

W03 - Part 3

Recap:

$$\hat{P}_{\underline{w}}(y_n | \underline{x}_n) = \sigma(y_n \underline{w}^\top \underline{x}_n)$$

$$e_n(\underline{w}) = -\log \hat{P}_{\underline{w}}(y_n | \underline{x}_n) = \log \left(1 + e^{-y_n \underline{w}^\top \underline{x}_n} \right)$$

how to minimize E_{in} ?

To day:

Gradient descent.

(For now, forget about logistic regression and machine learning. To day, we want to learn how to minimize a function in general)

We want to predict with randomness

$$\hat{P}_w(y|x) = \frac{1}{1 + e^{-y_w^T x}}$$

We saw why it makes sense

We use log-loss

$$e_n(w) = -\log \hat{P}_w(y_n|x_n)$$

We have a learning Model

We need a learning algorithm.
How to find $\min E_n(w)$?

■ In linear regression, we found a closed form for the minimizer of E_{in} .

↳ Oh, how beautiful life was

■ Unfortunately, we don't have the closed form solution for E_{in} of logistic regression.

■ What should we do then?

↳ Numerical methods to minimize E_{in} .

■ We will use "Gradient descent". A well-known, and widely used numerical method.

Note: For now, forget about E_{in} , logistic Regression
and Machine Learning.

For one lecture, we study optimization and
Gradient descent.

Next week, we'll get back to logistic regression
and how to vs gradient descent for it.

Gradient descent

Given differentiable function $f: \mathbb{R}^n \rightarrow \mathbb{R}$,

Want to Solve: $\min_{x \in \mathbb{R}^n} f(x)$

For difficult problems, we do not have a closed-form Solution.

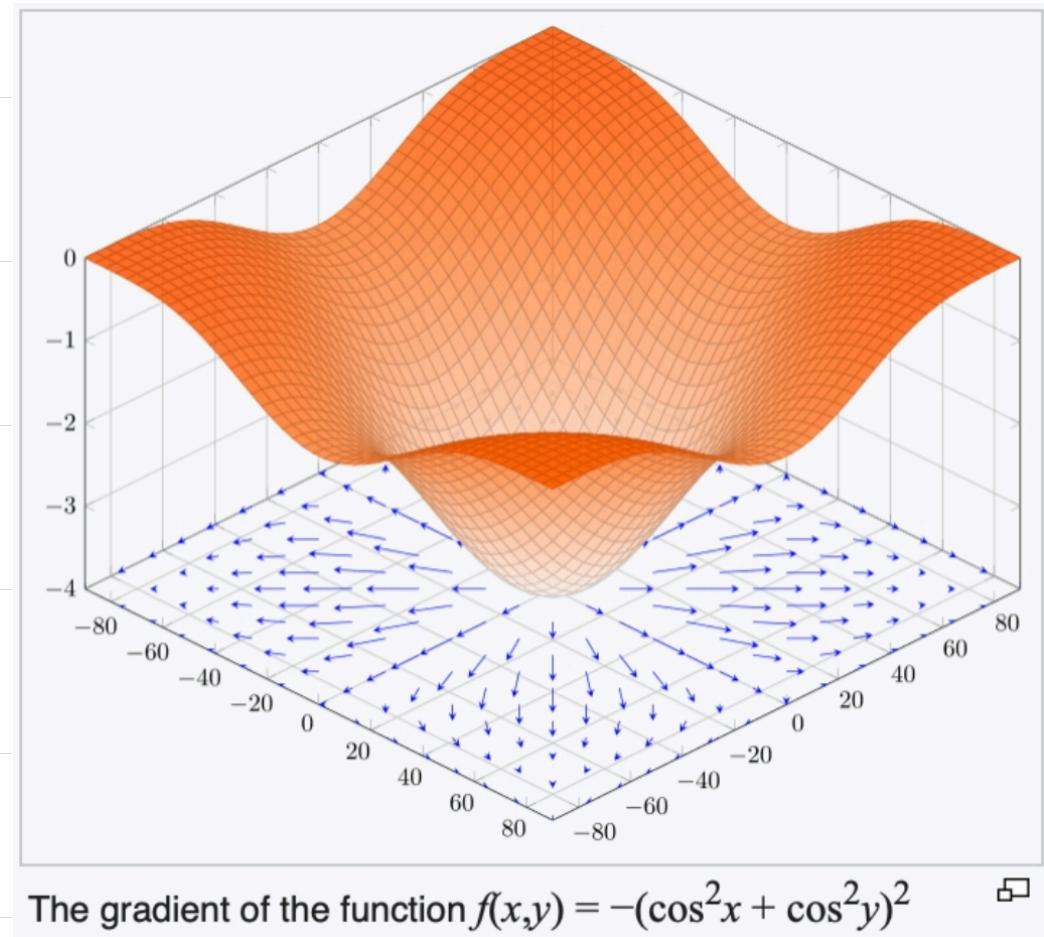
How Can We Solve it numerically?

