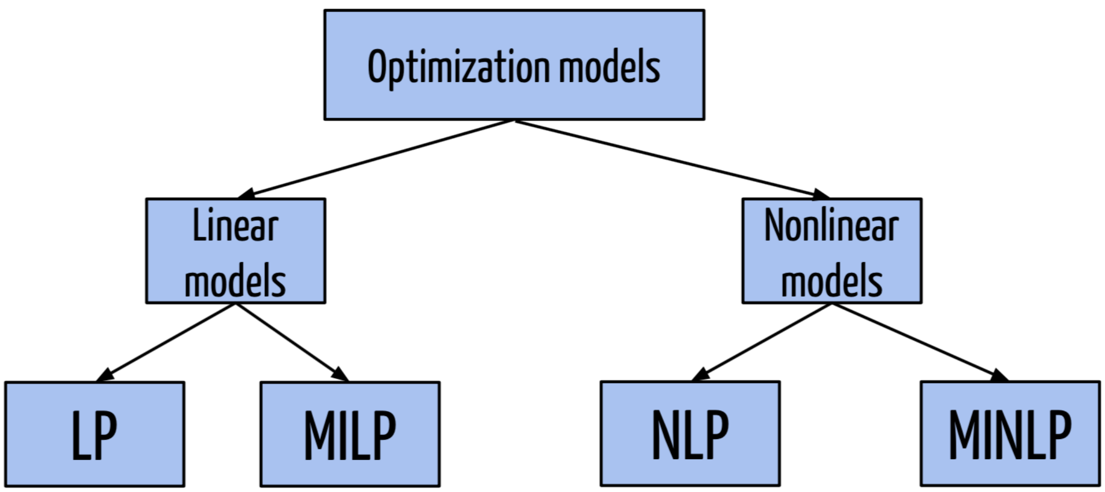
Chapter 1

Mathematical Programming

We will state main general problem classes to be associated with in these notes. These are Linear Programming (LP), Mixed-Integer Linear Programming (MILP), Non-Linear Programming (NLP), and Mixed-Integer Non-Linear Programming (MINLP).



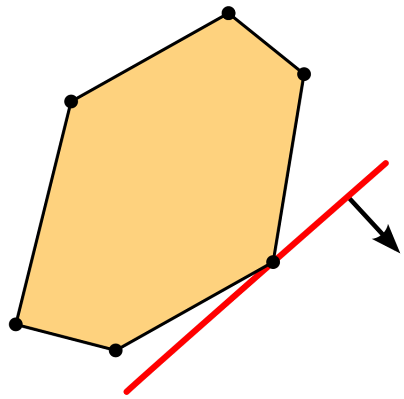
© **problem-class-diagram**[1]

Along with each problem class, we will associate a complexity class for the general version of the problem. See **??** for a discussion of complexity classes. Although we will often state that input data for a problem comes from $\mathbb{R}$, when we discuss complexity of such a problem, we actually mean that the data is rational, i.e., from $\mathbb{Q}$, and is given in binary encoding.

1.1 Linear Programming (LP)

Some linear programming background, theory, and examples will be provided in **??**.

[1] **problem-class-diagram**, from **problem-class-diagram**. **problem-class-diagram**, **problem-class-diagram**.



© **wiki/File/linear-programming.png**[2]

Figure 1.1: *Linear programming constraints and objective.*

Linear Programming (LP):

Polynomial time (P)

Given a matrix $A \in \mathbb{R}^{m \times n}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^n$, the *linear programming* problem is

$$\begin{aligned} \max \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0 \end{aligned} \tag{1.1.1}$$

Linear programming can come in several forms, whether we are maximizing or minimizing, or if the constraints are $\leq, =$ or $\geq$. One form commonly used is *Standard Form* given as

Linear Programming (LP) Standard Form:

Polynomial time (P)

Given a matrix $A \in \mathbb{R}^{m \times n}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^n$, the *linear programming* problem in *standard form* is

$$\begin{aligned} \max \quad & c^\top x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned} \tag{1.1.2}$$

Figure 1.1

[2]**wiki/File/linear-programming.png**, from **wiki/File/linear-programming.png**. **wiki/File/linear-programming.png**, **wiki/File/linear-programming.png**.

Exercise 1. *Start with a problem in form given as* (1.1.1) *and convert it to standard form* (1.1.2) *by adding at most m many new variables and by enlarging the constraint matrix A by at most m new columns.*

1.2 Mixed-Integer Linear Programming (MILP)

Mixed-integer linear programming will be the focus of Sections **??**,**??**, **??**, and **??**. Recall that the notation $\mathbb{Z}$ means the set of integers and the set $\mathbb{R}$ means the set of real numbers. The first problem of interest here is a *binary integer program* (BIP) where all n variables are binary (either 0 or 1).

Binary Integer programming (BIP):
NP-Complete
Given a matrix $A \in \mathbb{R}^{m\times n}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^n$, the *binary integer programming* problem is

$$\begin{aligned} \max \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \{0,1\}^n \end{aligned} \tag{1.2.1}$$

A slightly more general class is the class of *Integer Linear Programs* (ILP). Often this is referred to as *Integer Program* (IP), although this term could leave open the possibility of non-linear parts.

Figure 1.2

Integer Linear Programming (ILP):
NP-Complete
Given a matrix $A \in \mathbb{R}^{m\times n}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^n$, the *integer linear programming* problem is

$$\begin{aligned} \max \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \mathbb{Z}^n \end{aligned} \tag{1.2.2}$$

An even more general class is *Mixed-Integer Linear Programming (MILP)*. This is where we have n integer variables $x_1, \ldots, x_n \in \mathbb{Z}$ and d continuous variables $x_{n+1}, \ldots, x_{n+d} \in \mathbb{R}$. Succinctly, we can write this as $x \in \mathbb{Z}^n \times \mathbb{R}^d$, where $\times$ stands for the *cross-product* between two spaces.

