Gradient Descent

marco milanesio
MScDSAI

UNIVERSITÉ CÔTE D'AZUR

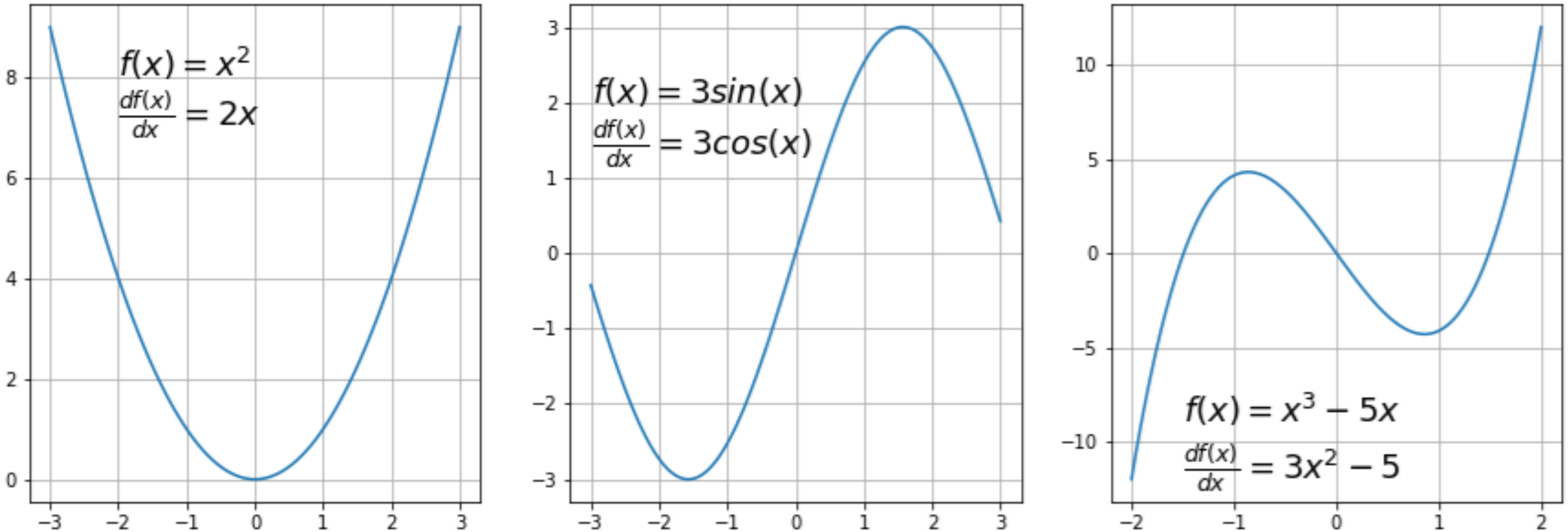
Gradient descent

- Iterative first-order optimisation algorithm (1847)
- Find local minimum/maximum
- ML/DL to minimize cost function

