Lecture 1: Introduction to optimization

January 9, 2025

Goals for the course:

- Important consepts and theory in optimization
- Formulate engineering problems as mathematical optimization problem
- Solve the problems numerically
- Applications of optimization in control engineering - model predictive control

Optimization problems show up all over the place. That makes this course one of the most useful we take (according to the lecturer).

Optimization

Includes tools to find the best solution to a given task, formulated as a mathematical problem. Most problems of interest can only be solved using numerical methods - *numerical optimization.* This field of study builds on multi-variable calculus, numerical linear algebra, optimization theory and more. Optimization problems are common in finance, operations research, control engineering and machine learning. This course focuses on the optimization problems that show up in control applications: Model predictive control.

A "best solution' is measured using an objective function f. An optimal solution is a legal input to this function that is "better' than all other legal inputs. "Better' might mean lower or higher than other outputs, depending on the type of problem we are solving. The convention is to only look at minimization problems, because a maximization problem can easily be converted. $\max f(\mathbf{x}) = \min -f(\mathbf{x})$ The objective function is a scalar function that takes a vector input.

Conditions is an important concept for determining if a candidate solution is or is not optimal. We have two types of conditions: necessary and sufficient. Should a candidate solution $\mathbf{x}$ fail to meet the necessary conditions, we can conclude that $\mathbf{x}$ is <u>not</u> an optimal solution. If a candidate solution meets the sufficient conditions, we can conclude that $\mathbf{x}$ is the optimal solution. Sufficient conditions clearly give us more information, but they are actually often not as useful. Necessary conditions are often much easier to check.

Unconstrained optimization looks like $\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$, where f is twice differentiable. The necessary conditions are then $f'(\mathbf{x}^*) = 0$ and $f''(\mathbf{x}^*) \geq 0$. The sufficient condition for unconstrained optimization is $f''(\mathbf{x}^*) > 0$. This is something that we learn early in calculus, but not framed as a optimization problem. Least squares is also a unconstrained optimization problem.

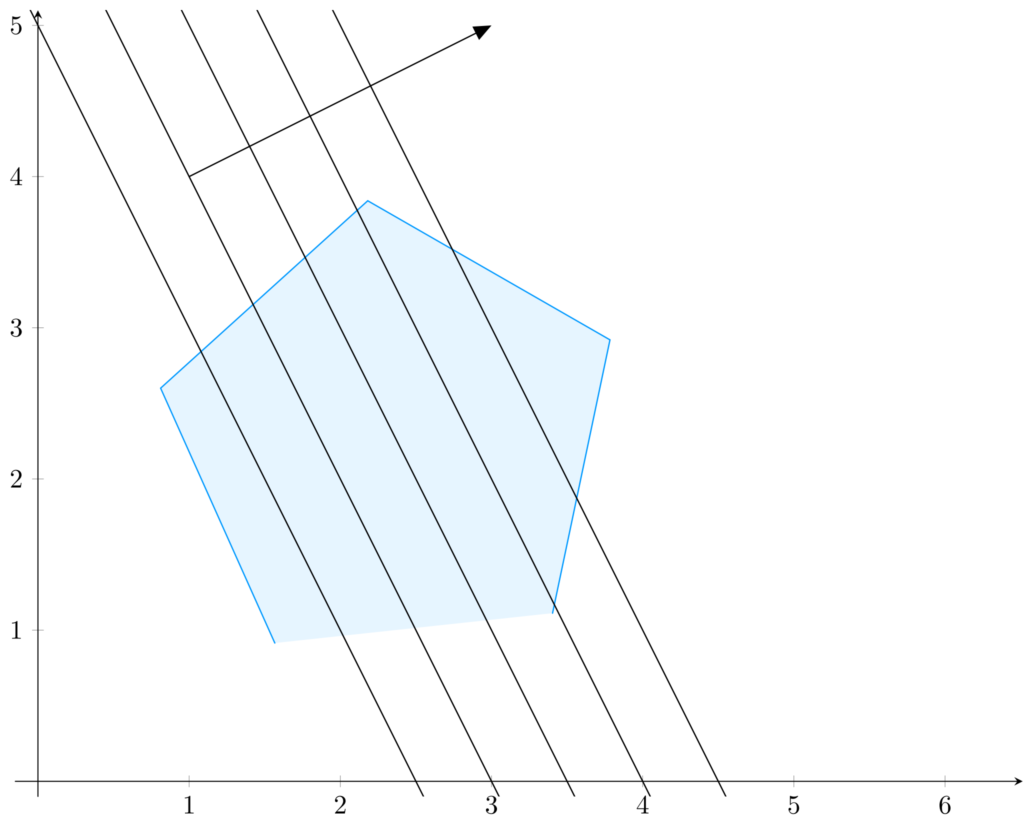
Constrained optimization problems adds constraints to the previous definition. Now we get

