W03 - Part 3

Recap:

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

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

how to minimize E_{in}?

Today:

Gradient descent.

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

We want to predict with randomness

$\mathcal{H}$: $\hat{P}_{\underline{w}}(y|\underline{x}) = \dfrac{1}{1+e^{-y\underline{w}^T\underline{x}}}$

We saw why it makes sense

We use log-loss

$e_n(w) = -\log \hat{P}_{\underline{w}}(y_n|\underline{x}_n)$

We have a learning Model

we need a learing algorithm.
How to find min $E_{in}(\underline{w})$?

- 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 us gradient descent for it.

Gradient descent

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

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

- For difficult problems, we do not have a closed-form Solution.
- How can we solve it numerically?

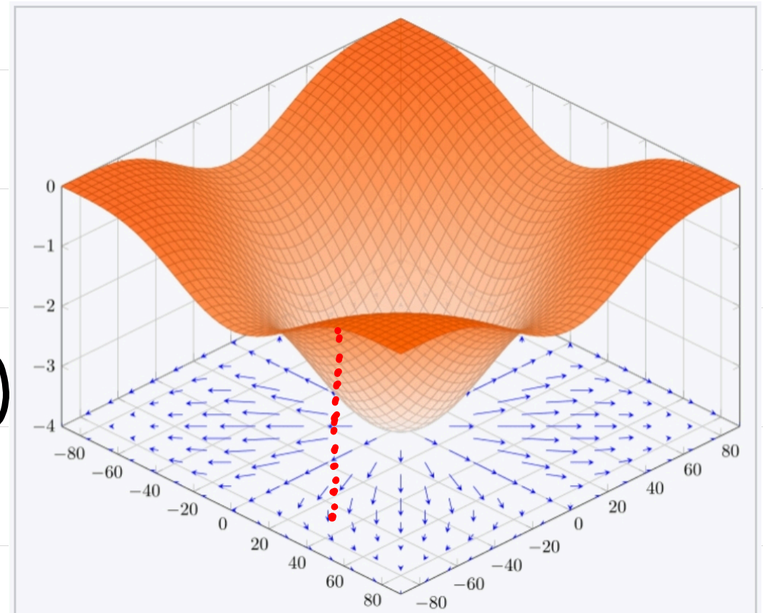
Optimization Primer

1) Gradient: For $f: \mathbb{R}^n \to \mathbb{R}$, its gradient $\nabla f: \mathbb{R}^n \to \mathbb{R}^n$ 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 is close to 0, when $f(\underline{x})$ has small variations.



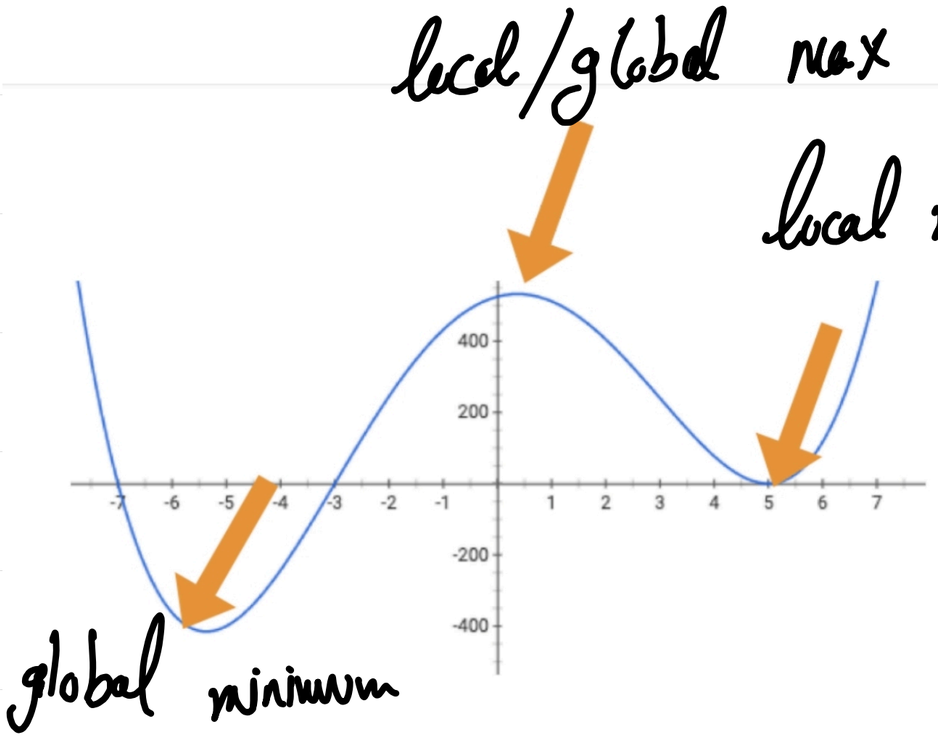
The gradient of the function $f(x,y) = -(\cos^2 x + \cos^2 y)^2$