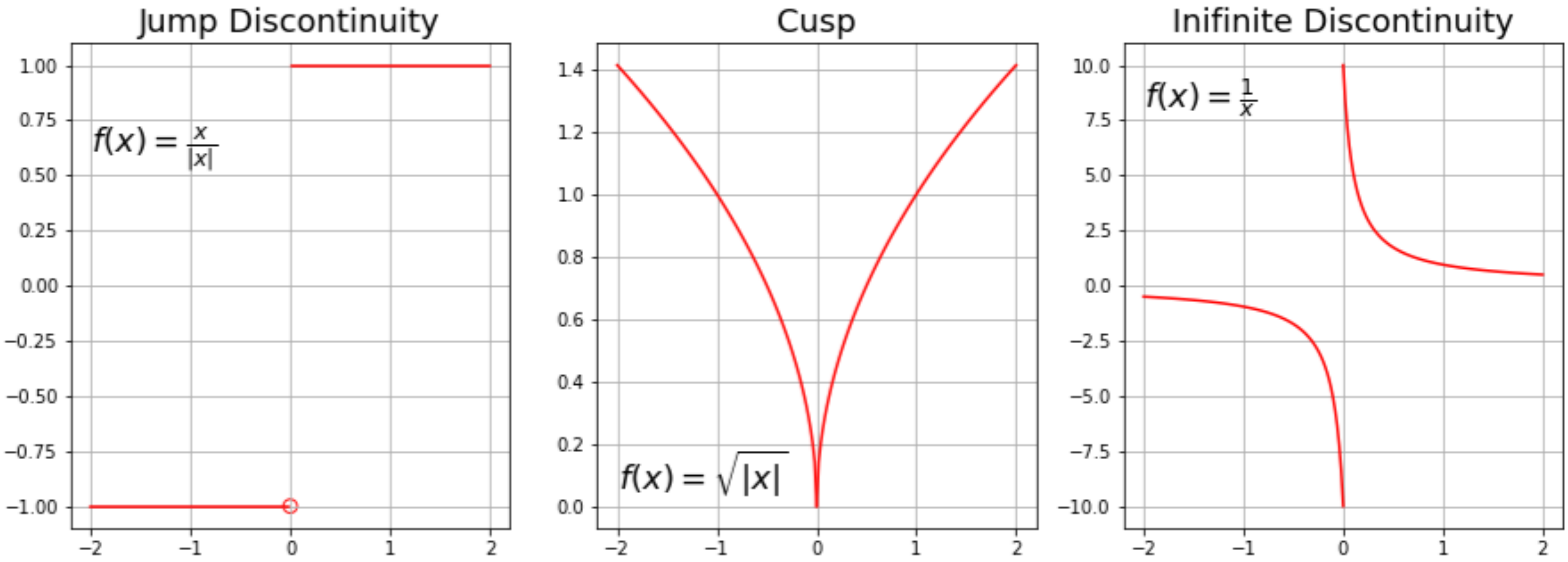
Requirements

- Function must be differentiable
- Function must be convex

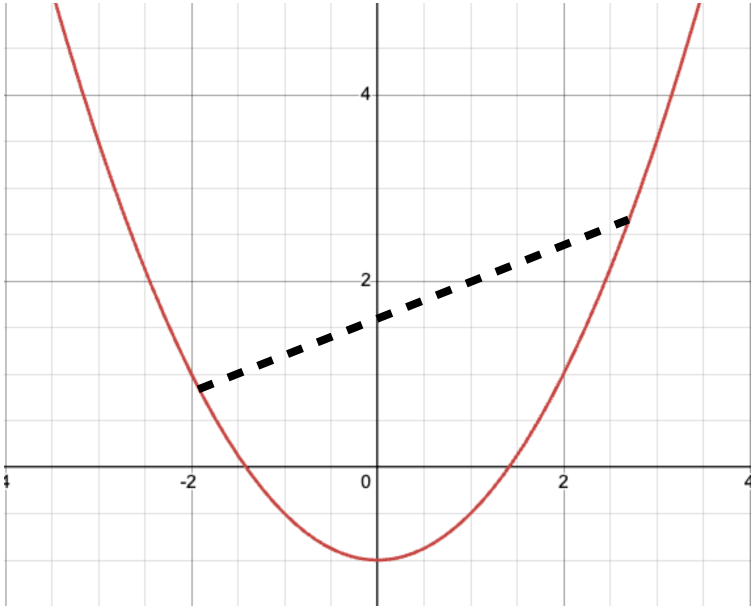
Differentiable

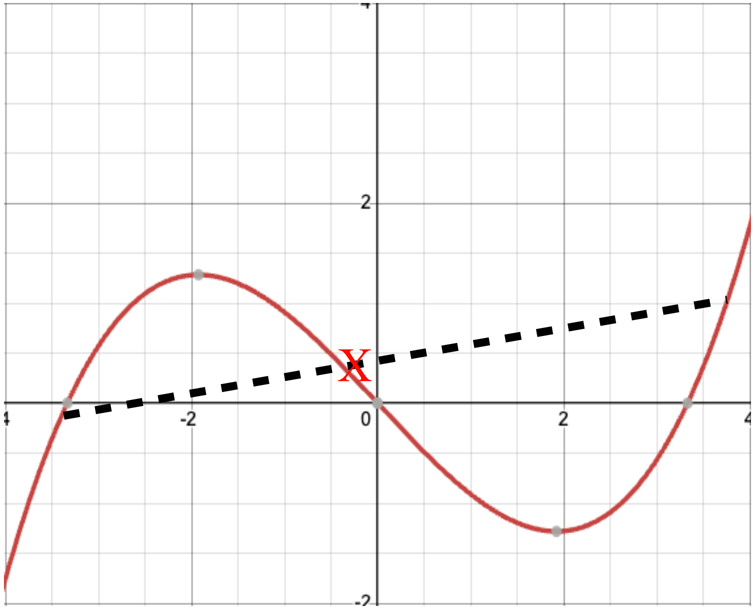


Non differentiable

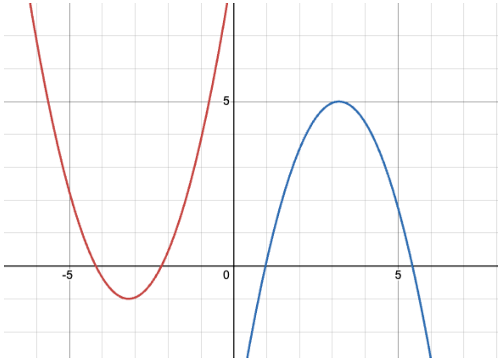


Convexity

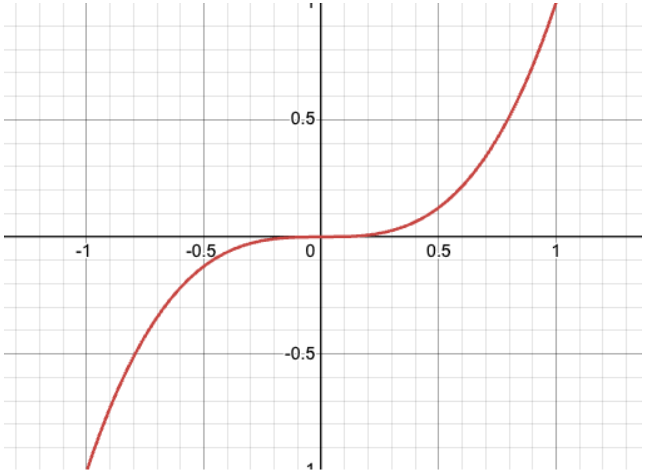




Gradient descent



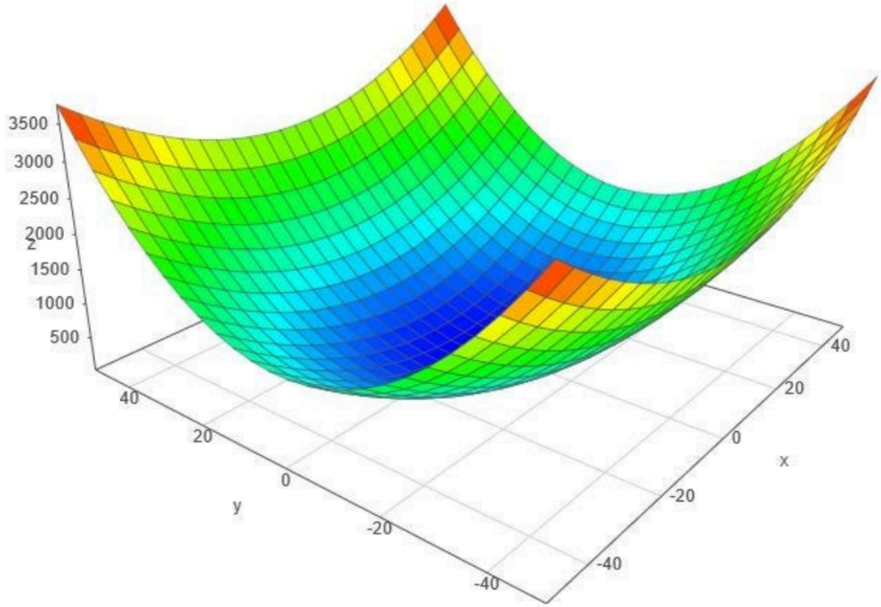
- Find minimum for a convex function
  - or a maximum for a concave function
  - derivative = 0
- BUT:
  - saddle points —> second derivative = 0
  - local minimum: non convex function
    - second derivative not always positive



Gradient

- Slope of a curve in a given point in a specified direction

  - $f(x, y) = 0.5x^2 + y^2$
  - $\nabla f(x, y) = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial x} \end{bmatrix} = \begin{bmatrix} x \\ 2y \end{bmatrix}$
  - $\nabla f(10,10) = \begin{bmatrix} 10 \\ 20 \end{bmatrix}$



