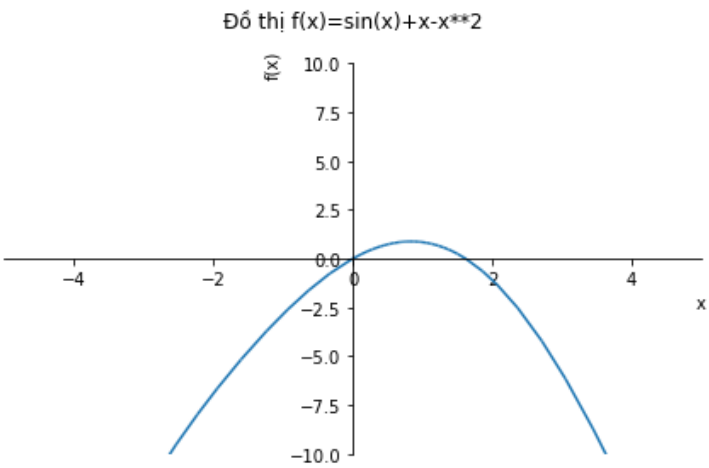
- So sánh thuật toán eta tĩnh và eta động

Xét hàm số $\sin(x) - x + x^2$ *với* $x_0 = -5$



Với eta=0.01 được đặt trước. Thuật toán eta tĩnh cho ta điểm cực trị sau 635 lần lặp.

```
Nhập điểm xuất phát a=-5
Số lần lặp khi đi tìm cực trị: 635
Điểm cực trị là  0.8354292299341026
```




```
Số lần lặp khi đi tìm mỗi cực trị: [77]

Tổng số lần lặp: 77

Local min []

Local max [0.8354295853115437]

Global min ( -5.0 , -29.04107572533686 )

Global max ( 0.8354295853115437 , 0.8790717646895272 )
```

Còn với thuật toán eta động, ta tìm được cực trị sau 77 lần lặp. Tốc độ ở đây tăng đáng kể, gấp hơn 8 lần so với việc sử dụng eta tĩnh (tuy không phải lúc nào cũng vậy do còn phụ thuộc nhiều vào hình dạng đồ thị hàm số). Qua ví dụ minh họa này, ta thấy việc sử dụng eta động giúp giảm đáng kể thời gian và khối lượng tính toán.

10. Đánh giá phương pháp

- Ưu điểm
  - Thuật toán hội tụ nhanh với nhiều dạng hàm.
  - Gradient Descent có lợi thế khi việc khảo sát hàm số để tìm cực trị là quá hoặc quá phức tạp.
- Hạn chế
  - f(x) và f'(x) phải cùng liên tục trên [a, b].
  - Gặp khó khăn với hàm có khoảng cách cực trị quá bé hoặc quá dốc.

11. Phương pháp duyệt thông thường

- Ý tưởng và xây dựng phương pháp
  - Chia [a, b] thành các đoạn rất nhỏ. Tính giá trị tại các điểm rồi so sánh giá trị để tìm ra f(x) nhỏ nhất và lớn nhất.
  - Sai số $|x_n - x^*| <$ step.
- Đánh giá
  - Chương trình khá tốt, sai số có thể điều chỉnh khoảng chia.
  - Thời gian chạy lâu, các bước lặp khá lớn.
- Ví dụ minh họa
  - Hàm số f(x)=sin(x)+x-x^2

```
Nhập b=5
Global min ( -5.0 , -29.04107572533686 )
Global max ( 0.8355712890625 , 0.8790717371637655 )
Tổng số lần lặp: 32769
```

  - Code

```
def duyet(a,b):
    #cài đặt giá trị
    if(f(a)>f(b)):
        tmax=a; tmin=b
    else:
        tmin=a; tmax=b
    it=0
    step=(b-a)/2**15
    x=a
    while x<=b:
        if f(x)<f(tmin):tmin=x
        if f(x)>f(tmax):tmax=x
        x+=step
        it+=1
    return (tmin,tmax,it)
```

12. Ứng dụng

Như phần đầu nhóm đã đề cập đến, phương pháp Gradient Descent có nhiều ứng dụng trong các bài toán của môn học Giải tích số như phương pháp tiếp tuyến, bài toán tìm nghiệm đa thức, tìm điểm cực đại, cực tiểu của hàm một biến, ngoài ra phương pháp Gradient Descent còn có ứng dụng trong Machine Learning nói riêng và Toán Tối Ưu nói chung như các hàm mất mát trong hai bài Linear Regression và K-means Clustering. Ngoài ra ta còn có thể giải phương trình f'(x)=0 bằng các biến thể của Gradient Descent như: Newton's method, Mini-batch Gradient Descent, Batch Gradient Descent, Stochastic Gradient Descent.

13. Tài liệu tham khảo

[1] [Gradient Descent] – Phần 1: "Gradient Descent là gì?" – AI CLUB TUTORIALS (uit.edu.vn)

[2] Machine Learning cơ bản (machinelearningcoban.com)

[3] Giáo trình Giải tích số của thầy Lê Trọng Vinh