[3]**wiki/File/integer-programming.png**, from **wiki/File/integer-programming.png**. **wiki/File/integer-programming.png**, **wiki/File/integer-programming.png**.

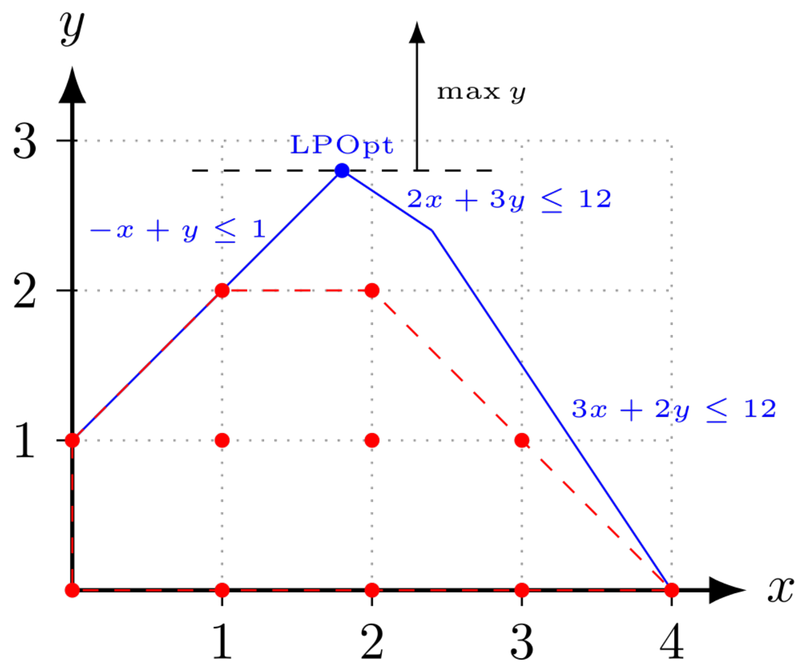


Figure 1.2: *Comparing the LP relaxation to the IP solutions.*

Below, the matrix A now has $n+d$ columns, that is, $A \in \mathbb{R}^{m\times n+d}$. Also note that we have not explicitly enforced non-negativity on the variables. If there are non-negativity restrictions, this can be assumed to be a part of the inequality description $Ax \leq b$.

Mixed-Integer Linear Programming (MILP):
NP-Complete
Given a matrix $A \in \mathbb{R}^{m\times(n+d)}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^{n+d}$, the *mixed-integer linear programming* problem is

$$\begin{aligned} \max \quad & c^\top x \\ \text{s.t.} \quad & Ax \leq b \\ & x \in \mathbb{Z}^n \times \mathbb{R}^d \end{aligned} \tag{1.2.3}$$

1.3 Non-Linear Programming (NLP)

NLP:
NP-Hard
Given a function $f(x)\colon \mathbb{R}^d \to \mathbb{R}$ and other functions $f_i(x)\colon \mathbb{R}^d \to \mathbb{R}$ for $i = 1, \ldots, m$, the *nonlinear programming* problem is

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & f_i(x) \leq 0 \quad \text{for } i = 1, \ldots, m \\ & x \in \mathbb{R}^d \end{aligned} \tag{1.3.1}$$

Nonlinear programming can be separated into convex programming and non-convex programming. These two are very different beasts and it is important to distinguish between the two.

1.3.1 Convex Programming

Here the functions are all **convex!**

Convex Programming:
Polynomial time (P) (typically)
Given a convex function $f(x)\colon \mathbb{R}^d \to \mathbb{R}$ and convex functions $f_i(x)\colon \mathbb{R}^d \to \mathbb{R}$ for $i = 1, \ldots, m$, the *convex programming* problem is

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & f_i(x) \leq 0 \quad \text{for } i = 1, \ldots, m \\ & x \in \mathbb{R}^d \end{aligned} \tag{1.3.2}$$

Example 1: Convex programming is a generalization of linear programming. This can be seen by letting $f(x) = c^\top x$ and $f_i(x) = A_i x - b_i$.

1.3.2 Non-Convex Non-linear Programming

When the function f or functions f_i are non-convex, this becomes a non-convex nonlinear programming problem. There are a few complexity issues with this.

IP as NLP As seen above, quadratic constraints can be used to create a feasible region with discrete solutions. For example

$$x(1 - x) = 0$$

has exactly two solutions: $x = 0, x = 1$. Thus, quadratic constraints can be used to model binary constraints.

Binary Integer programming (BIP) as a NLP:
NP-Hard
Given a matrix $A \in \mathbb{R}^{m \times n}$, vector $b \in \mathbb{R}^m$ and vector $c \in \mathbb{R}^n$, the *binary integer programming* problem is

$$\begin{aligned} \max \quad & c^\top x \\ \text{s.t.} \quad & Ax \le b \\ & \cancel{x \in \{0,1\}^n} \\ & x_i(1 - x_i) = 0 \quad \text{for } i = 1, \dots, n \end{aligned} \tag{1.3.3}$$

1.4 Mixed-Integer Non-Linear Programming (MINLP)

1.4.1 Convex Mixed-Integer Non-Linear Programming

1.4.2 Non-Convex Mixed-Integer Non-Linear Programming