Continuous Optimization

Francesc, Mateo, Tomas, Izan, Eric

CONTENTS

- Continuous Optimization for Machine learning
    - Continuous Optimization
    - Gradient Descent
        - Step-Size
        - With Momentum
    - Stochastic Gradient Descent
    - Duality in Optimization
    - Constrained Optimization

Continuous Optimization for Machine learning

Training a machine learning model

How can we do it?

Find the best Parameters

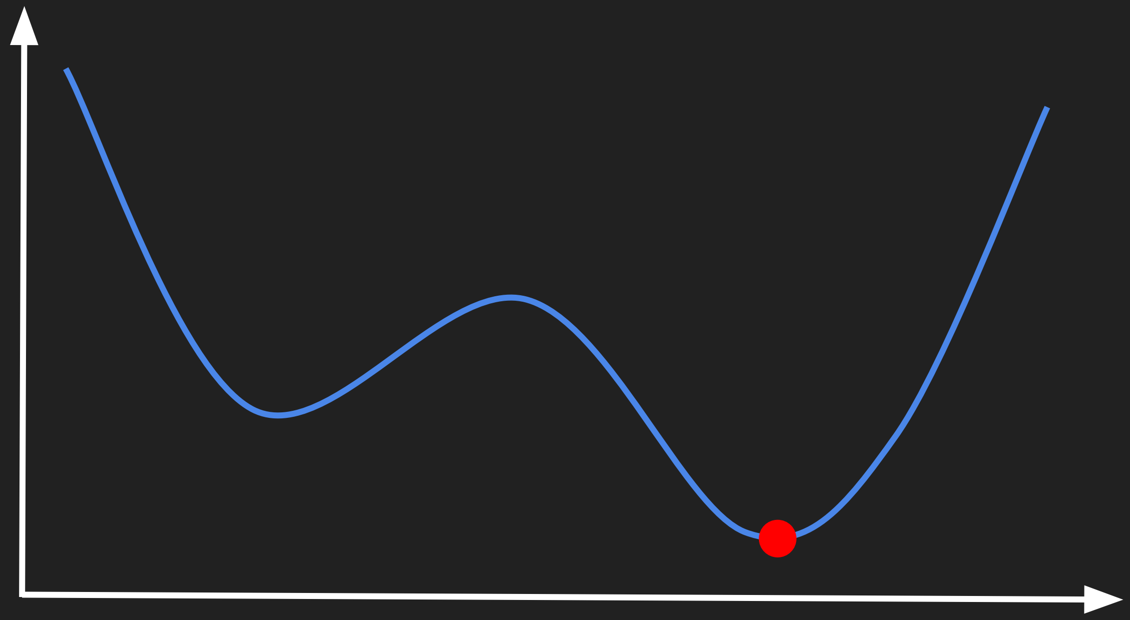
Continuous Optimization

Objective of Continuous Optimization

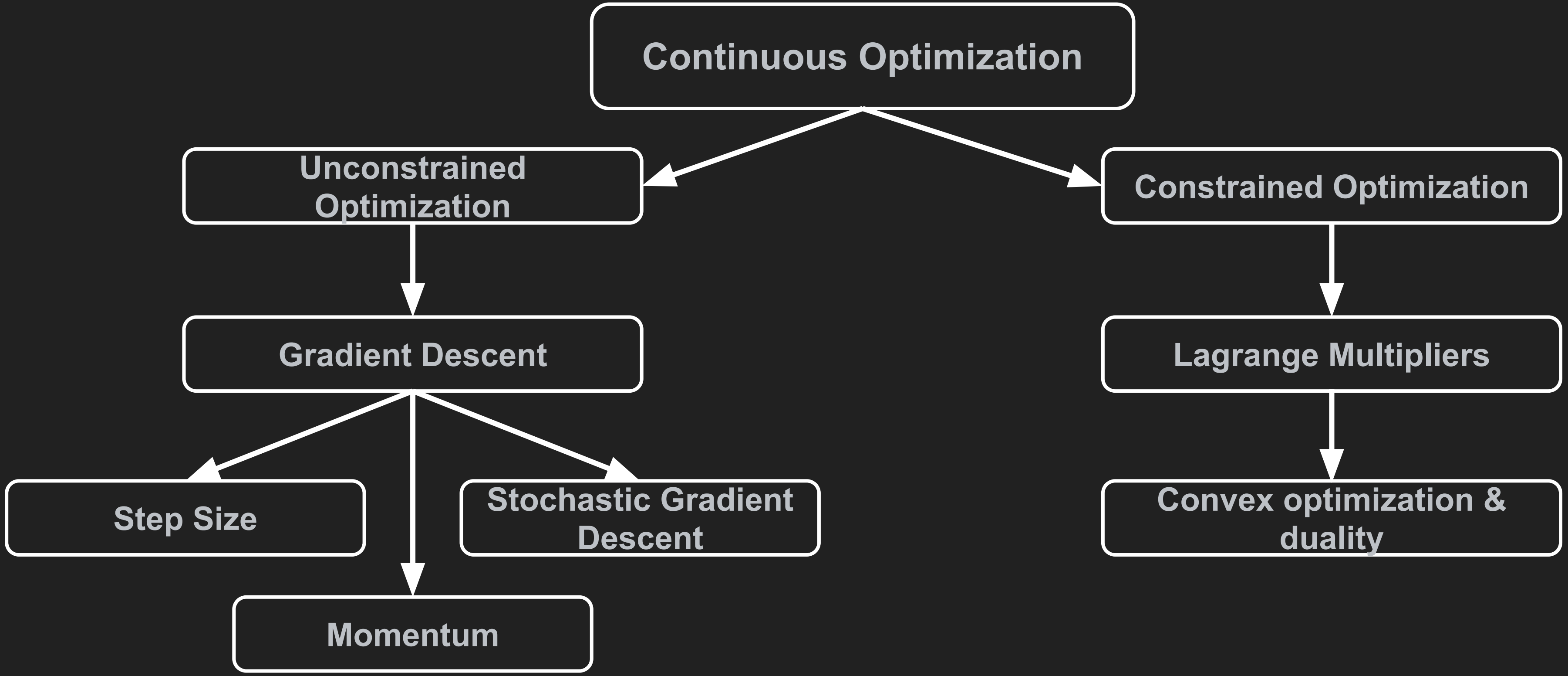
Find the best value

Minimise / maximise an objective function

By convention, objective functions in machine learning are minimized



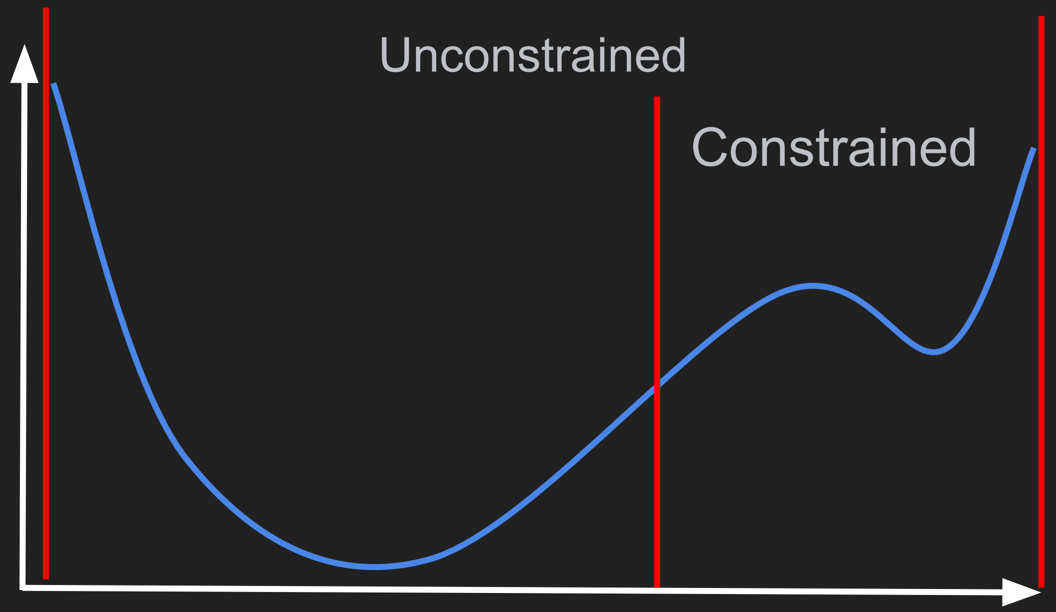
