AI505

Optimization

Course Introduction

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

Who is here?

44 in total registered in ItsLearning

- **AI505 (7.5 ECTS)**
  31 from Bachelor in AI

- **AI801 (10 ECTS)** or **AI505 (7.5 ECTS)** + **IAAI501 (2.5 ECTS)**
  13 from Master Educations (which ones?)

Prerequisites

- Calculus
- Linear Algebra
- Programming

Outline

Aims of the course

Learn about **optimization**:

- continuous multivariate optimization
- discrete optimization

Optimization is an important tool in **machine learning**, **decision making** and in analysing physical systems.
In mathematical terms, an **optimization problem** is the problem of finding the **best solution** from the set of all **feasible solutions**.

The first step in the optimization process is constructing an appropriate **mathematical formulation**. The second is devising an **algorithm** for solving the mathematical formulation.

Contents of the Course (Pensum)

Unit	Main topic
1	Introduction, Univariate Problems
2	Multivariate Problems, Gradient-Based Methods
3	Derivative-Free Methods
4	Optimization for Machine Learning
5	Constrained Optimization, Linear Programming
6	Sampling Methods
7	Discrete Optimization and Heuristics

Practical Information

Teacher:	Marco Chiarandini (imada.sdu.dk/u/marco/)
Instructor:	Sai Ganesh Nagarajan (H21)

Schedule, alternative views:

- mitsdu.sdu.dk, SDU Mobile
- Official course description (skemaplanen)
- ItsLearning
- https://ai-505.github.io

Schedule (16 weeks):

- Introductory classes: 16 classes (32 hours)
- Training classes: 16 classes (32 hours)
- Scheduled: $(16 \times 2) + 2$ classes $\times 2 = 68$ hours

Communication Means

- ItsLearning $\Leftrightarrow$ External Web Page
  (link `https://ai-505.github.io`)
- Announcements + Discussion Board + Submissions in ItsLearning
- Write to Marco (`marco@imada.sdu.dk`) or to instructor
- Collaborate with peers

$\leadsto$ **It is good to ask questions!!**

$\leadsto$ **Let me know if you think we should do things differently!**

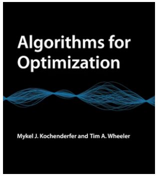
Sources — Reading Material

Main reference:

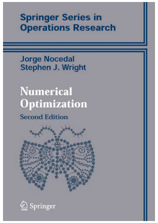
[KW] Mykel J. Kochenderfer, Tim A. Wheeler. Algorithms for Optimization 2019. The MIT Press.

Others

[NW] J. Nocedal and S. J. Wright, Numerical Optimization, Second Edition. Springer Series in Operations Research, 2006


Algorithms for Optimization

Mykel J. Kochenderfer and Tim A. Wheeler


Springer Series in Operations Research

Jorge Nocedal
Stephen J. Wright

Numerical Optimization
Second Edition

Springer

Course Material

External Web Page is the main reference for list of contents (ie[1], syllabus, pensum).

It will collect:

- slides
- list of topics and references
- exercises
- links
- tutorials for programming tasks

[1] ie = id est = that is; eg = exempli gratia = for example; wrt = with respect to; et al. = et alii = and others

Assessment

Portfolio consisting of:

- mandatory assignments in groups of 2:
  1. assignment
  2. assignment

  Time line to be announced.
- oral exam on June 29-30, 2026

The oral examination consists of questions on the basis of the handed in assignments and can be extended to other parts of the course curriculum.

Both elements must individually meet the learning objectives.

Final grade: primarily based on element 1, but element 2 may adjust the grade up or down by one grade level.

(language: Danish and/or English)

Exercise Sessions (1/2)

For exercises in general:

- Both theoretical, modelling and programming tasks
- No mandatory hand-ins, voluntary participation, but all lectures and exercises are relevant for the exam
- Help each other! Teaching others is the best form of learning
- Sai will be there to help. First hour: you work and Sai gives personalize help. Second hour: Sai presents a subset of solutions in plenary.
- Exercises should help you to learn - find the setting that works best for you, don't hesitate to extend with other material that works better for you. You can use Gen AI tools.