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) \qquad \text{s.t.} \qquad \begin{aligned} c_i(\mathbf{x}) &= 0, \quad i \in \mathcal{E} \\ c_i(\mathbf{x}) &\geq 0, \quad i \in \mathcal{I} \end{aligned}$$

Here we have equality and inequality constraints. We keep track of the indices of the constraints. The area enclosed by the constraints is called the feasible region, with symbol Ω. The points here are the possible solutions. Here we choose to define the inequality constraints with greater-than 0 functions.

Linear programming is a special case of constraint optimization where the objective function and constraints are all linear. These types of problems are easier to solve.

$$\min_{\mathbf{x}\in\mathbb{R}^n} c^\top \mathbf{x} \qquad s.t. \qquad \begin{aligned} a_i^\top \mathbf{x} &= b_i, \ i \in \mathcal{E} \\ a_i^\top \mathbf{x} &\geq b_i, \ i \in \mathcal{I} \end{aligned}.$$



The figure shows an example of a linear programming problem. The blue region is the feasible region, and the black lines are the objective function. The arrow indicating in which direction the objective function improves. Linear programming is common in economics and operations research. The symbols can often be compared with real world scenarios. The objective function describe profit (price of sale for each variable), and the constraints describe the recipe for each product, with a limit on the available resources.

Solutions to linear programming problems use the fact that the solution must be on the corner of the feasible region. That means a solver need only iterate over all the corners, and stop after finding the best one.

Here is an example solving a problem in matlab:

```
c = [-7000; -6000];         % Objective function coefficients

A = [4000, 3000;
       60,   80];           % Constraint matrix

b = [100000; 2000];         % Constraint vector

lb = [0; 0];                % Lower bounds on x
```

Solve it:

```
% Solve optimization problem:
[x, fval, exitflag, output, lambda] = linprog(c, A, b, [], [], lb, []);
```

Quadratic programming Is similar to linear programming, with the addition of a quadratic term in the objective function.

$$\min_{\mathbf{x}} \frac{1}{2}\mathbf{x}^\top \mathbf{G}\mathbf{x} + \mathbf{c}^\top \mathbf{x} \qquad s.t. \qquad \begin{array}{l} a_i^\top \mathbf{x} = b_i, \;\; i \in \mathcal{E} \\ a_i^\top \mathbf{x} \geq b_i, \;\; i \in \mathcal{I} \end{array}.$$

That one extra term changes the problem quite a bit. Instead of straigt lines (from the previous figure) we have circles making up the objective function. Quadratic programming is used in model predictive control, so it will be important in this course.

Convexity is a property. A set can be convex if:

$$x, y \in S \implies \alpha x + (1-\alpha) y \in S,\ \forall \alpha \in [0,1].$$

This has a nice geometric interpretation. A set is convex if you can pick two points and draw a straight line between them without leaving the set.

A function $f : S \to R$ is convex if S is convex and

$$\forall x, y \in S : f(\alpha x + (1-\alpha) y) \leq \alpha f(x) + (1-\alpha) f(y),\ \alpha \in [0,1].$$

The geometric interpretation here is that a straight line between two points on f is above the values of f between.

An optimization problem is convex if (1) f(x) is a convex function, and (ii) the feasible set is convex.

A feasible set is convex when equality constraints are linear, and inequality constraints are concave.

Local and global solutions $\mathbf{x}^*$ is a global solution when $f(\mathbf{x}^*) \leq f(x),\ \forall \mathbf{x} \in \Omega$

$\mathbf{x}^*$ is a local solution when $f(\mathbf{x}^*) <= f(\mathbf{x}),\ \forall \mathbf{x} \in \mathcal{N} \cap \Omega$ where $\mathcal{N}$ is a neighborhood of $\mathbf{x}^*$.

We care about convexity because convex problems are easy to solve. This is because local solutions are easy to find, and global solutions are hard (in general). But convex problems have the nice property that local solutions are also global solutions. This is one of the reason why linear programming is easier than other forms. These problems are always convex.