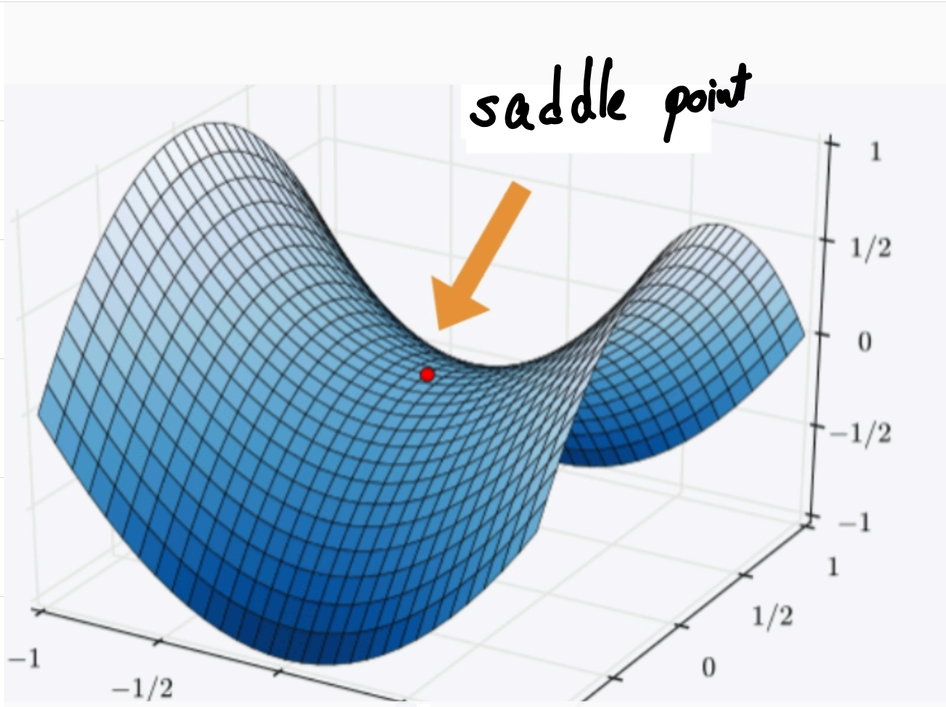
E.g. $f(x,y,z) = x^2 + y^3 + z$

$$\nabla f(x,y,z) = \begin{bmatrix} 2x \\ 3y^2 \\ 1 \end{bmatrix}$$

2) Gradient and Optimality:

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





3) Convex Functions:

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

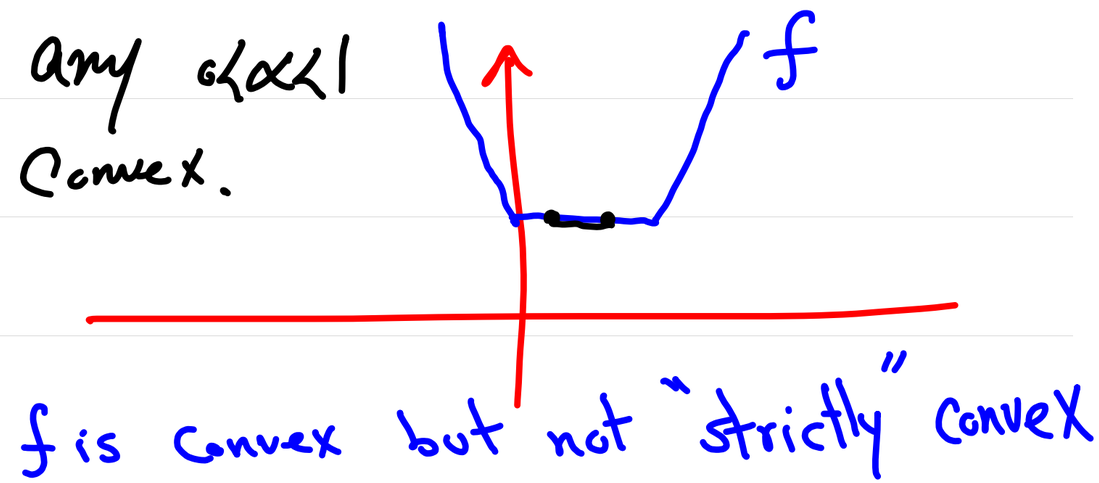
for any $0 \leq \alpha \leq 1$ and any $\underline{x}_1$ and $\underline{x}_2$, $f(\alpha \underline{x}_1 + (1-\alpha)\underline{x}_2) \leq \alpha f(\underline{x}_1) + (1-\alpha) f(\underline{x}_2)$

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

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

Defn: If $f(\alpha \underline{x}_1 + (1-\alpha)\underline{x}_2) < \alpha \underline{x}_1 + (1-\alpha)\underline{x}_2$ for any $0<\alpha<1$ and any $\underline{x}_1, \underline{x}_2$, then f is "strictly" convex.

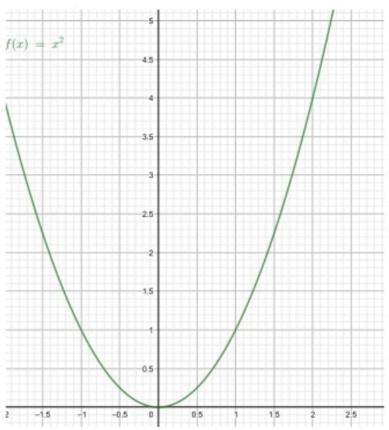
Remark: For strictly convex functions, there is only one global optimum.



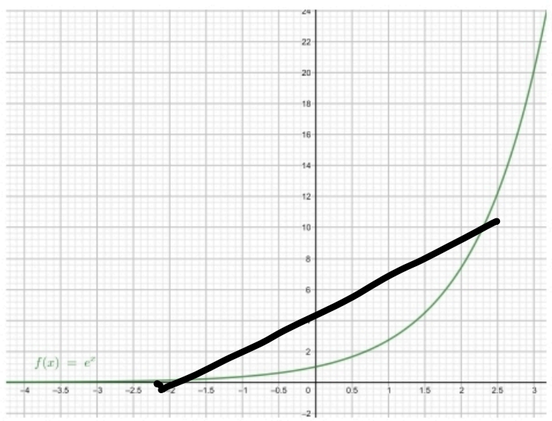
Let's See Some Simple examples:

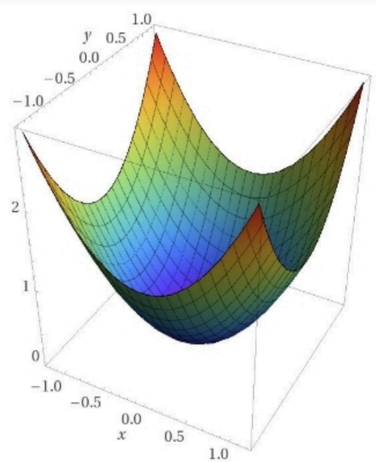
Convex ☑

Non-Conv ☐



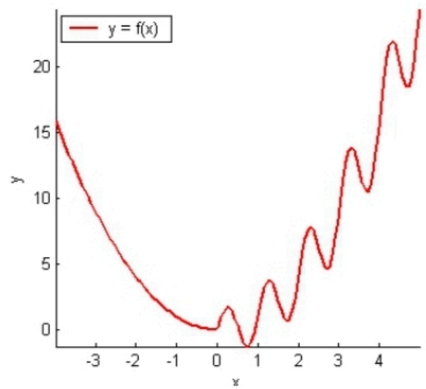
cOnvex ☑ non-Conv ☐



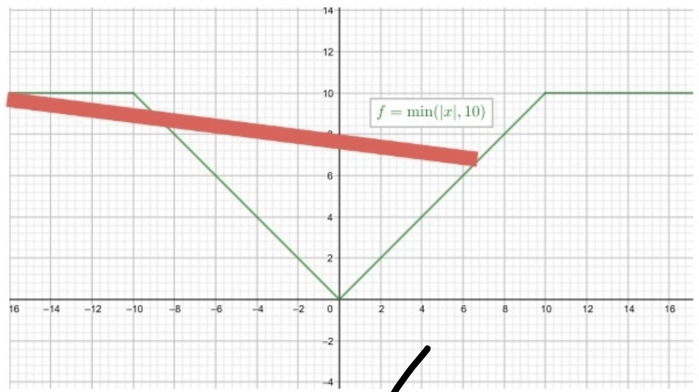


☑ Convex

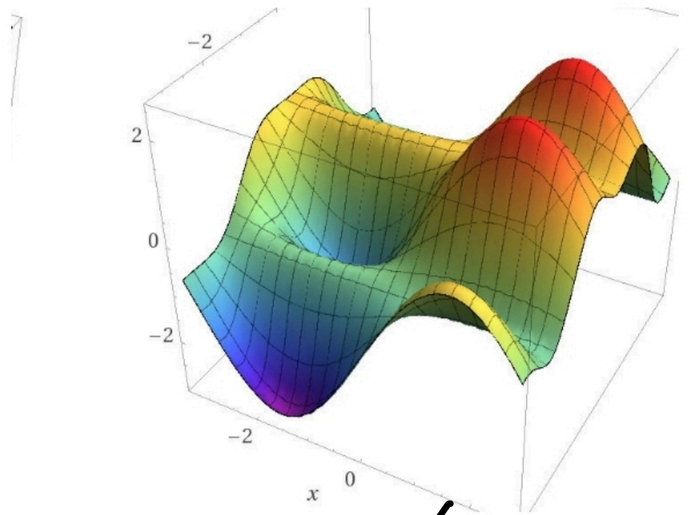
☐ non-Conv



☐ Convex ☑ Non-Conv



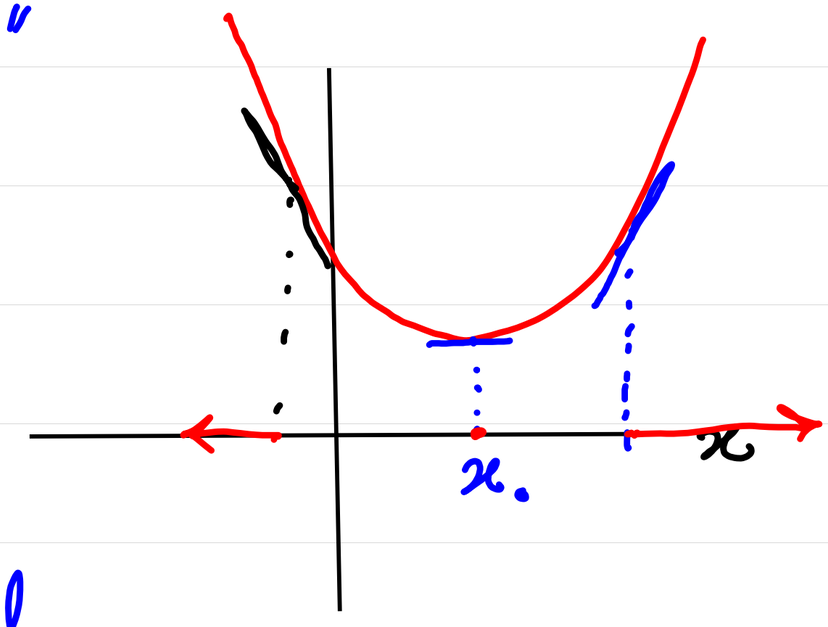
☐ Convex ☑ Non-Conv



☐ Convex ☑ non-Conv

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

"assume $f(x)$ is convex"