How can we reach the minimum value? | Main topics of Continuous Optimization



Unconstrained vs constrained optimization

Unconstrained: The variable can take on any value, there are no restrictions

Constrained: the variable can only take on certain values within a larger range

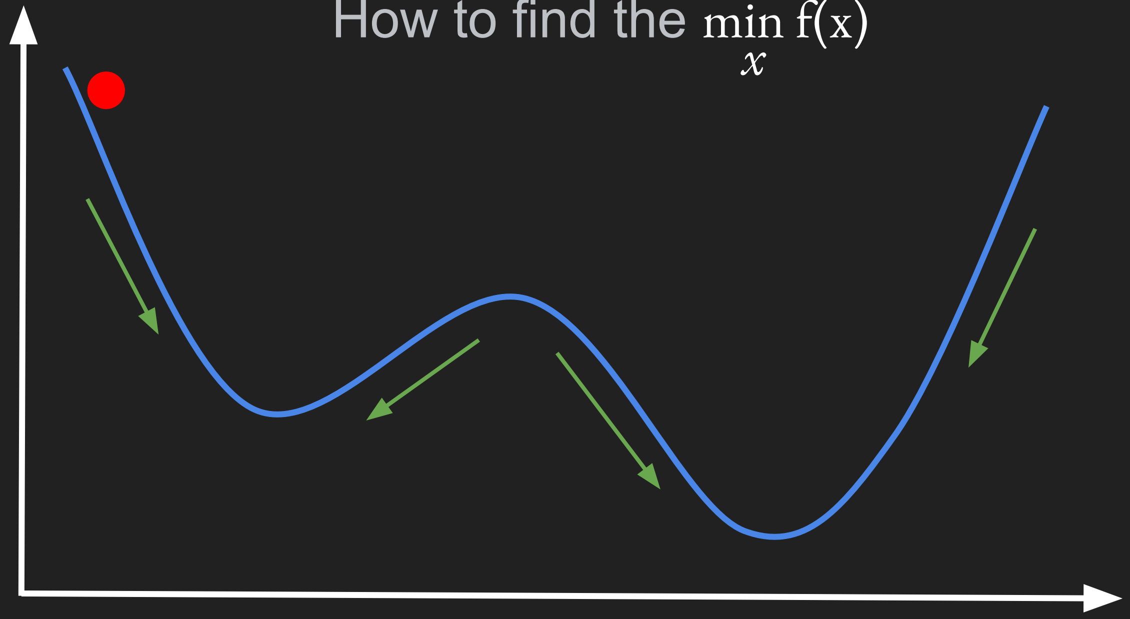


Gradient Descent

Constrained Optimization

Follow the negative gradient

How to find the $\min_{x} f(x)$



We only use one variable for simplicity

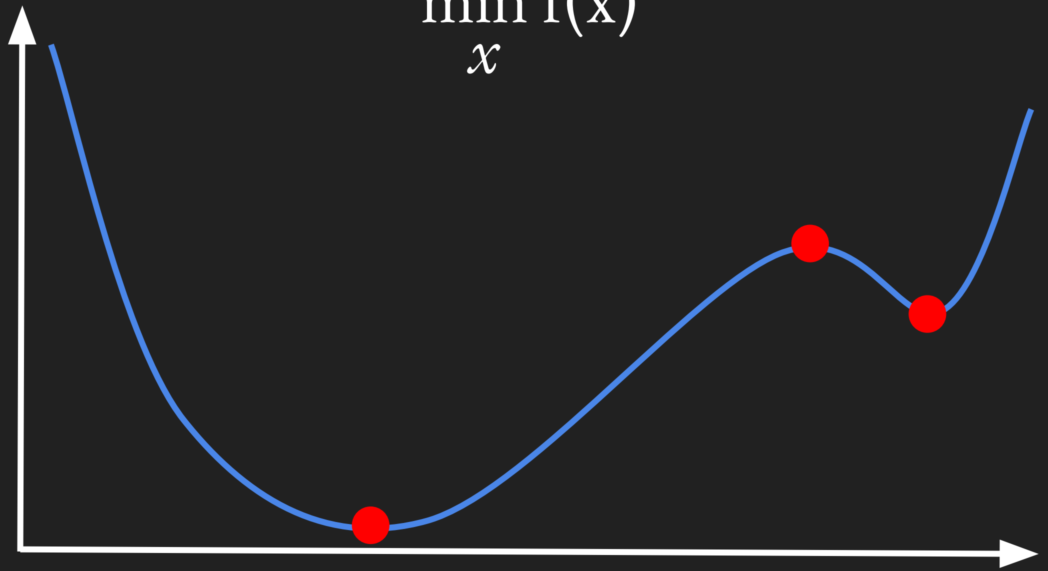
Why do we need Gradient Descent in first place?

Analytic Solutions

$$f(x) = x^3 + x^2 \ldots$$

$$\frac{df(x)}{d(x)} = 0$$

$$\min_{x} f(x)$$



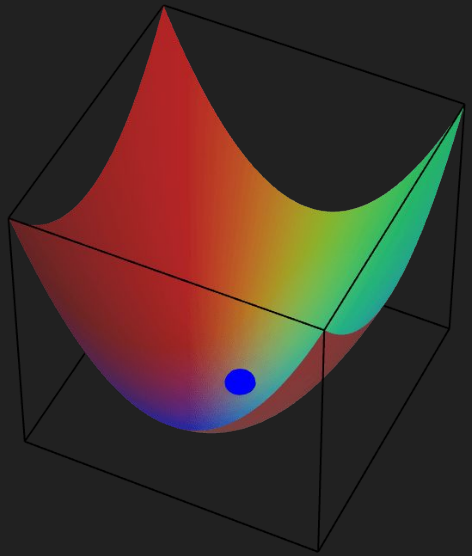
We only use one variable for simplicity

Why do we need Gradient Descent in first place?

