Continuous Optimization: Algorithms and Complexity
ORIE 6365

Lecture 1: Introduction. Complexity of Optimization Problems.

Nikita Doikov

Cornell University
School of Operations Research and Information Engineering (ORIE)

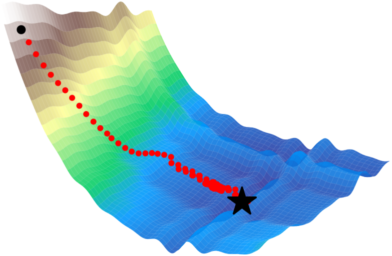
January 21, 2026

Theory Course in Continuous Optimization

The problem: $\min_{x \in Q} f(x)$

- $f : Q \to \mathbb{R}$ is the objective function
- x is the decision variable which belongs to a feasible set: $x \in Q \subseteq \mathbb{R}^n$

Our goal: to find a best possible $x^\star$



Modern and classic applications:

- dimension n is huge ($n \approx 10^{12}$ for a frontier LLM)
- f or Q are "difficult"

The scope of this course:

- design and analysis of optimization algorithms
- complexity theory — how to estimate their efficiency? which algorithm is better and when?

Sources of Optimization Problems

Machine Learning, AI:

$f(x) =$ loss of the model on training data

x are the weights of the model that we train

- Linear regression, Logistic regression, Neural networks ($x \in \mathbb{R}^n$)

Graphs:

$f(x) =$ the value of the flow x in a network ($x \in Q$, where Q is the set of flows)

Finance:

$f(x) =$ the expected return for a portfolio selection x

...

- Notice that $\max_x f(x) = -\min_x[-f(x)]$

Course Information

Instructor: Nikita Doikov — please call me *Nikita*

Course website: doikov.com/teaching/orie6365-s26/ and Canvas

Disclaimer: this is a graduate-level theory course

Grading:

- Theory homeworks (around 3-4) ≈ *50%*
- Practical assignments (around 2) ≈ *30%*
- Final take-home exam ≈ *20%*

The first homework is already out! **Due on January 31**

Today's Outline

Terminology

Core components of algorithmic design

Analysis: grid search algorithm

General Forms of Optimization Problems

$$\min_{x \in Q} f(x) \qquad \text{where} \qquad Q \subseteq \mathbb{R}^n$$

- **Unconstrained optimization**: $Q = \mathbb{R}^n$
  - Smooth / Non-smooth problems: whether $f : \mathbb{R}^n \to \mathbb{R}$ is "smooth" (at least differentiable) / non-differentiable
- **Constrained optimization**: $Q \subset \mathbb{R}^n$ is a constraint set
  1. Simple constraints. For example, a ball in some norm $\| \cdot \|$, for $R > 0$:

     $$Q \quad = \quad \{x \in \mathbb{R}^n \quad : \quad \|x\| \leq R\}.$$

     "Simplicity" means that we can perform basic operations with Q efficiently, e.g.

     $$\text{proj}_Q(x) \quad = \quad \arg\min_{y \in Q} \|y - x\|$$

  2. Functional constraints. $Q = \big\{x \in \Omega : g_1(x) \leq 0, \ldots, g_m(x) \leq 0\big\}$, for a simple $\Omega \subseteq \mathbb{R}^n$ where $g_i : \Omega \to \mathbb{R}$ are some functions.
     NB: It is enough to use only "$\leq$" instead of "$\geq$" and "$=$".
- **Convex optimization:** Q is a convex set, and $f : Q \to \mathbb{R}$ is a convex function

Types of Solutions

Optimization problem: $\boxed{\min_{x \in Q} f(x)}$

A point $x \in Q$ is called **feasible point**, and $x \notin Q$ is **infeasible**

A point $x^* \in Q$ is called **global solution** if

$$f(x^*) \quad \leq \quad f(x), \qquad \forall x \in Q.$$

- An ideal goal, but can be difficult to find
- We will denote $f^* = f(x^*)$ — sometimes it is easier to compute

A point $\bar{x} \in Q$ is called **local solution** if there exists a neighborhood of $\bar{x}$ (an open set $U \in \mathbb{R}^n$ containing $\bar{x}$) such that

$$f(\bar{x}) \quad \leq \quad f(x), \qquad \forall x \in Q \cap U.$$

- Usually much easier to find than a global solution

Why Continuous?

Continuous optimization means that the objective function f is continuous (or even differentiable) and the variables $x \in Q \subseteq \mathbb{R}^n$ can take continuous values.