Exercises are group work:

- Exercises best done in pairs of 2-3 people
- Try to gather in different groups every now and then

Exercise Sessions (1/2)

Notation: $^+$, *, ' '

- plus exercises are to be done before the class
- starred exercises are done in class. They are good examples of assignment questions
- unmarked exercises are for self study

Coding

Set up your local Python programming environment and use the available tutorial to:

- Brush up some knowledge on Python-IDEs and Git
- Grasp some basics on jupyter notebooks
- Recap basics of data processing and visualization

We will span several programming modes: functional, imperative, object oriented, small scripts, large code bases, use of modules and libraries. We will use git.

Outline

Introduction

- Applications of Optimization
  - Physics
  - Business
  - Biology
  - Engineering
  - Machine Learning
  - Logistics and Scheduling
- Objectives to Optimize
  - Efficiency
  - Safety
  - Accuracy
- Constraints
  - Cost
  - Weight
  - Structural Integrity
- Challenges
  - High-Dimensional Search Spaces
  - Multiple Competing Objectives
  - Model Uncertainty

Real World, Model, Representation, Implementation

The Real World: That messy thing we are trying to study (with computers).

Model: Mathematical object in some class M.

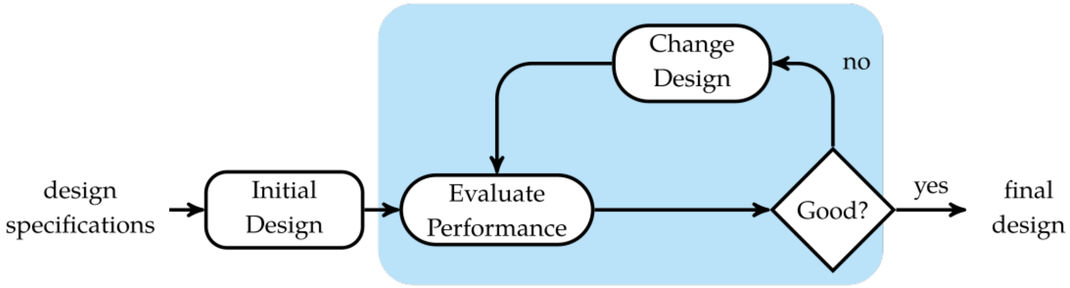
Representation: An object of an abstract data type R used to store the model.

Implementation: An object of a concrete type used to store the model.

Any object from the real world might have different models.
Any model might have several representations (exact).
And representation might have different implementations (exact).

We will focus on the algorithmic aspects of optimization that arise after the problem has been properly formulated

Optimization Process



An optimization algorithm is used to incrementally improve the design until it can no longer be improved or until the budgeted time or cost has been reached.

Optimization Process

Search the space of possible designs with the aim of finding the best one.

Depending on the application, this search may involve:

- evaluating an analytical expression (white or glass box)
- running physical experiments, such as wind tunnel tests (black box)
- running computer simulations

Modern optimization techniques can be applied to problems with millions of variables and constraints.

Definitions – Functions

- Univariate single-valued function: $f : \mathbb{R} \to \mathbb{R}$ is a function that takes as input a single real number, say x, and produces a single real number as output.
- Multivariate single-valued function: $f : \mathbb{R}^n \to \mathbb{R}$ is a function that takes as input a vector $\boldsymbol{x}$ of n real numbers and produces a single real number as output.
- (Multivariate) Vector-valued function: $f : \mathbb{R}^n \to \mathbb{R}^m$ is a function that takes as input a vector $\boldsymbol{x}$ of n real numbers and produces a vector $\boldsymbol{y}$ of m real numbers as output.

We will work primarily with multivariate single-valued functions.

Basic Optimization Problem

$$\boxed{\begin{aligned} &\underset{\boldsymbol{x}}{\text{minimize}}\ f(\boldsymbol{x}) \\ &\text{subject to } \boldsymbol{x} \in \mathcal{X} \end{aligned}}$$

- Feasible Set $\mathcal{X} \subseteq \mathbb{R}^n$
- Design Point $\boldsymbol{x}$
- Design Variables
- Objective Function: $f : \mathbb{R}^n \to \mathbb{R}$ (scalar-valued function)
- Minimizer

Any value of $\boldsymbol{x}$ from among all points in the feasible set $\mathcal{X}$ that minimizes the objective function is called a **solution** or **minimizer**. A particular solution is written $\boldsymbol{x}^*$.