Optimization Primer

1) Gradient: For $f: \mathbb{R}^n \rightarrow \mathbb{R}$,
its gradient $\nabla f: \mathbb{R}^n \rightarrow \mathbb{R}^n$
is defined as

$$\nabla f(\underline{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\underline{x}) \\ \frac{\partial f}{\partial x_2}(\underline{x}) \\ \vdots \\ \frac{\partial f}{\partial x_n}(\underline{x}) \end{bmatrix}$$

E.g.

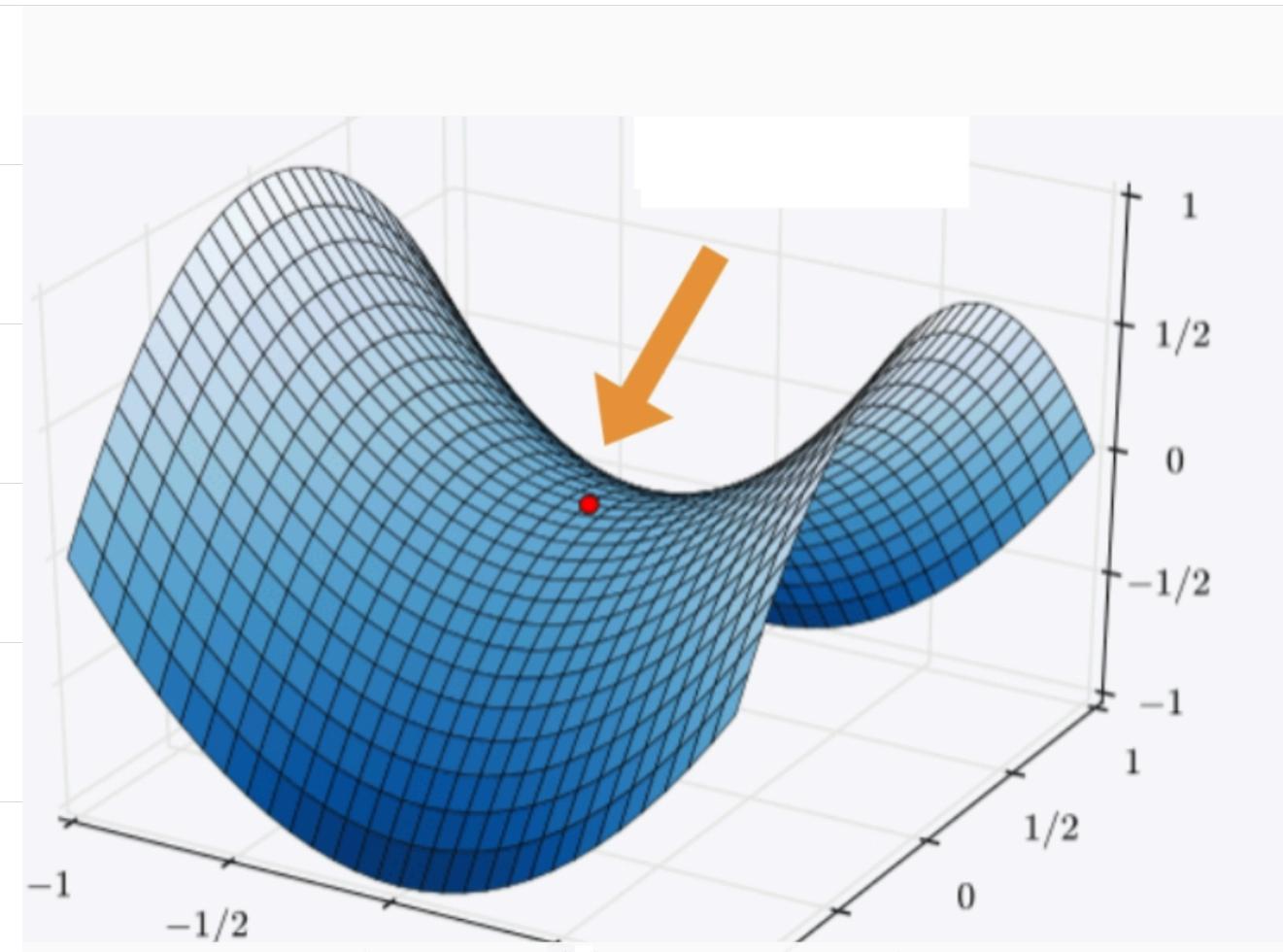
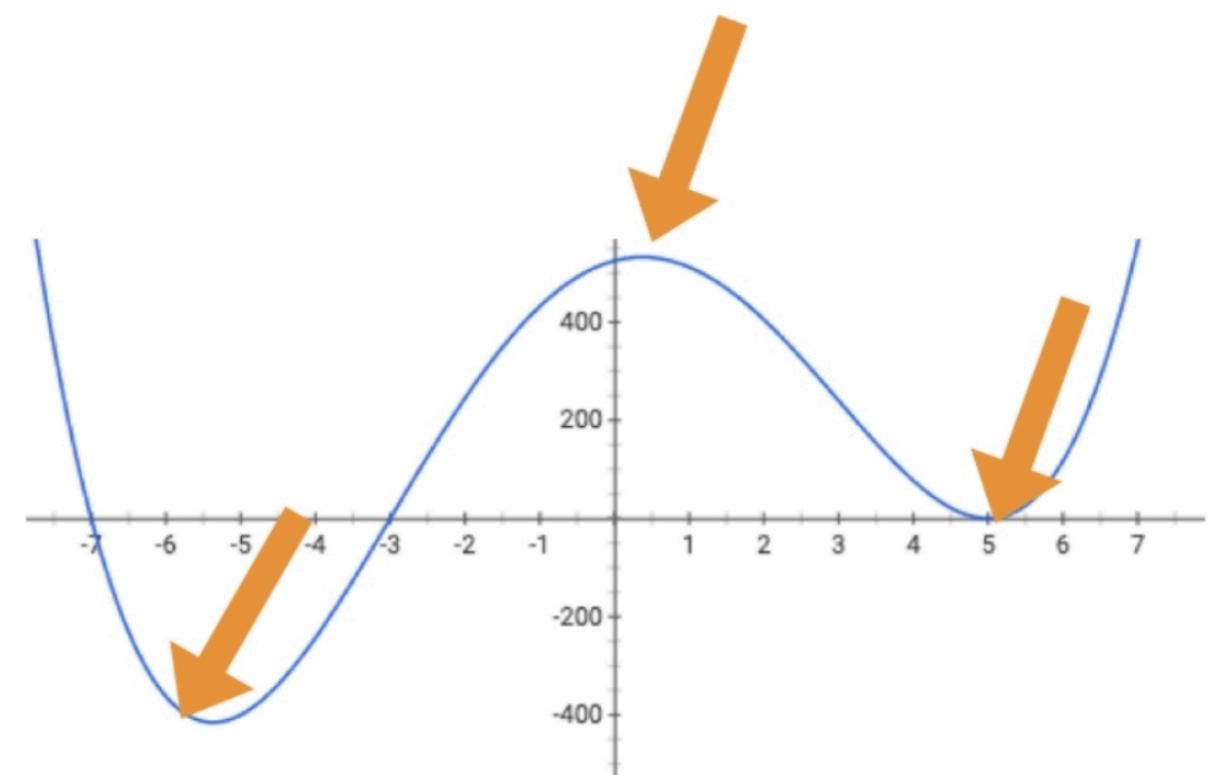


The gradient of the function $f(x,y) = -(\cos^2 x + \cos^2 y)^2$

E.g.

2) Gradient and Optimality:

When $\nabla f(x^*) = \underline{0}$, x^* can be any of 5 cases:



3) Convex Functions:

Defn: A function f on \mathbb{R}^m is convex if any line segment connecting two points of the graph of f lies above or on the graph.