A question: *Will you keep attending this course?*

1. Yes, I will. $x \in \{0, 1\}$ **Very hard** to predict reliably in practice (even for advanced AI systems).

2. Probably. $x \in [0, 1]$ **Easier to predict**, e.g.: $x \geq 0.8$.

Continuous decisions are often easier to make than **discrete** ones.

- It is very fruitful to work with the continuous space of parameters

An Example of Constrained Problem

Let $x \in \mathbb{R}$ and consider 2 functions:

$$g_1(x) = x^2 - 1, \qquad g_2(x) = 1 - x^2$$

- Both are continuous nice functions
- However, the constraint set

$$\begin{aligned} Q &= \{x \in \mathbb{R} : g_1(x) \le 0, \; g_2(x) \le 0\} \\ &= \{x \in \mathbb{R} : x^2 \le 1, \; x^2 \ge 1\} \\ &= \{x \in \mathbb{R} : x^2 = 1\} = \{\pm 1\} \end{aligned}$$

  is discrete!

- Thus, $\min_{x \in Q} f(x)$ will be discrete problem
- Problems with functional constraints can be hard even for "simple" functions

Example: Linear Programming

Linear Programming: both objective and constrains are linear / affine functions

- $f(x) = \langle c, x \rangle$ for some $c \in \mathbb{R}^n$
- For all $1 \leq i \leq m$: $g_i(x) = \langle a_i, x \rangle - b_i$ where $a_i \in \mathbb{R}^n$, $b_i \in \mathbb{R}$
- We obtain the problem of minimizing a linear function over a polyhedron:

$$\min_{x \in \mathbb{R}^n} \Big\{ \langle c, x \rangle \ : \ \langle a_1, x \rangle \leq b_1, \ \dots, \ \langle a_m, x \rangle \leq b_m \Big\}$$

- Matrix notation: form $A \in \mathbb{R}^{n \times m}$ with $a_1, \dots, a_m \in \mathbb{R}^n$ to be its columns:

$$A = \begin{bmatrix} a_1 \ a_2 \ \cdots \ a_m \end{bmatrix} \in \mathbb{R}^{n \times m} \quad \text{and} \quad b = [b_1 \ b_2 \ \cdots \ b_m]^\top \in \mathbb{R}^m.$$

  Then $Q = \{ x \in \mathbb{R}^n \ : \ A^\top x \leq b \}$.
- An instance of the problem is given by **input data**: $P = \{A, b, c\}$.
- Linear programming is already non-trivial to solve.

This course: two polynomial-time algorithms for linear programming (*ellipsoid* method and *interior-point* method)

Feasibility Problem

Let $Q \subseteq \mathbb{R}^n$ be given.

Feasibility problem: to find $x^* \in Q$. (an optimization problem with $f \equiv 0$)

Example

Let $Q = \{x \in \mathbb{R}^n : Ax = b\} = \{x \in \mathbb{R}^n : Ax - b \leq 0, b - Ax \leq 0\}$
Then the feasibility problem is to solve the linear system: $Ax^* = b$. □

Consider an **optimization problem**: $\min_{x \in Q} f(x)$ with a function $f : Q \to \mathbb{R}$

- Introduce an extra variable $t \in \mathbb{R}$ and extra constraint $f(x) \leq t$
- Then, $\min_{x \in Q} f(x)$ is equivalent to the problem with linear objective:

$$\min_{(x,t) \in Q'} t$$

where $Q' = \{(x, t) \in \mathbb{R}^{n+1} : x \in Q, f(x) \leq t\}$

- Running binary search over t: **an optimization problem** $\Rightarrow$ **feasibility problem**
- Feasibility problems are as hard as optimization ones

The Most Difficult Problem in the World

Let $x^* \in \mathbb{R}^n$ be an arbitrary point.
Set

$$f(x) \quad = \quad \begin{cases} 0, & x = x^* \\ 1, & \text{everywhere else} \end{cases}$$

$\min_x f(x)$ means to find x^* (which is arbitrary!)

- Thus, we can encode any problem in the world as optimization problem
- **NB:** we can even make f continuous

Optimization problems are generally unsolvable

- We will investigate problem classes that we can solve by efficient algorithms
- For each problem class, we will associate its corresponding complexity

Problem Class

Imagine we have one fixed function: $f_0 : \mathbb{R}^n \to \mathbb{R}$ that has global minimum x^*.

The best algorithm for solving this problem? `Return` x^*

- Perfect on functions with minimum x^*
- Wrong on all other functions (silly method!)