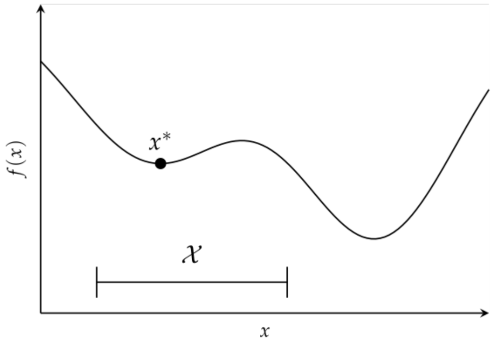
$$\boxed{\boldsymbol{x}^* = \operatorname{argmin} f(\boldsymbol{x}) \text{ subject to } \boldsymbol{x} \in \mathcal{X}}$$

There is only one minimum but there can be many minimizers

maximize $f(\boldsymbol{x})$ subject to $\boldsymbol{x} \in \mathcal{X} \equiv$ minimize $-f(\boldsymbol{x})$ subject to $\boldsymbol{x} \in \mathcal{X}$

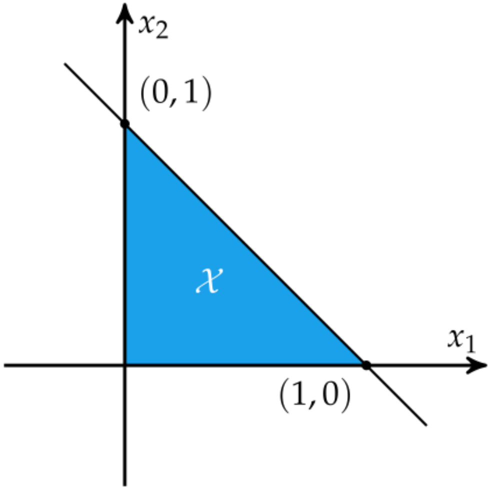
Basic Optimization Problem

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & f(\boldsymbol{x}) \\ \text{subject to} \quad & \boldsymbol{x} \in \mathcal{X} \end{aligned}$$



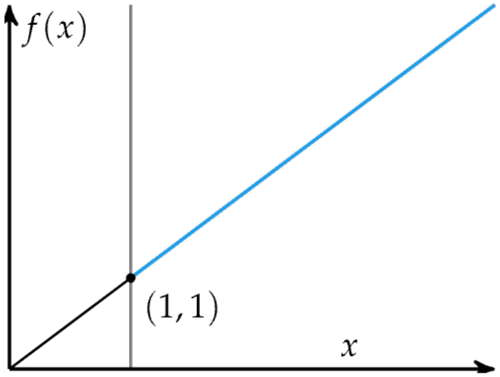
Constraints

$$\begin{aligned} \underset{x_1, x_2}{\text{minimize}} \quad & f(x_1, x_2) \\ \text{subject to} \quad & x_1 \geq 0 \\ & x_2 \geq 0 \\ & x_1 + x_2 \leq 1 \end{aligned}$$



Constraints

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & x > 1 \end{aligned}$$



The problem has no solution.
$x = 1$ would not be feasible.

$x = 1$ would be the solutions to

$$\underset{x}{\text{infimum}}\ f(x) \text{ subject to } x > 1$$

infimum of a subset $\mathcal{X}$ of a partially ordered set $\mathcal{P}$ is the greatest element in P that is less than or equal to each element of $\mathcal{X}$, if such an element exists. Aka, **greatest lower bound**.

Decision Space and Objective Space

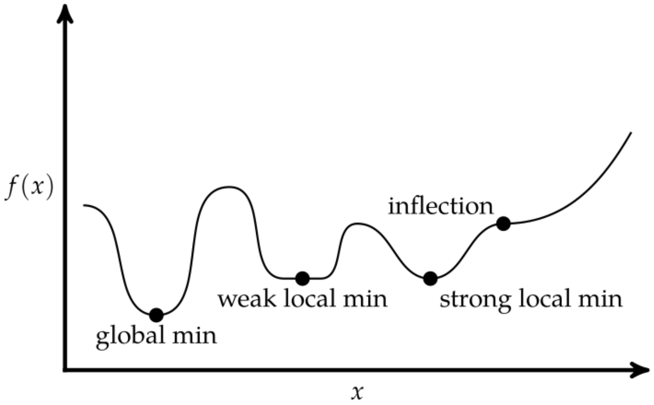
Plot to add.