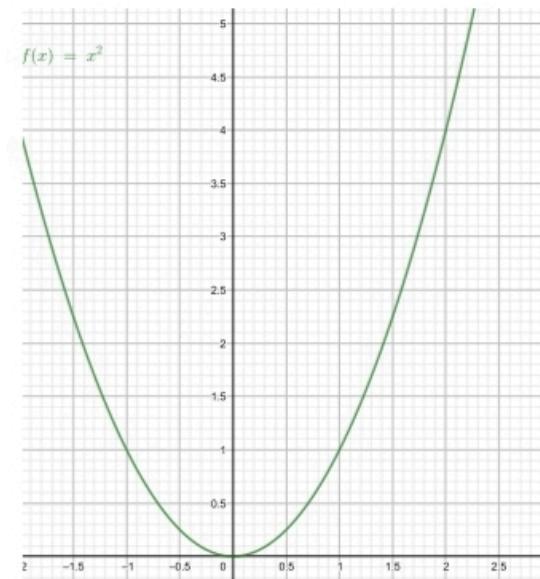
Defn: f is concave if $-f$ is convex.

Remark: For convex functions, local minima are all global minima.

<demo>

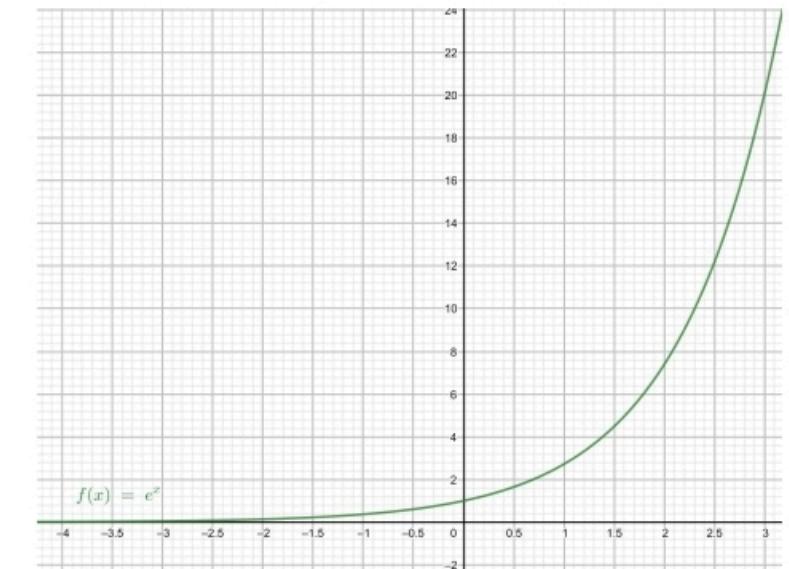
Let's See Some Simple Examples:

Convex

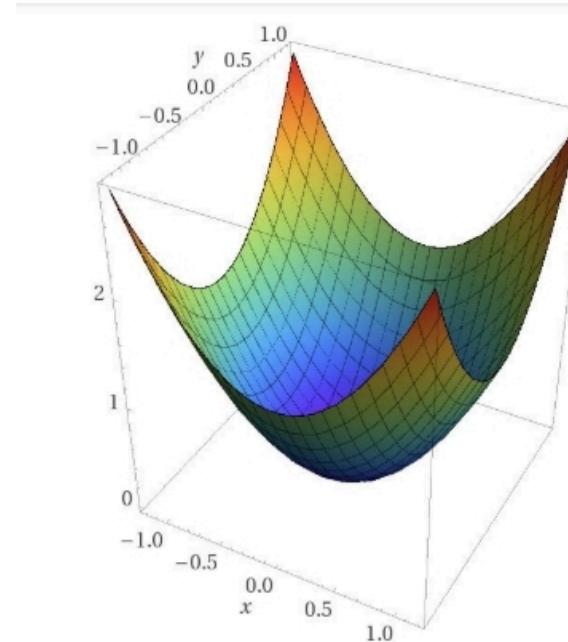


Non-Convex

Convex

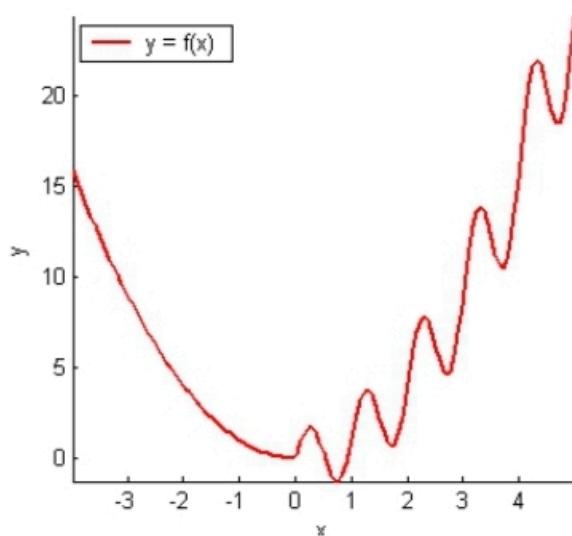


non-Convex



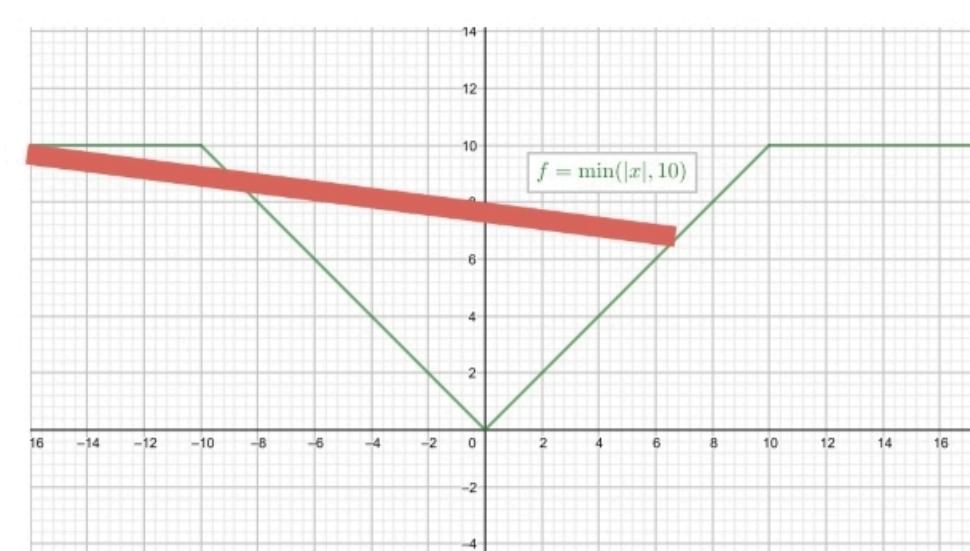
Convex

non-Convex



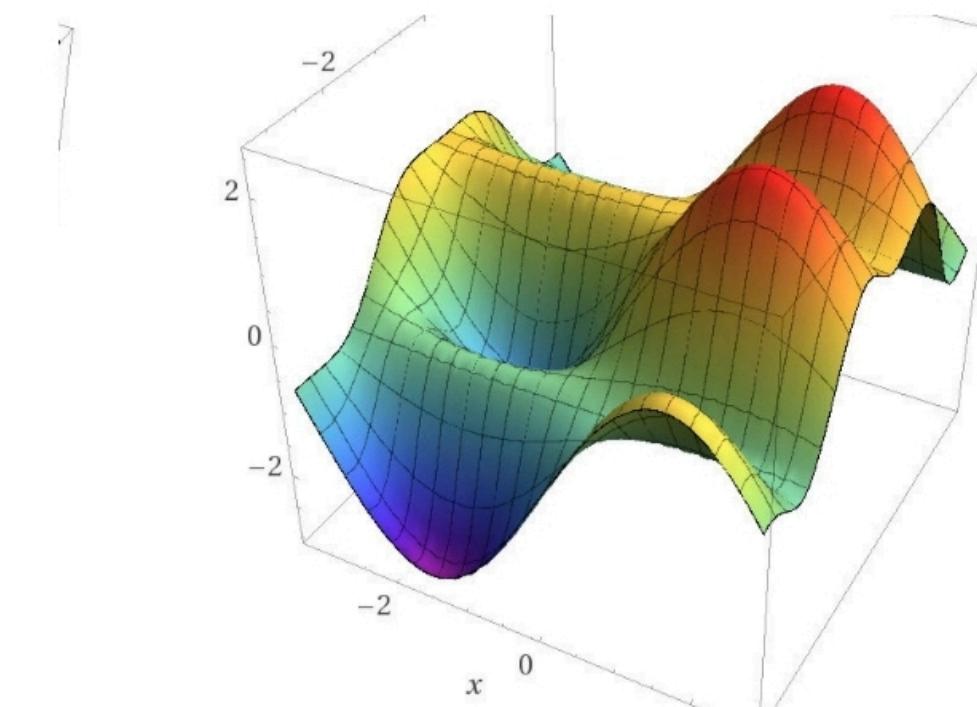
Convex

Non-Convex



Convex

Non-Convex



Convex

Non-Convex

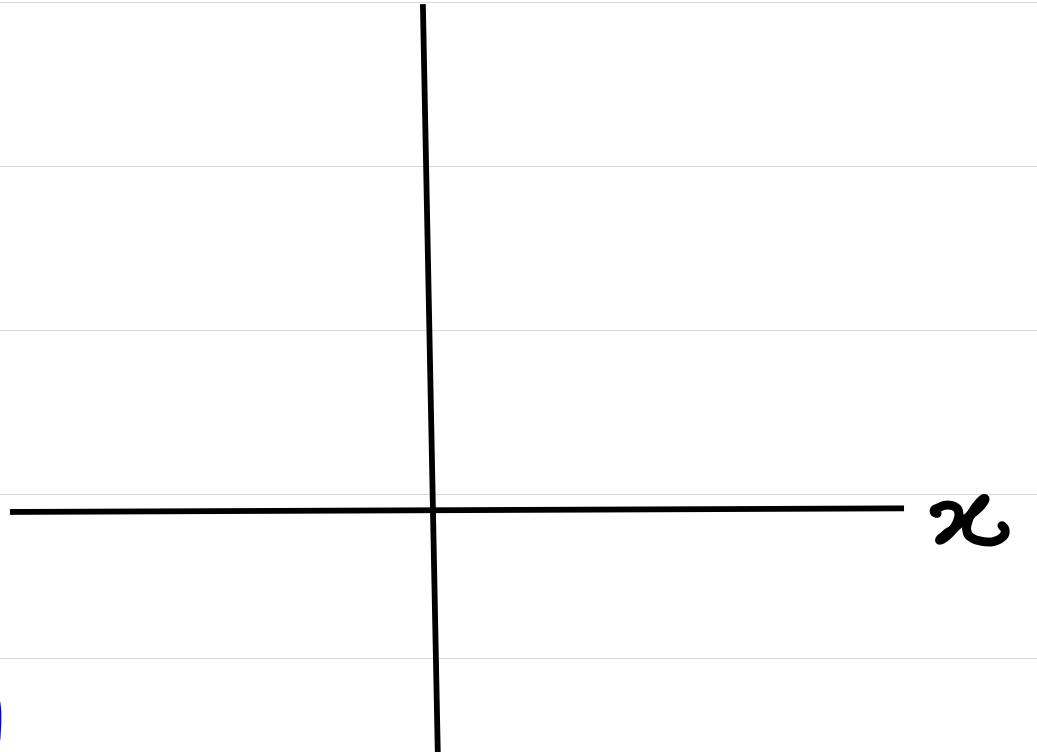
Let's study a 1-dimensional space, i.e., $f: \mathbb{R} \rightarrow \mathbb{R}$.

◻ $f'(x) = \frac{df}{dx}(x)$

◻ $x=x^* : f'(x)=0$

• what does it mean? x is optimal

◻ $x > x^*, f'(x) > 0$. What does it mean?



◻ $x < x^*, f'(x) < 0$. What does it mean?

A simple algorithm to find $\min f(x)$ (for 1-dimensional)

1. Initialize $x = x_0$.
2. If $f'(x) \approx 0$, then Stop
3. If $f'(x) > 0$, then $x = x - \epsilon$
4. If $f'(x) < 0$, then $x = x + \epsilon$
5. Go to step 2.

ϵ is the step size.

■ What should be the value of ϵ ?

↳ What happen if we use a large ϵ ?