In general, we are unable to find analytic solutions

Consider:

- When the training set is enormous
- When no simple formulas exist.



Imagine a function with a large number of variables

Gradient Descent

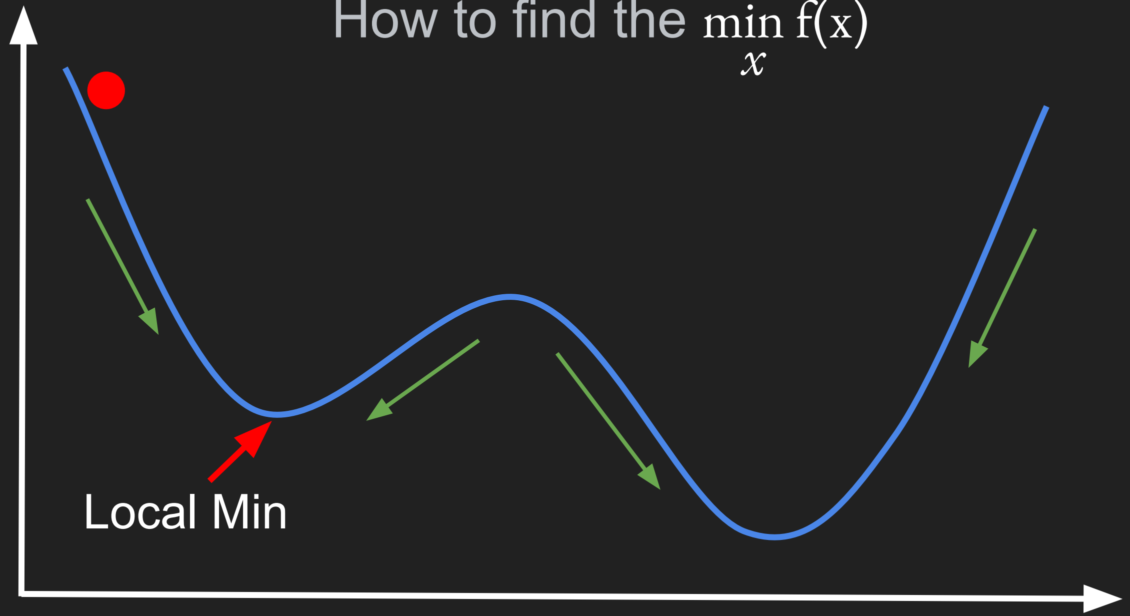
Unconstrained Optimization

Follow the negative gradient

Problems:

- False/Local Minimum
- The gradient indicates the direction but we don't know how to advance

How to find the $\min_{x} f(x)$



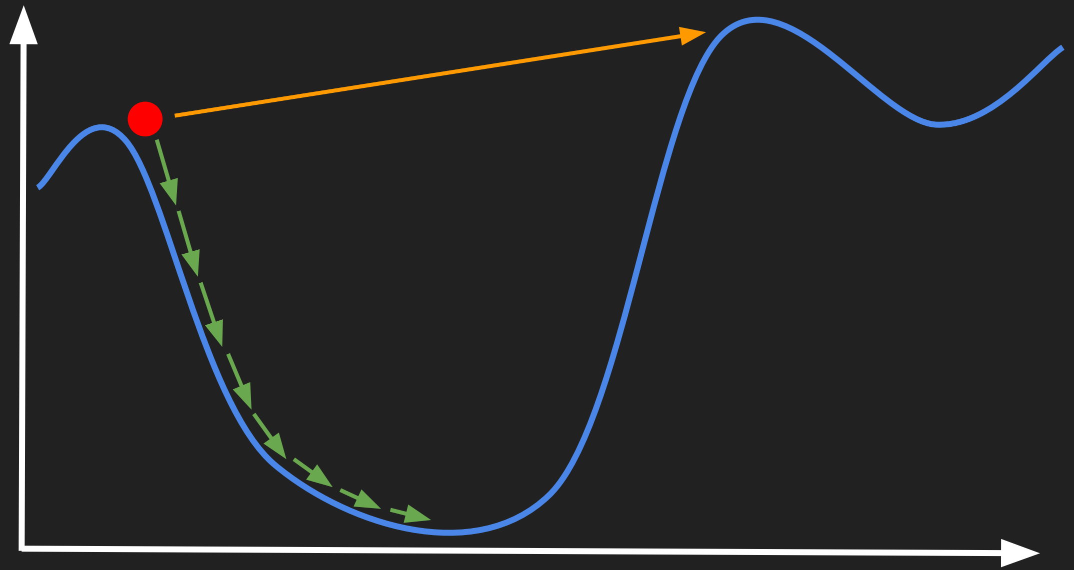
We only use one variable for simplicity

Gradient Descent: Step-Size

How to advance?

Choosing a good step-size is important

- Small → Slow
- Large → Overshoot



We only use one variable for simplicity

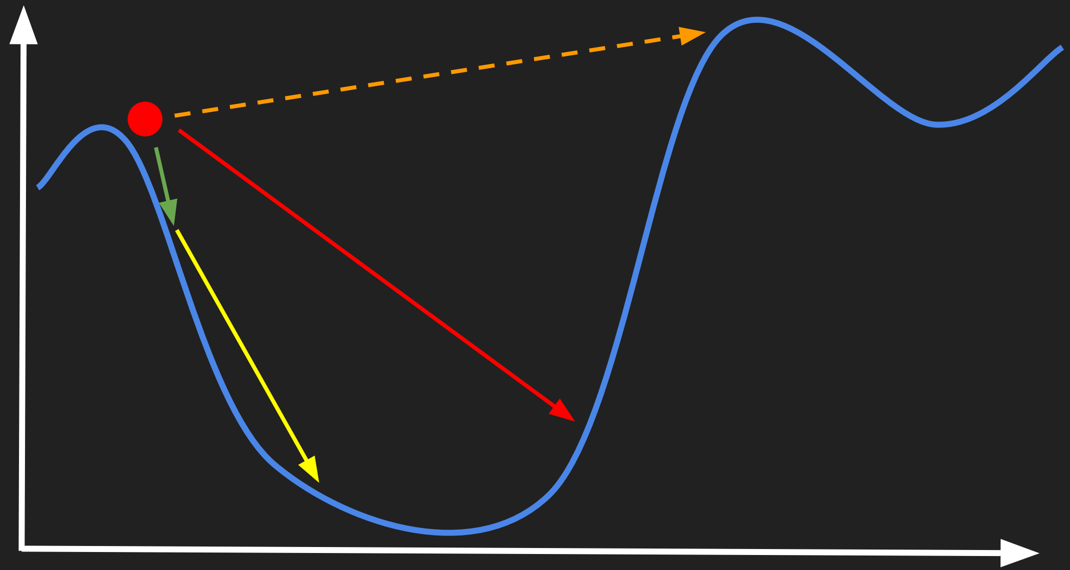
Gradient Descent: Step-Size

How to choose an optimal Step?