@M. Riveill - UniCA

Gradient descent algorithm

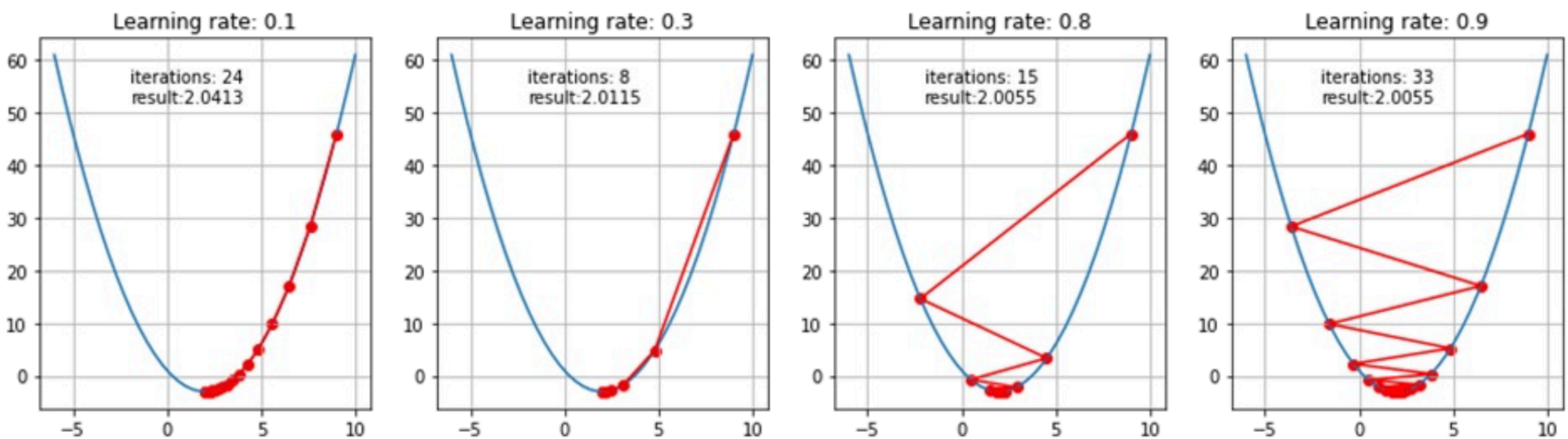
- Choose a point
- While not (c1 and c2):
  - Compute the gradient
  - Scale it by a "learning" factor η
  - Subtract the value (minimize)
  - Update the point

$$p_{t+1} = p_t - \eta \nabla f(p_t)$$

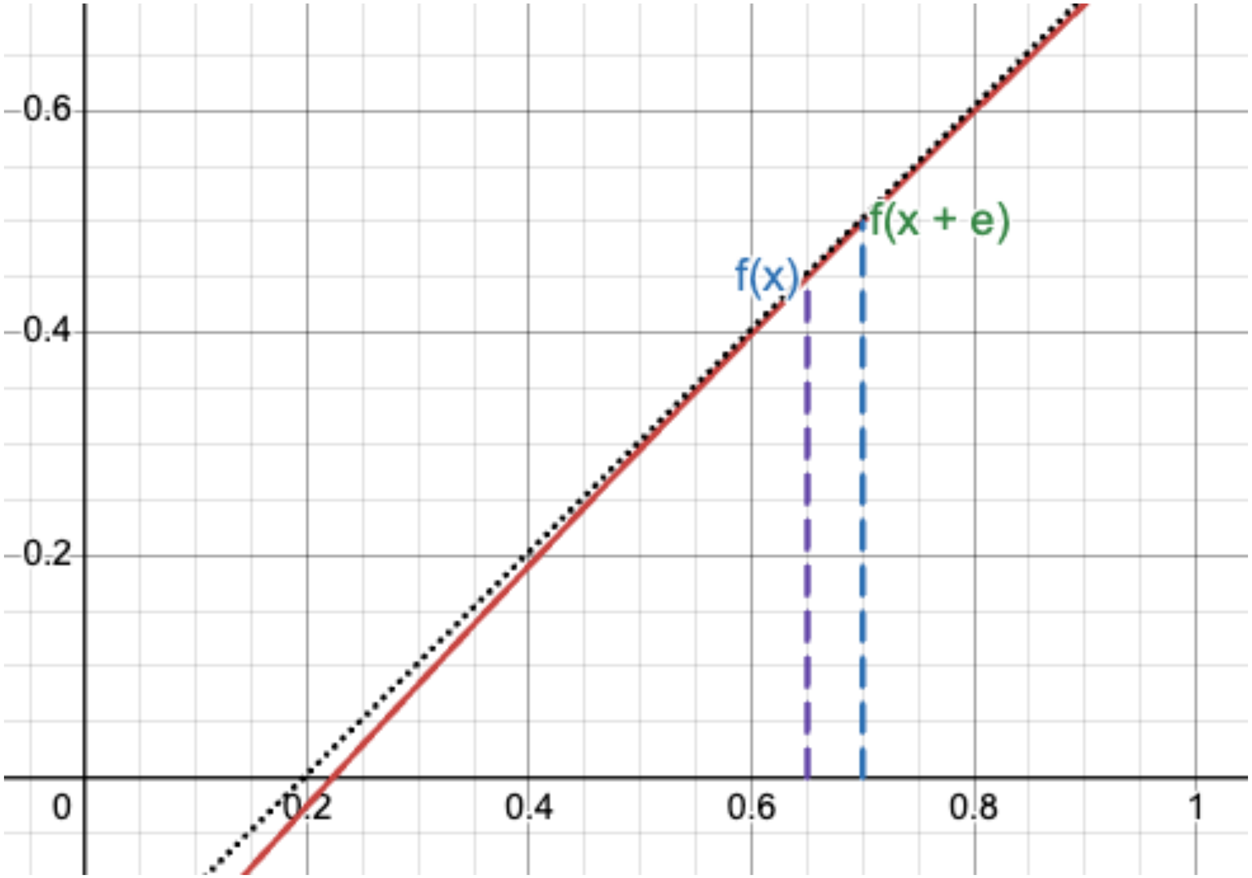
c1 = max number of iterations
c2 = step size smaller than the tolerance