Critical Points

Univariate function

global vs local minimum

Def. A point $\boldsymbol{x}^*$ is at a **local minimum** (or is a local minimizer) if there exists a $\delta > 0$ such that $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x}$ with $\|\boldsymbol{x} - \boldsymbol{x}^*\| < \delta$.

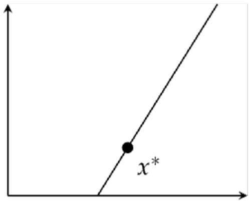


Conditions for Local Minima

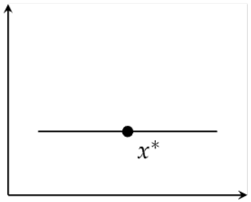
Univariate objective functions, assuming they are differentiable (Derivatives exist), without constraints

Local minimum: Necessary condition but not sufficient condition:

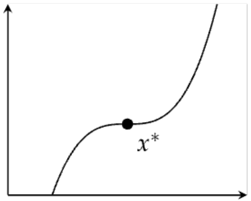
1. $f'(x^*) = 0$, the first-order necessary condition (FONC)
2. $f''(x^*) \geq 0$, the second-order necessary condition (SONC)



SONC but not FONC



FONC and SONC



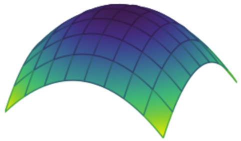
FONC and SONC

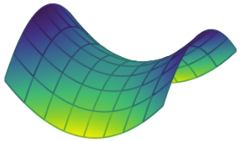
Conditions for Local Minima

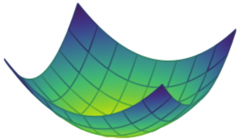
Multivariate objective functions, assuming they are differentiable (gradients and Hessians exist), without constraints

Local minimum: Necessary condition but not sufficient condition:

1. $\nabla f(\boldsymbol{x}^*) = 0$, the first-order necessary condition (FONC)
2. $\nabla^2 f(\boldsymbol{x}^*) \geq 0$, the second-order necessary condition (SONC)



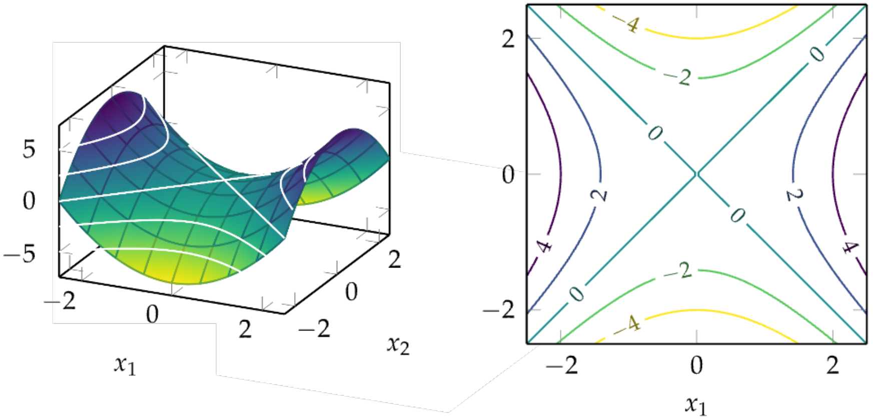




Contour Plots

$$f(x_1, x_2) = x_1^2 - x_2^2$$

can be rendered in a 3D space but convenient to represent it also in 2D showing the lines of constant output value



Taylor Expansion

From the *first fundamental theorem of calculus*,[2] we know that

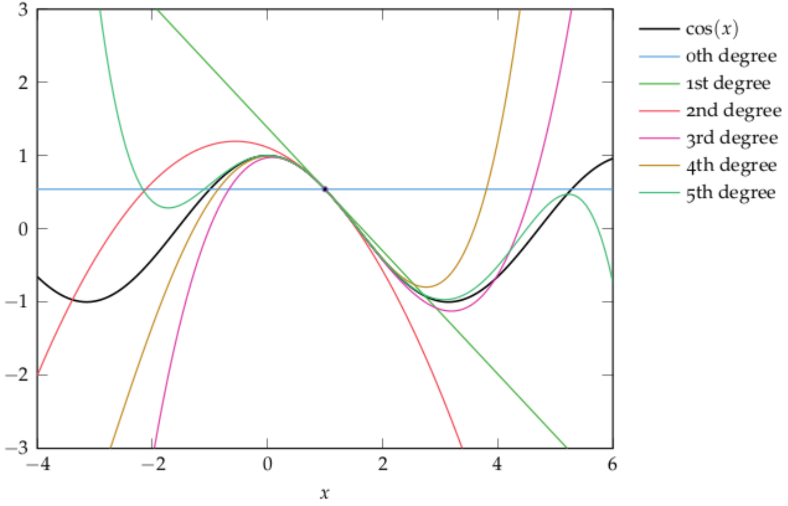
$$f(x+h) = f(x) + \int_0^h f'(x+a)\,da$$