Two simple heuristics:

Value Decreases → Increase Step

Value Increases → Undo and Decrease Step



We only use one variable for simplicity

Gradient Descent

FORMULA: $x_{i+1} = x_i - \gamma_i \nabla f(x_i)$

What does this mean?

x_i	γ_i	$\nabla f(x_i)$
Initial parameter	Step Size	Gradient

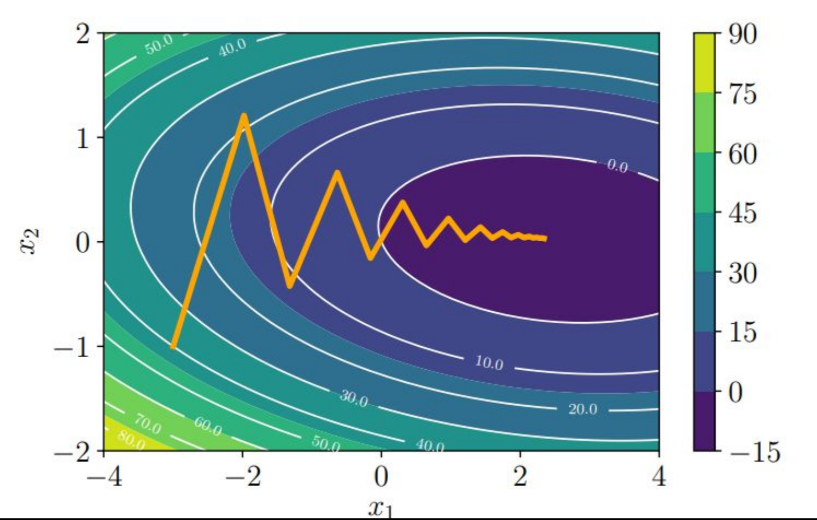
Gradient Descent

Example

$$f\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \frac{1}{2}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^\top \begin{bmatrix} 2 & 1 \\ 1 & 20 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} 5 \\ 3 \end{bmatrix}^\top \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\nabla f\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^\top \begin{bmatrix} 2 & 1 \\ 1 & 20 \end{bmatrix} - \begin{bmatrix} 5 \\ 3 \end{bmatrix}^\top$$

$$x_{i+1} = x_i - \gamma_i \nabla f(x_i)$$

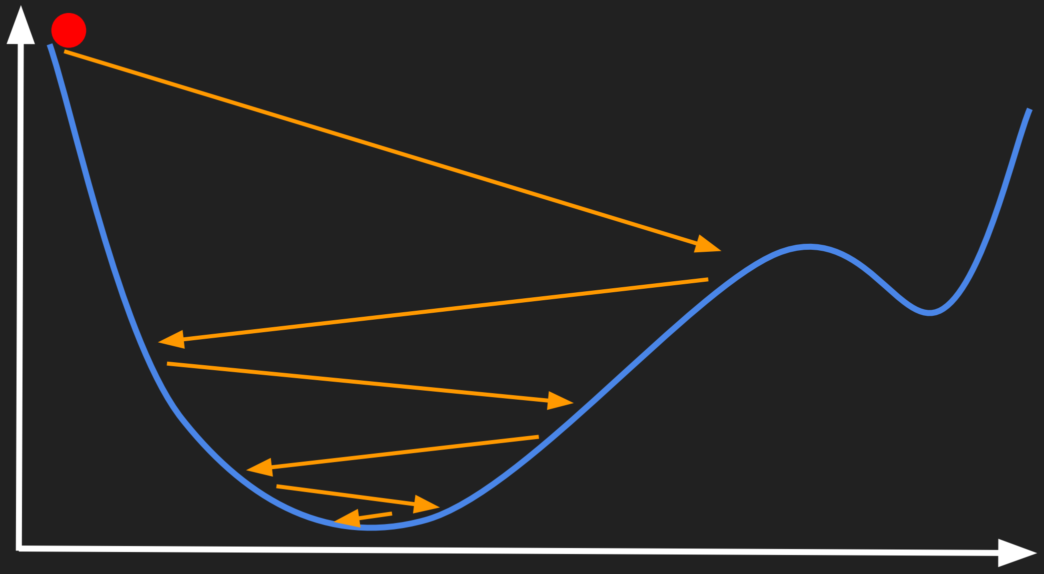


Example with 2 variables

Gradient Descent: With Momentum

What happens when we try to reach the optimum point?

To improve the convergence we give gradient descent some <u>memory</u>



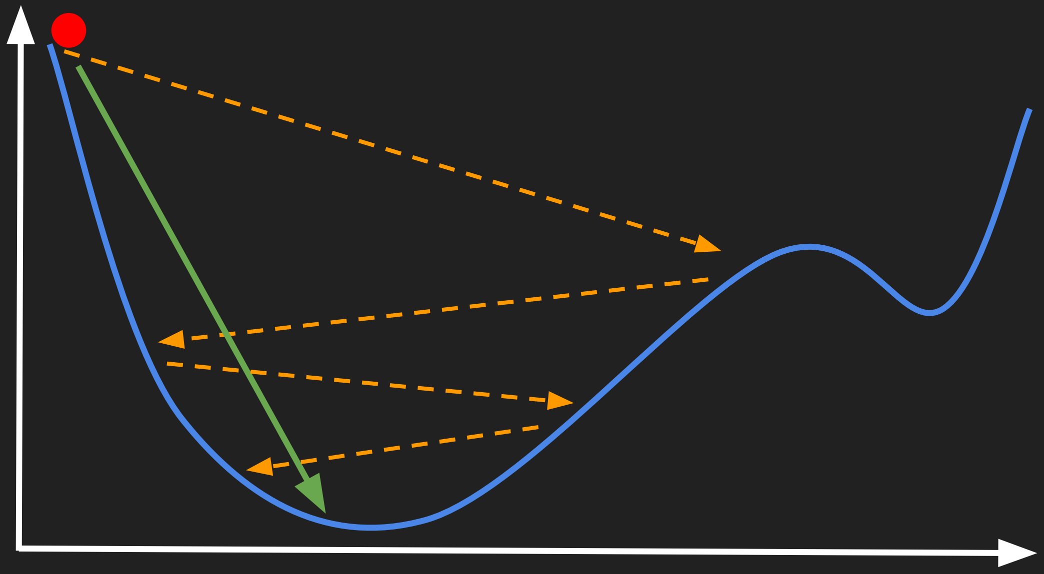
We only use one variable for simplicity

Gradient Descent: With Momentum

Memory smoothes gradient implementing a moving average.