Instead of fixing one function, we always fix a **problem class** $\mathcal{P}$, a family of problems

Examples:

- $\mathcal{P} = \{f_0\}$ consisting of one function — too small
- $\mathcal{P} = \{f \text{ s.t. } f \in \mathcal{C}(\mathbb{R}^n)\}$ consisting of all continuous functions — too large
- $\mathcal{P} = \{f \text{ s.t. } f \text{ is convex and } \nabla f \text{ is Lipschitz}\}$ — already much better
- $\mathcal{P} = \{(f, Q) \text{ s.t. } f \text{ is linear and } Q \text{ is a polyhedron}\}$ LP — polynomially solvable
- ...

Performance of an algorithm is measured over all problems from $\mathcal{P}$

Oracles

An **oracle** is how the algorithm gets access to the function

Most often, we will consider the black-box local oracles:

at any point $x \in Q$, an algorithm can "learn" a local information about objective

$$x \mapsto \mathcal{O}(x)$$

- **Zeroth-order oracle:** $\mathcal{O}(x) = \{f(x)\}$, only function values
- **First-order oracle:** $\mathcal{O}(x) = \{f(x), \nabla f(x)\}$, function and gradient
- **Second-order oracle:** $\mathcal{O}(x) = \{f(x), \nabla f(x), \nabla^2 f(x)\}$, function, gradient, and Hessian
- ...

What is Optimization Algorithm?

A general scheme of an optimization method:

Initialization: $x_0 \in Q$

For $k \geq 0$ **iterate:**

$$I_k = \{ \mathcal{O}(x_0), \ldots, \mathcal{O}(x_k) \}$$ // collect information

If $S_k(I_k)$ **then** // stopping condition

return $R_k(I_k)$ // return the result

$$x_{k+1} = A_k(I_k)$$ // compute next point

To define the method we need to specify three sequences of mappings

- Main iterates: $(A_0, A_1, \ldots)$
- Stopping conditions: $(S_0, S_1, \ldots)$
- How to form the result: $(R_0, R_1, \ldots)$

Stopping Condition

In practice, we always run methods for a finite number of iterations

- We cannot hope to get exact solution x^*
- Therefore, we always work with approximate solutions $x_k \approx x^*$, where x_k is the result of the method

Measures of Inexactness (unconstrained minimization $f^* = \min_{x \in \mathbb{R}^n} f(x)$):

- Functional residual: $f(x_k) - f^* \leq \varepsilon$
- Pointwise distance: $\|x_k - x^*\| \leq \varepsilon$
- Gradient norm: $\|\nabla f(x_k)\| \leq \varepsilon$

where $\varepsilon > 0$ is a desired accuracy, which is part of the problem formulation

Key Elements of Algorithmic Design

- Problem class $\mathcal{P}$
- Measure of inexactness and desired accuracy $\varepsilon > 0$
- Oracle $\mathcal{O}$ (type of information we have access to) $\leftrightarrow$ class of algorithms $\mathcal{A}$

Complexity of a method on a problem is the minimal number of iterations required to solve the problem with the fixed accuracy $\varepsilon > 0$

- Can be $+\infty$
- Also called *oracle complexity*, *analytical complexity*, or *iteration complexity*

Complexity of a method on a problem class is the maximum over complexity on a problem p, over all $p \in \mathcal{P}$

- Among all problems, we pick the worst one for the method

Example: Global Optimization

Problem class: $\min_{x \in B} f(x)$

where

- $B = \{x \in \mathbb{R}^n \;:\; \|x\|_\infty \le R\}$ is a box
- f is continuous and Lipschitz:

$$|f(y) - f(x)| \quad \le \quad L\|y - x\|_\infty, \quad \text{for all} \quad x, y \in B$$

Parameters of the problem class:

- Dimension $n \ge 1$
- Radius of the box $R > 0$
- Lipschitz constant $L > 0$

Accuracy condition:

$$f(\bar{x}) - f^* \quad \le \quad \varepsilon$$

Type of oracle:

- Zeroth-order black-box oracle: $x \mapsto f(x)$

Grid Search Algorithm

1. Choose $p \geq 1$ (an integer parameter of the method)
2. Generate p^n points,

$$x_{(t_1,\ldots,t_n)} \quad = \quad \left[-\frac{p-1}{p} \cdot R + \frac{2R}{p} t_1 \ , \ \ldots \ , \ -\frac{p-1}{p} \cdot R + \frac{2R}{p} t_n\right]$$

   where $0 \leq t_i \leq p-1$ for each coordinate $1 \leq i \leq n$
3. Among all these p^n points, find the point $\bar{x}$ with the smallest function value. **Return** $\bar{x}$.

- This is a zeroth-order method.

Theorem. Let $p \geq 1$ and $\bar{x}$ be the result of the grid search algorithm. Then,

$$f(\bar{x}) - f^* \quad \leq \quad \frac{2LR}{p}$$