Nesting this definition produces the Taylor expansion of f about x:

$$\begin{aligned}
f(x+h) &= f(x) + \int_0^h \left(f'(x) + \int_0^a f''(x+b)\,db \right) da \\
&= f(x) + f'(x)h + \int_0^h \int_0^a f''(x+b)\,db\,da \\
&= f(x) + f'(x)h + \int_0^h \int_0^a \left(f''(x) + \int_0^b f'''(x+c)\,dc \right) db\,da \\
&= f(x) + f'(x)h + \frac{f''(x)}{2!}h^2 + \int_0^h \int_0^a \int_0^b f'''(x+c)\,dc\,db\,da \\
&\vdots \\
&= f(x) + \frac{f'(x)}{1!}h + \frac{f''(x)}{2!}h^2 + \frac{f'''(x)}{3!}h^3 + \dots \\
&= \sum_{n=0}^{\infty} \frac{f^{(n)}(x)}{n!}h^n
\end{aligned}$$

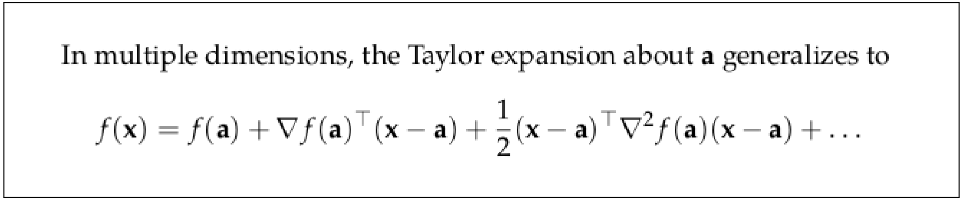
Taylor Expansion

A smooth function looks like a polynomial when you zoom in closely enough.



Taylor Expansion for Multivariate Scalar-Valued Functions

The Taylor expansion of a multivariate function $f : \mathbb{R}^n \to \mathbb{R}$ around a point $\boldsymbol{a}$ is given by:



In multiple dimensions, the Taylor expansion about $\mathbf{a}$ generalizes to

$$f(\mathbf{x}) = f(\mathbf{a}) + \nabla f(\mathbf{a})^\top (\mathbf{x} - \mathbf{a}) + \frac{1}{2}(\mathbf{x} - \mathbf{a})^\top \nabla^2 f(\mathbf{a})(\mathbf{x} - \mathbf{a}) + \ldots$$

- $\nabla f(\boldsymbol{a})$ is the gradient vector of f at $\boldsymbol{a}$, and
- $\nabla^2 f(\boldsymbol{a})$ is the Hessian matrix of f at $\boldsymbol{a}$.
- $O(||\boldsymbol{h}||^2)$ denotes the error by omitting higher-order terms. It becomes negligible as h approaches 0.

Example: Rosenbrock function

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

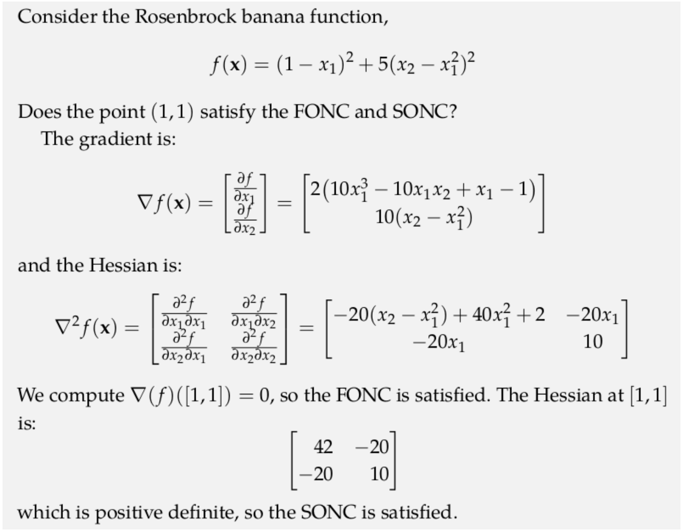
It has a global minimum at $(x, y) = (a, a^2)$, where $f(x, y) = 0$. Usually, these parameters are set such that $a = 1$ and $b = 100$. Only in the trivial case where $a = 0$ the function is symmetric and the minimum is at the origin.