We achieve this by creating a linear combination of the current and previous gradients

Resembles the movement of a heavy ball reluctant to change direction



We only use one variable for simplicity

Gradient Descent: With Momentum

FORMULA: $$x_{i+1} = x_i - \gamma_i \nabla f(x_i) + \underline{\alpha \Delta x_i}$$

Really similar to regular gradient descent

Gradient Descent: Types

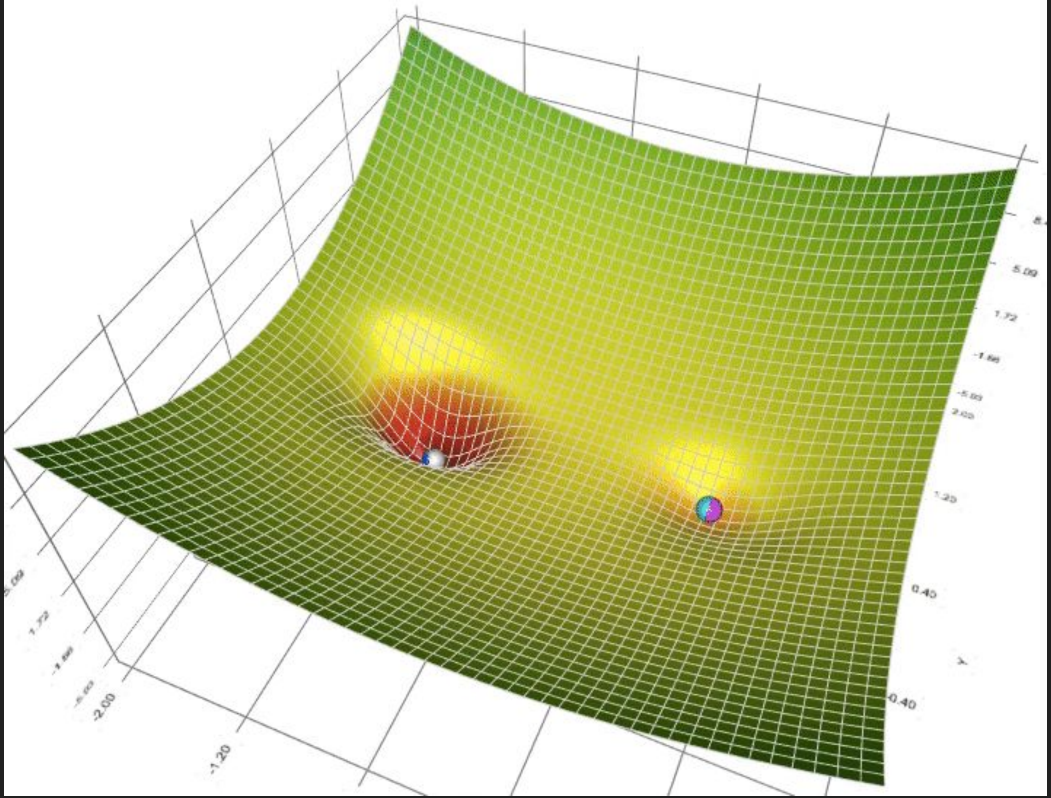
Momentum

Gradient Descent

RMSProp

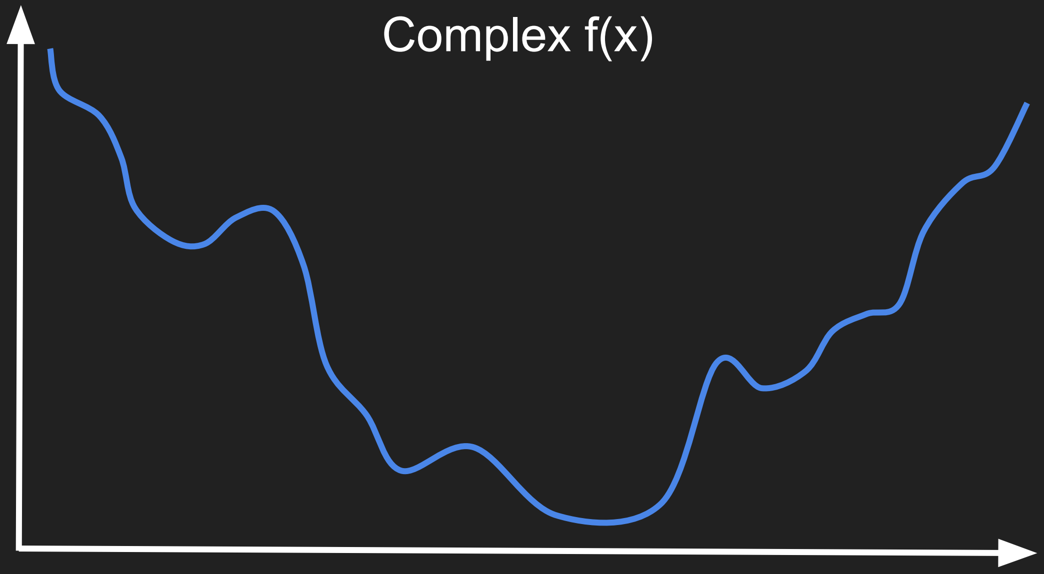
AdaGrad

Adam



Stochastic Gradient Descent

Computing the gradient can be very <u>time consuming</u>

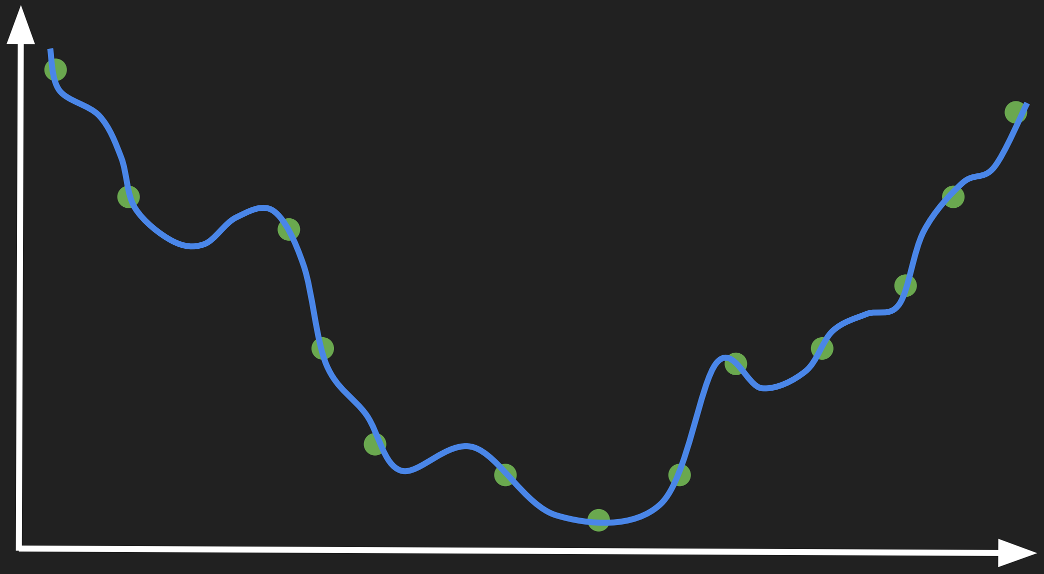


We only use one variable for simplicity

Stochastic Gradient Descent

How can we find a “cheap” approximation of the gradient?

We can reduce the amount of computation by taking a sum over a smaller set.