↳ What happen if we use a small ϵ ?

■ What should be the value of step size?

□ large ϵ is bad. Small ϵ is bad. So, medium ϵ is good.

□ what value would be medium?

■ May be it's better to use time varying step size, ϵ_t .

□ "t" is the iteration number.

■ How should we change ϵ_t over time?

■ increase ϵ_t or decrease ϵ_t ?

□ start with small step size and increase it over time

OR

□ Start with large step size and decrease over time

■ intuition:

E.g. $\epsilon_t = \frac{1}{t}$

One can prove that with $\epsilon_t = \frac{1}{t}$ if the function is strongly convex (e.g. $f''(x) > 0 \forall x \in \mathbb{R}$) then we are guaranteed to converge to the optimal solution.

■ Let's study n-dimensional Space.

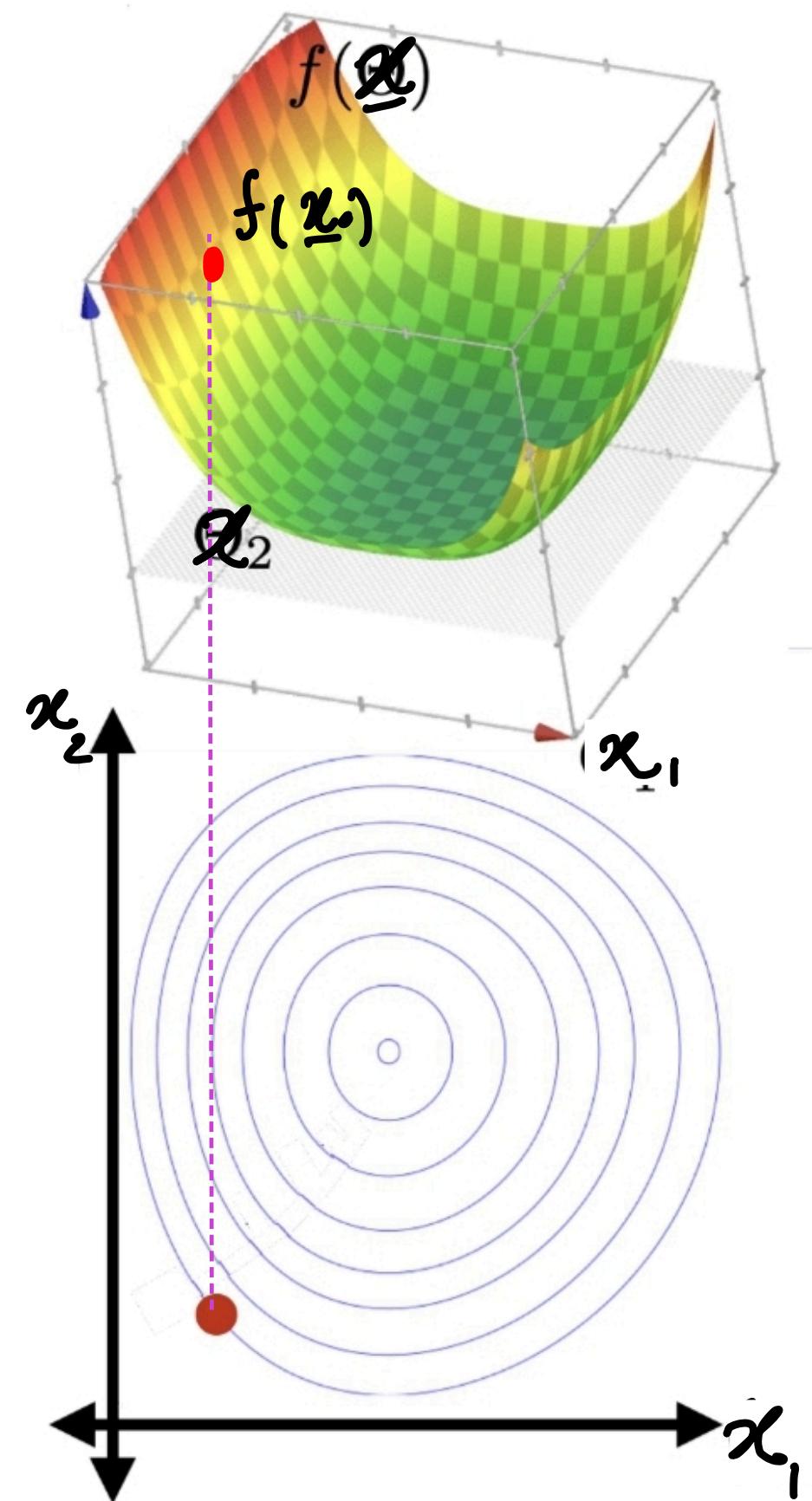
■ Given the current location \underline{x} , what should be the next stop?