Multivariate generalization
sum of $N/2$ uncoupled 2D Rosenbrock problems, and defined only for even N:

$$f(\mathrm{x}) = f(x_1, x_2, \ldots, x_N) = \sum_{i=1}^{N/2} \left[100(x_{2i-1}^2 - x_{2i})^2 + (x_{2i-1} - 1)^2\right].$$

This variant has predictably simple solutions.

Example: Rosenbrock function



Consider the Rosenbrock banana function,

$$f(\mathbf{x}) = (1 - x_1)^2 + 5(x_2 - x_1^2)^2$$

Does the point $(1, 1)$ satisfy the FONC and SONC?

The gradient is:

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 2\left(10x_1^3 - 10x_1x_2 + x_1 - 1\right) \\ 10(x_2 - x_1^2) \end{bmatrix}$$

and the Hessian is:

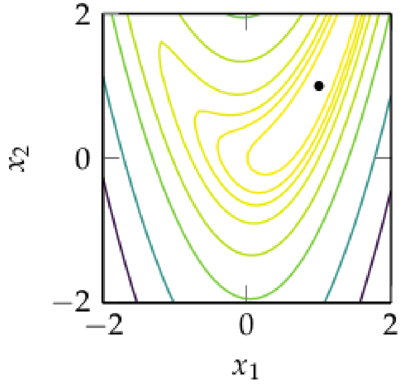
$$\nabla^2 f(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \frac{\partial^2 f}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2 \partial x_2} \end{bmatrix} = \begin{bmatrix} -20(x_2 - x_1^2) + 40x_1^2 + 2 & -20x_1 \\ -20x_1 & 10 \end{bmatrix}$$

We compute $\nabla(f)([1, 1]) = 0$, so the FONC is satisfied. The Hessian at $[1, 1]$ is:

$$\begin{bmatrix} 42 & -20 \\ -20 & 10 \end{bmatrix}$$

which is positive definite, so the SONC is satisfied.