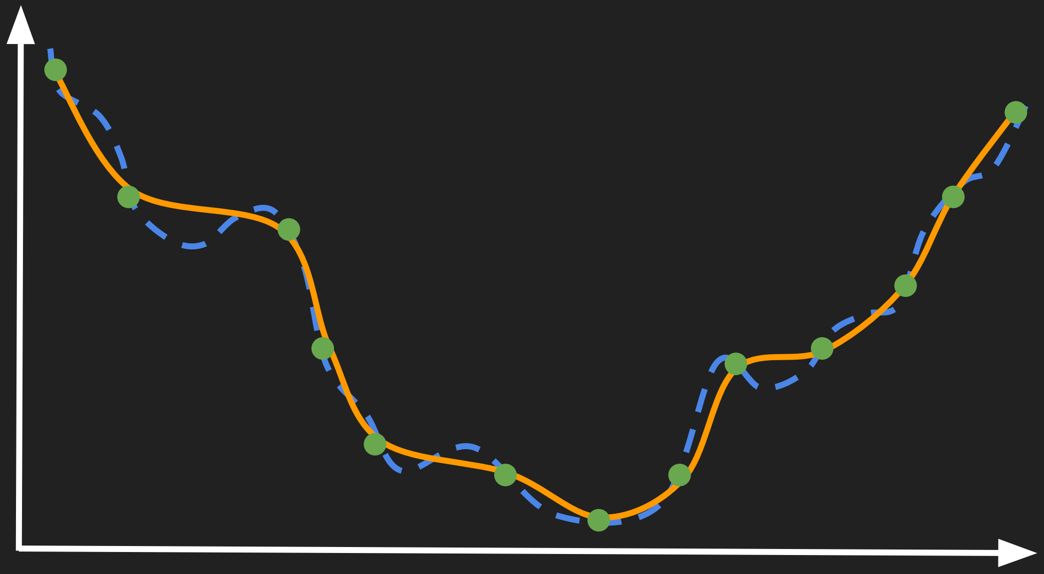


We only use one variable for simplicity

Stochastic Gradient Descent

With this approach we do not know the gradient precisely, but instead only know a noisy approximation to it.

The estimate also allows us to get out of local minimums

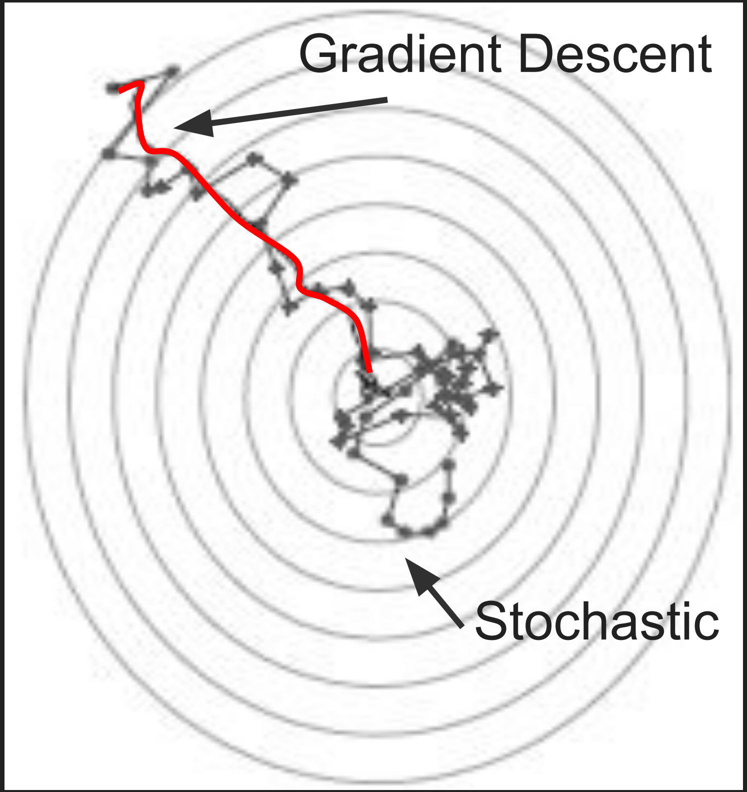


We only use one variable for simplicity

Stochastic Gradient Descent

The goal in machine learning does not necessarily need a precise estimate of the minimum of the objective function.

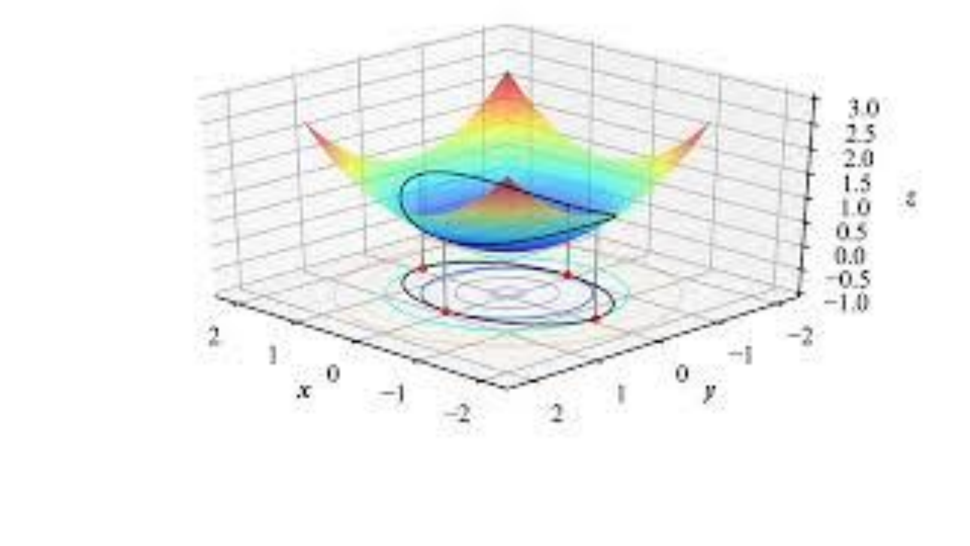
Stochastic gradient descent is <u>very effective</u> in large-scale machine learning problems