Learning rate

- Most important hyper-parameter
- Scales the gradient and control the step size
- Difference between convergence and divergence



Numerical differentiation



$$slope = \frac{f(x+\epsilon) - f(x)}{\epsilon}$$

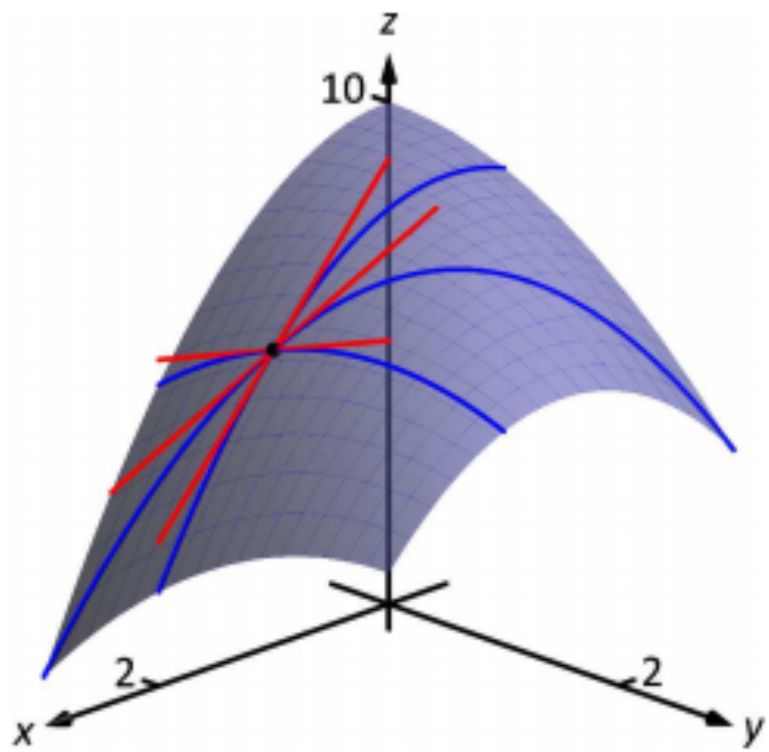
$$\frac{df(x)}{dx} = lim_{\epsilon \to 0} \frac{f(x+\epsilon) - f(x)}{\epsilon}$$

$$\frac{f(x+\epsilon) - f(x-\epsilon)}{2\epsilon}$$ central difference

Multivariate

$z = f(x, y)$

$$\nabla f(x, y) = \begin{pmatrix} \dfrac{\partial f(x, y)}{\partial x} \\ \dfrac{\partial f(x, y)}{\partial y} \end{pmatrix}$$



@https://math.libretexts.org

LAB SESSION

- Implement Gradient Descent for any multivariate function (NO NUMPY)
- Start with pen and paper. Discuss in pairs.
  - Algorithm definition
  - Function design
- Continue alone:
  - Code the algorithm and the main file.

LAB SESSION

- API Description:
  - Function hardcoded
  - Return the final solution (i.e. the point in which the algorithm stopped) and the summary of the gradient descent algorithm (i.e., final function value)
  - Print to screen the results

LAB SESSION

- Examples of functions to test:

$$f(x) = -2x + 5$$

$$f(x, y) = (x - 3)^2 + (y + 2)^2$$

$$f(x, y, z) = -cos(x) - cos(y) - cos(z) + \frac{1}{10}(x^2 + y^2 + z^2)$$

- But of course it will work for any function ;)

LAB SESSION

- Return:
  - 1 python script "gd.py" containing the main algorithm
  - 1 python script "main.py" containing the function definition and the program entry point
  - 1 text file "yourname_gd.txt" containing a brief description of your implementation choices.
- Pack everything in a zip archive "yourname_gd.zip" and upload on Moodle (alt. email)
- Deadline: 16-9-2025 08:00 AM