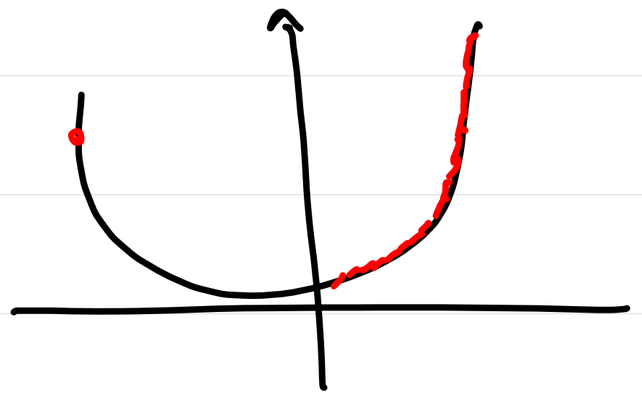


- $f'(x) = \frac{df}{dx}(x)$
- $x = x^*$: $f'(x) = 0$
  - what does it mean? x is optimal
- $x > x^*$, $f'(x) > 0$. What does it mean?
  - $f(x)$ increases as x increase
- $x < x^*$, $f'(x) < 0$. What does it mean?
  - $f(x)$ decreases as x increases.

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 $x = x - \varepsilon$
4. If $f'(x) < 0$, then $x = x + \varepsilon$
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?
    - Bounce around, specially close to optimal point.
  - What happen if we use a small ε?
    - It can take forever.

- 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?
    - a medium ε for $f(x) = x^2$, is very small step size for $g(x) = 10^3 x^{10}$

- 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: We want to get close to the optimal point. and when around the optimal point, we want to be more cautious (small step sizes)

- E.g. $\varepsilon_t = 1/t$

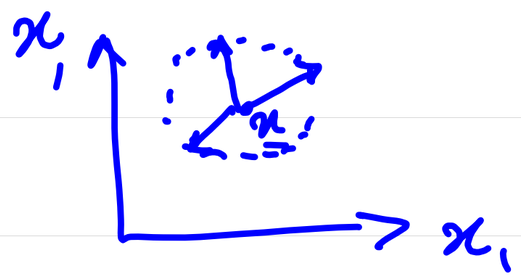
  - One can prove that with $\varepsilon_t = 1/t$ if the function is strongly convex (e.g. $f''(x) > 0 \quad \forall x \in \mathbb{R}$) then we are guaranteed to converge to the optimal solution.

- Let's study n-dimensional space.