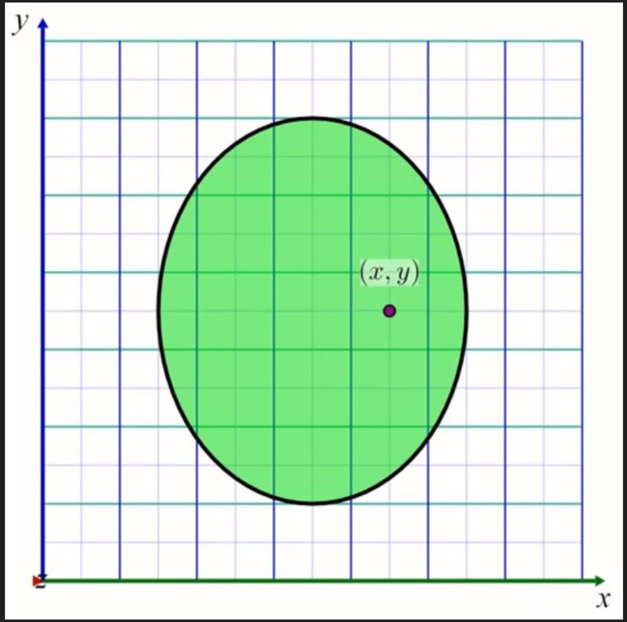


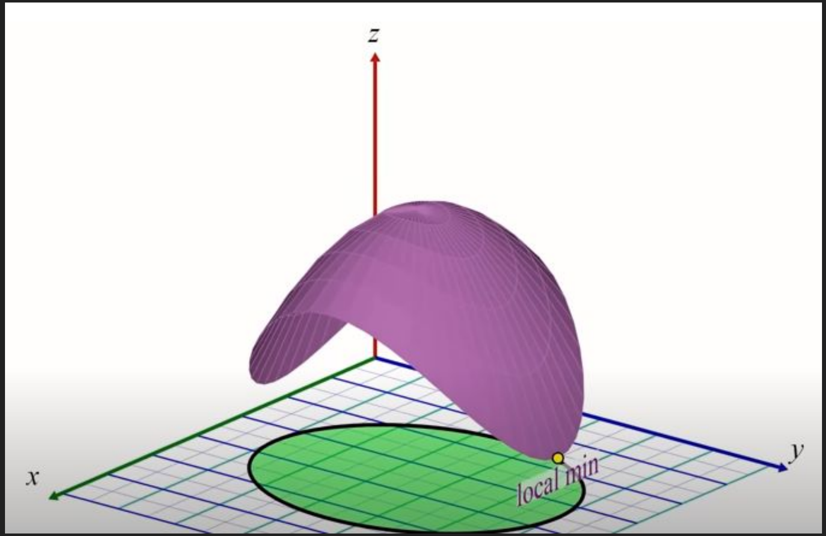
Constrained Optimization

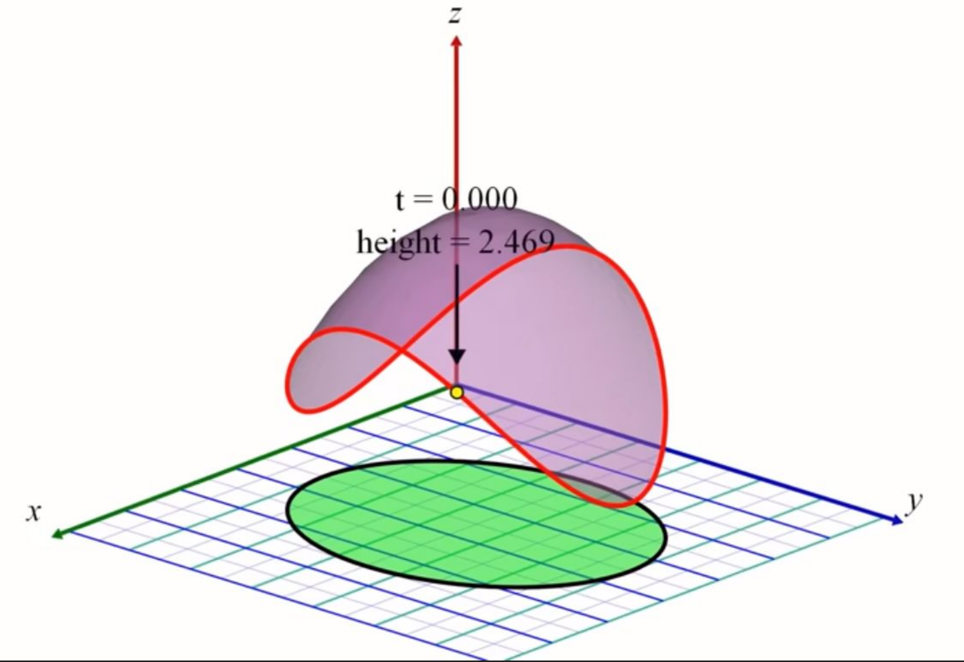
What is a constraint?

imitation or Restriction that we impose on a function

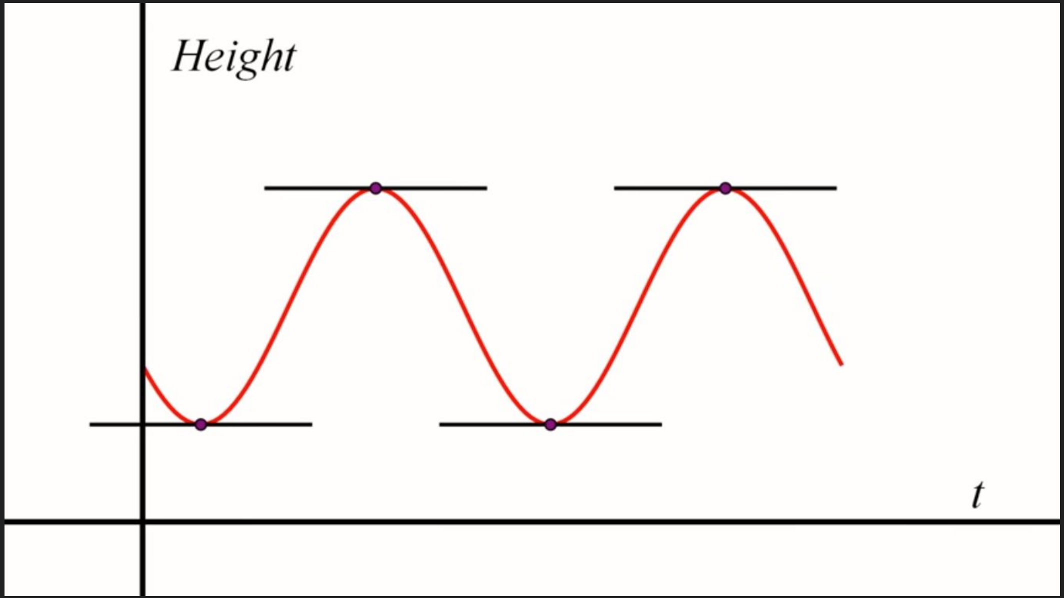



y
(x, y)
x


z
x
y
local min



Parameterize with a variable t



Single variable function

Lagrange Multipliers

We want to find the minimum value

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & g_i(\boldsymbol{x}) \leqslant 0 \quad \text{for all} \quad i = 1, \dots, m \end{aligned}$$