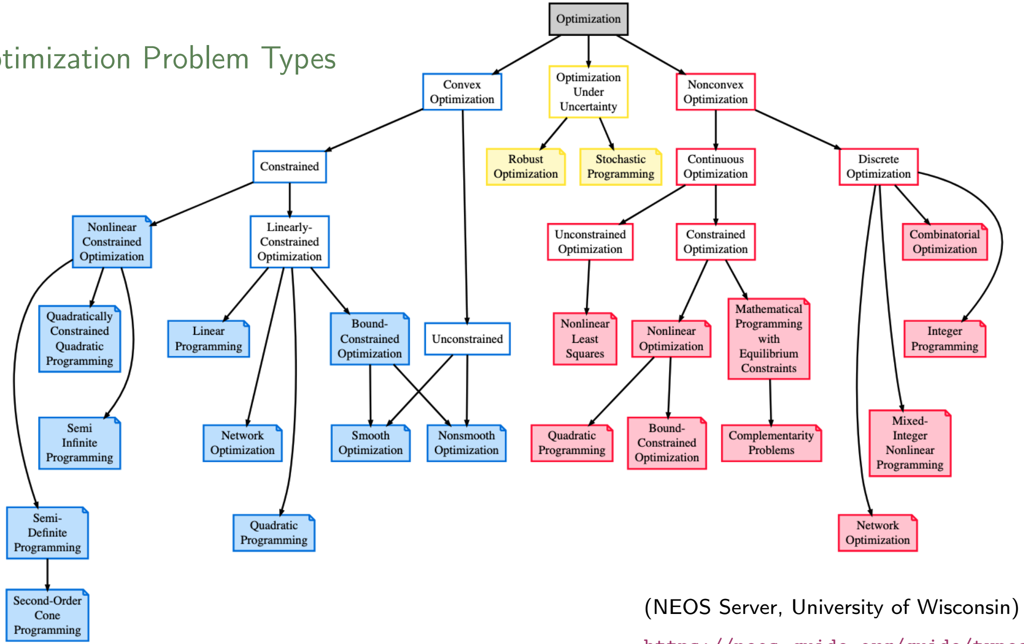
Example: Rosenbrock function



Overview

$$\begin{tikzcd}
\text{2. Derivatives} \arrow[r] & \text{4. Descent} \arrow[r] \arrow[dr] & \text{5. First-Order} & \\
\text{3. Bracketing} & & \text{6. Second-Order} & \\
\text{7. Direct} & & & \\
\text{8. Stochastic} \arrow[r] & \text{9. Population} \arrow[r] & \text{20. Expressions} & \\
\text{10. Constraints} \arrow[r] \arrow[dr] & \text{11. Linear} \arrow[r] & \text{19. Discrete} & \\
 & \text{12. Multiobjective} & & \\
\text{13. Sampling Plans} \arrow[r] & \text{14. Surrogate Models} \arrow[r] & \text{15. Probabilistic Surrogate Models} \arrow[r] & \text{16. Surrogate Optimization} \\
\text{17. Optimization under Uncertainty} \arrow[rr] & & \text{18. Uncertainty Propagation} & \\
\text{21. Multidisciplinary Design Optimization} & & &
\end{tikzcd}$$

Optimization Problem Types


Optimization
Convex Optimization
Optimization Under Uncertainty
Nonconvex Optimization
Constrained
Robust Optimization
Stochastic Programming
Continuous Optimization
Discrete Optimization
Nonlinear Constrained Optimization
Linearly-Constrained Optimization
Unconstrained Optimization
Constrained Optimization
Combinatorial Optimization
Quadratically Constrained Quadratic Programming
Linear Programming
Bound-Constrained Optimization
Unconstrained
Nonlinear Least Squares
Nonlinear Optimization
Mathematical Programming with Equilibrium Constraints
Integer Programming
Semi Infinite Programming
Network Optimization
Smooth Optimization
Nonsmooth Optimization
Quadratic Programming
Bound-Constrained Optimization
Complementarity Problems
Mixed-Integer Nonlinear Programming
Semi-Definite Programming
Quadratic Programming
Network Optimization
Second-Order Cone Programming

(NEOS Server, University of Wisconsin)

https://neos-guide.org/guide/types/

Problem classification

Different classification factors:

- Univariate $f : \mathbb{R} \to \mathbb{R}$ vs Multivariate $f : \mathbb{R}^n \to \mathbb{R}$
- Scalar-valued $f : \mathbb{R}^n \to \mathbb{R}$ vs vector-vector functions $f : \mathbb{R}^n \to \mathbb{R}^m$
- Linear vs Nonlinear
- Nonlinear: Convex vs Nonconvex, unimodal vs multimodal
- Constrained vs unconstrained
- Smooth (differentiable) vs non smooth (non differentiable)
- Continuous vs Discrete
- Deterministic vs Uncertain

Application Example

In robotics, an autonomous agent (e.g., a drone, robotic arm, or self-driving car) needs to determine an **optimal path** from a starting position to a target while satisfying constraints such as avoiding obstacles, minimizing energy consumption, or optimizing smoothness.

Formulating the Optimization Problem

The goal is to find a smooth trajectory $x(t)$ that minimizes a cost function:

$$J = \int_{t_0}^{t^f} C(\boldsymbol{x}(t), \boldsymbol{u}(t))dt$$

$\boldsymbol{x}(t)$ is the state (e.g., position, velocity), $\boldsymbol{u}(t)$ is the control input (e.g., force, acceleration), $C(\boldsymbol{x}, \boldsymbol{u})$ is the cost function, which could represent energy usage, time, or distance.

Constraints:

- Dynamic Constraints: Governed by the system's physics (e.g., Newton's laws for a robot arm or quadcopter): $\boldsymbol{x}' = f(\boldsymbol{x}, \boldsymbol{u})$
- Obstacle Avoidance: Ensures that the trajectory does not collide with obstacles: $g(\boldsymbol{x}(t)) \geq 0, \forall t$
- Boundary Conditions: The system must start and end at given positions with certain velocities.

Summary

- Optimization is the process of finding the best system design subject to a set of constraints
- Optimization is concerned with finding global minima of a function
- Minima can occur where the gradient is zero, but zero-gradient does not imply optimality