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

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

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

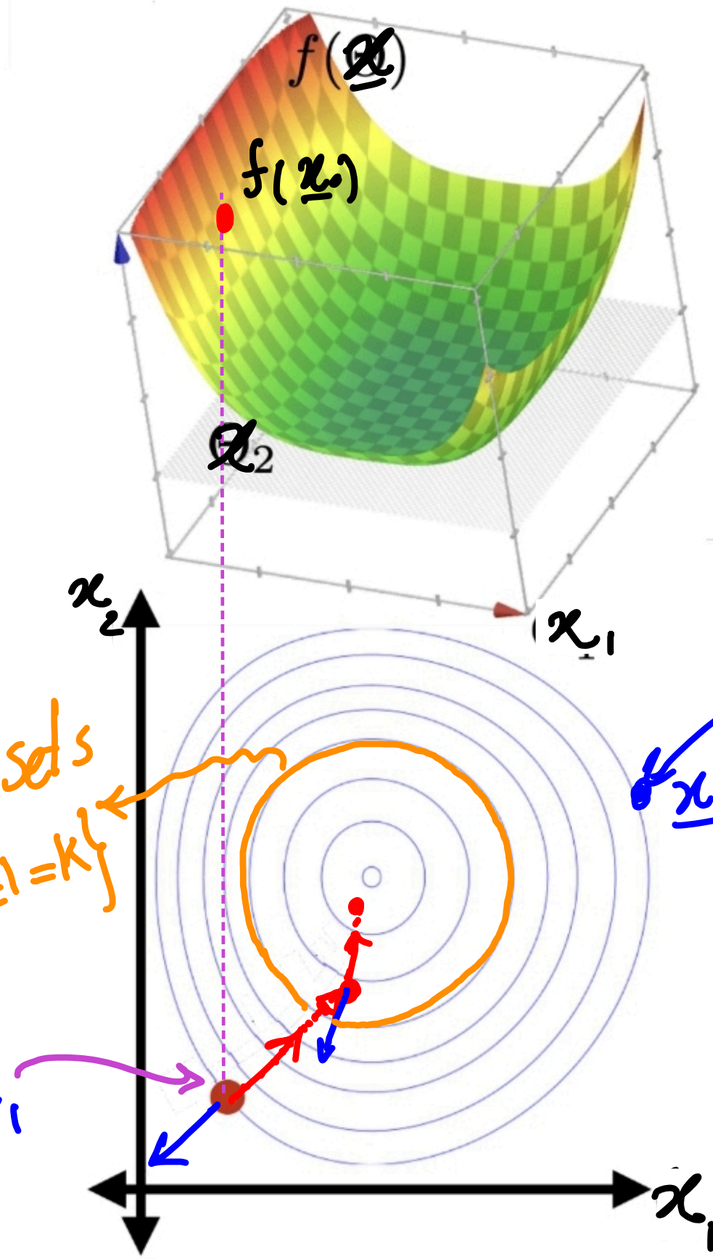


- Let's make a 2-D illustration.

- Where should we go from $\underline{x}_0$?
  - choose the direction s.t it give you the smalles $f(\underline{x}')$ (in a small step ε)

$$\min \quad f(\underline{x}') = f(\underline{x} + \varepsilon \underline{U})$$

where $\|\underline{U}\| = 1$.



- By taylor series :

negligible

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

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

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

  - $\underline{v}^T\nabla f(\underline{x})$ is minimized when $\cos\theta = -1 \Leftrightarrow \theta = 180^\circ$

- So, $\underline{v}^* = -\dfrac{\nabla f(\underline{x})}{\|\nabla f(\underline{x})\|}$

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 $\varepsilon \underline{v}^* = -\dfrac{\varepsilon}{\|\nabla f(\underline{x})\|} \nabla f(\underline{x})$

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

Gradient descent Alg.

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

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

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

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

update $\underline{x}_{t+1} = \underline{x}_t + \varepsilon_t \underline{V}_t$

Stop if stopping criteria are reached.

End for

Note:

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

② ε_t: "learning rate"

③ If we have a Constant learning rate for Gradient descent Alg., (ie, $\varepsilon_t = \alpha \quad \forall t$), then

$\underline{U}^*$ with $\|U^*\|=1$

$$\varepsilon_t \underline{V}_t = -\alpha \underline{g}_t = -\alpha \nabla f(\underline{x}) = -\alpha \|\nabla f(\underline{x})\| \frac{\nabla f(\underline{x})}{\|\nabla f(\underline{x})\|}$$

Most often in ML programming libraries (like PyTorch), ε_t is fixed.

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

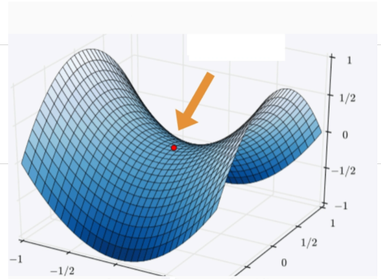
<demo>

Theorem: Gradient descent performance

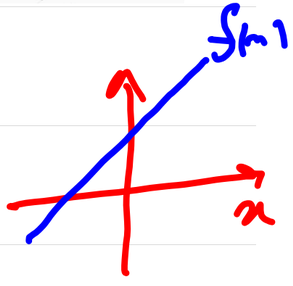
Assumptions: (choose any $\gamma > 0$). if

1) f is sufficiently "smooth" ← if not, can't run gradient descent

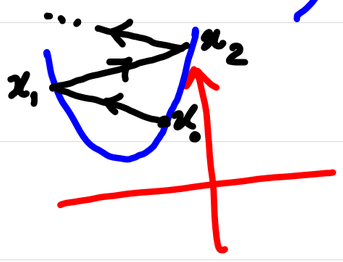
2) f is convex ← if not, may stuck in saddle point, e.g.



3) f has at least one global optimum
↖ If not, may not terminate, e.g.,



4) η is sufficiently small
↖ If not, may diverge, e.g.,



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