Lagrangian

We have the lagrangian depending on two variables

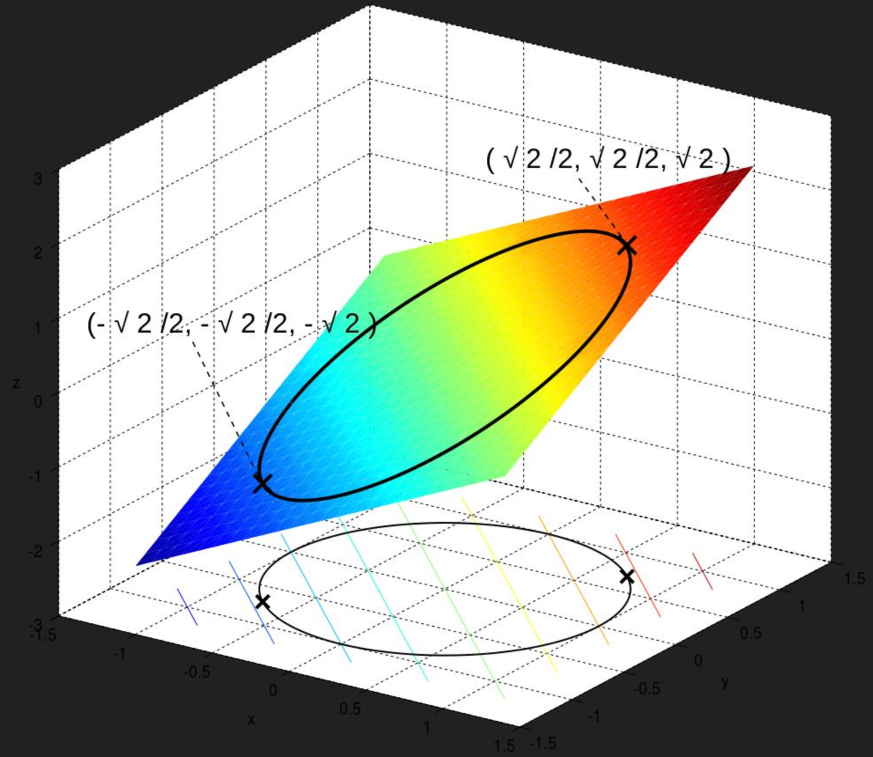
$$\mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x})$$

$$\text{Find:} \quad \max_{x,y} f(x) = \max_{x,y}[x + y]$$

$$\text{subject to} \quad g(x, y) = x^2 + y^2 = 1$$

$$g(x, y) = x^2 + y^2 - 1 = 0$$

$$\Rightarrow \mathcal{L}(x, y) = f(x) + \lambda g(x)$$
$$= x + y + \lambda(x^2 + y^2 - 1)$$



Duality in Optimization

What do we call "duality"?

Changing the set of variables that we optimize, we can redefine the problem:

We go from … to …

Minimization → Maximization

Maximization → Minimization

We call these two problems the **Dual** and the **Primal**

Primal→**Dual**

We get two outcomes from this framing

- **Weak Duality**

The solutions to the primal problem will be greater or equal

$$\min_x \mathcal{D}(x) \leq \max_x \mathcal{P}(x)$$

- **Strong Duality**

The solution to the Dual and the Primal problems are the same

$$\min_x \mathcal{D}(x) = \max_x \mathcal{P}(x)$$

When creating the lagrangian we get a Primal problem

Reminder: **Primal**→**Dual**

$$\mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x})$$

$$\begin{aligned} &\min_{\boldsymbol{x}} \quad f(\boldsymbol{x}) \\ &\text{subject to} \quad g_i(\boldsymbol{x}) \leqslant 0 \end{aligned}$$

we can transformed it into a Dual Prob just by a change of variables:

$$\mathcal{D}(\boldsymbol{\lambda}) = \min_{x} \mathcal{L}(\mathbf{x}, \lambda)$$

$$\Rightarrow \max_{\boldsymbol{\lambda} \geq 0} \mathcal{D}(\boldsymbol{\lambda}) = \max_{\boldsymbol{\lambda}} \min_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{x})$$

Has weak duality!

Because of the minimax inequality:

$$\max_{\boldsymbol{y}} \min_{\boldsymbol{x}} \varphi(\boldsymbol{x}, \boldsymbol{y}) \leqslant \min_{\boldsymbol{x}} \max_{\boldsymbol{y}} \varphi(\boldsymbol{x}, \boldsymbol{y})$$

Lagrangian:

$$\mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x})$$

$$\max_{\boldsymbol{\lambda} \geq 0} D(\lambda) = \max_{\boldsymbol{\lambda}} \min_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{x})$$

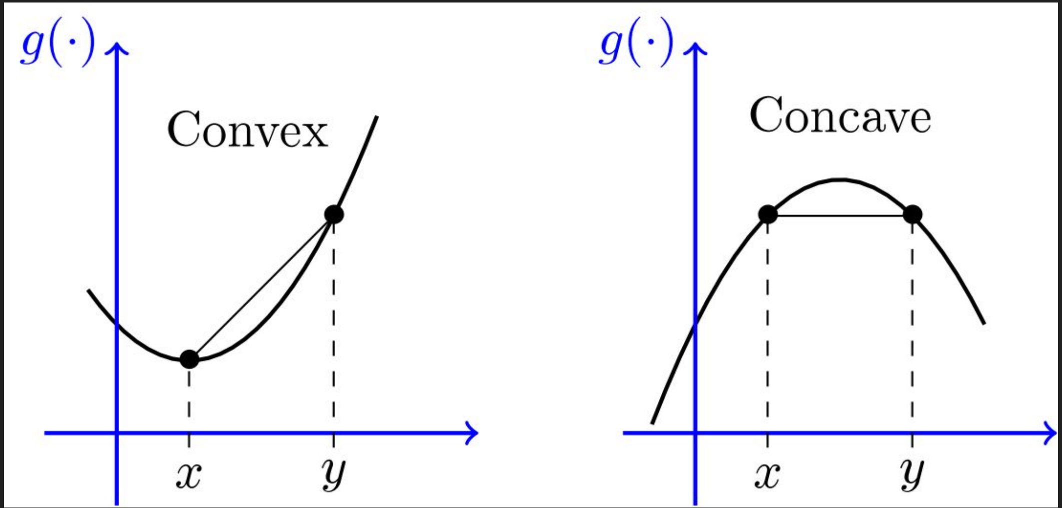
$$\max_{\boldsymbol{\lambda}} \min_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{x}) \leq \min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\lambda}, \boldsymbol{x})$$

Dual Problem

“““Primal Problem”””

What happens if the function is convex?

Examples:



When a optimization constrained problems has all convex functions we say is a **convex optimization problem**

Convex Optimization Problem

$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$

$$\text{subject to } g_i(\boldsymbol{x}) \leqslant 0 \quad \text{for all} \quad i = 1, \ldots, m$$

… and the functions are complex.

Reminder Strong Duality:

$$\min_x \mathcal{D}(x) = \max_x \mathcal{P}(x)$$

We get strong duality!

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & g_i(\boldsymbol{x}) \leqslant 0 \end{aligned}$$

$$\begin{aligned} \max_{\boldsymbol{\lambda}} \mathcal{D}(\boldsymbol{\lambda}) = \min_{\boldsymbol{x}} & \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) \\ \text{subject to} \quad & \boldsymbol{\lambda} \geq 0 \end{aligned}$$

Therefore is easier this way!

We can choose which function to optimize (the dual or primal)

A small recap

- We can use gradient descent very effectively
- Specially useful when training Neural Networks

- We can use Lagrange Multipliers to optimize with constraints
- Convex functions are easier to optimize

Thanks for your attention!