$$\underline{x}' \leftarrow \underline{x} + \epsilon \underline{u}$$

■ ϵ : step size, \underline{u} : direction of update, $\|\underline{u}\|=1$.

Let's make a 2-D illustration.

Where should we go from \underline{x}_0 ?



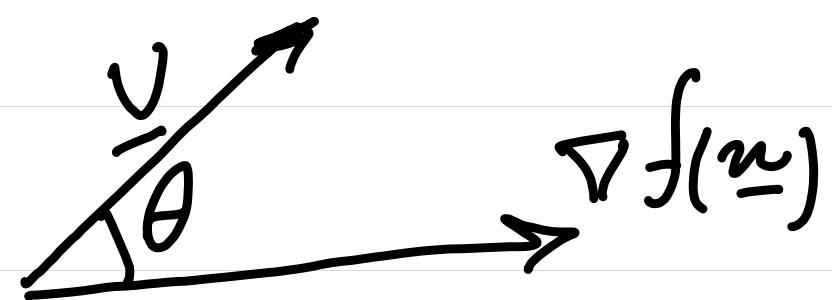
■ By taylor Series :

$$f(\underline{x} + \varepsilon \underline{v}) = f(\underline{x}) + \varepsilon \underline{v}^T \nabla f(\underline{x}) + O(\varepsilon^2 \|\underline{v}\|^2)$$

negligible

■ So, we want to minimize $\underline{v}^T \nabla f(\underline{x})$.

$$\square \quad \underline{v}^T \nabla f(\underline{x}) = \|\underline{v}\| \cdot \|\nabla f(\underline{x})\| \cdot \cos \theta$$



□ $\underline{v}^T \nabla f(\underline{x})$ is minimized when

■ So, $\underline{v}^* =$

Note:

1- From the analysis in last page, $\nabla f(\underline{x})$ points in the direction where $f(\underline{x})$ has maximum ascent.

2- Observe that $\epsilon \underline{v}^* = -\frac{\epsilon}{\|\nabla f(\underline{w})\|} \nabla f(\underline{w})$

most often we combine ϵ and $\|\nabla f(\underline{w})\|$ and call this fraction the step size ϵ_t .

Gradient descent Alg.

Initialize \underline{x}_0 . (typically at random)

For $t=0, 1, 2, \dots$

Compute $\underline{g}_t = \nabla f(\underline{x}_t)$

Select direction $\underline{v}_t = -\underline{g}_t$

update $\underline{x}_{t+1} = \underline{x}_t + \epsilon_t \underline{v}_t$

Stop if stopping Criteria are reached.

End for

Note:

① Condition for stopping: $\nabla f(\underline{x}_t) \approx 0$ is reasonable.

② ϵ_t : "learning rate"

③ If we have a constant learning rate for Gradient descent Alg.,
(i.e., $\epsilon_t = \alpha$ $\forall t$), then \underline{v}^* with $\|\underline{v}^*\| = 1$

$$\epsilon_t \underline{v}_t = -\alpha \underline{g}_t = -\alpha \nabla f(\underline{x}) = -\alpha \|\nabla f(\underline{x})\| \cdot \frac{\nabla f(\underline{x})}{\|\nabla f(\underline{x})\|}$$

Most often in ML programming

libraries (like PyTorch),

ϵ_t is fixed.

actual step size diminishes
as $\underline{x}_t \rightarrow \underline{x}^*$

(demo)

z_c

Theorem: Gradient descent performance

Assumptions: (choose any $\gamma > 0$). if

- 1) f is sufficiently "smooth" and convex
- 2) f has at least one global optimum
- 3) η is sufficiently small

Conclusion: If run long enough, gradient descent will return a value within γ of a global optimum.