ACM 41030 / ACM 40990

Chapter 1 — Introduction to Optimization

Why? In many ML algorithms, it is required to solve an optimization problem. But these algorithms are now mature, and involve "only" running some code e.g. in Python. But what if your code throws up an error – we should try to understand these error messages. Otherwise, we run the risk of GIGO.

The module has two parts:

- Weeks 1–7: Continuous Optimization
- Weeks 8–12:
  - ACM 40990: applications of continuous optimization to ML.
  - ACM 41030: We look at constrained optimization.

Wider context:

- Investment – portfolio management
- Manufacturing
- "Operations Research"

In nature, optimization is important:

- Systems tend to equilibrium — a state of minimum energy.
- Fermat's principle of least time — rays of light follow paths that minimize the travel time.

These observations can be used to make optimization algorithms — e.g. <u>simulated annealing</u>.

<u>Terminology</u>:

- $\underline{x} \in \mathbb{R}^n$ is the vector of <u>variables</u>, also referred to as the <u>unknowns</u>, or the parameters.
- f is the <u>cost function</u> (objective function). This is a scalar-valued function:
$$f : \Omega \to \mathbb{R}. \quad (\Omega \subset \mathbb{R}^n).$$
The goal in optimization is to minimize f.
- c_i are constraint functions. These are scalar-valued, these define certain equations or inequalities. The vector $\underline{x}$ has to satisfy these equations and/or inequalities.

Fundamental Optimization Problem (OP)

Constrained:

$$\min_{\underline{x} \in \mathbb{R}^n} f(\underline{x}) \quad \text{subject to} \quad \begin{cases} c_i(\underline{x}) = 0, & i \in \mathcal{E} \\ c_i(\underline{x}) \geq 0, & i \in \mathcal{I} \end{cases}$$

where $\mathcal{E}$ and $\mathcal{I}$ are sets of indices for equality and inequality constraints, respectively.

Unconstrained:

$$\min_{\underline{x} \in \mathbb{R}^n} f(\underline{x})$$

✓

Numerical algorithms:
- Line Search
- Newton
- Trust-Region

Example: ~~Exa~~

$$\min \ (x_1 - 2)^2 + (x_2 - 1)^2$$

$$\text{Subject to:} \quad \begin{cases} x_1^2 - x_2 \leq 0 \\ x_1 + x_2 \leq 2. \end{cases}$$

Cost function: $f(\underline{x}) = (x_1 - 2)^2 + (x_2 - 1)^2$,

$\underset{\sim}{x} = (x_1, x_2)^T$.

Constraint functions:

$c_1(\underline{x}) = -x_1^2 + x_2$, $\quad c_1(x) \geq 0.$

$c_2(\underline{x}) = -x_1 - x_2 + 2$, $\quad c_2(x) \geq 0.$

$\mathcal{E} = \emptyset$, $\quad \mathcal{I} = \{1, 2\}.$

<u>Solution</u> — via graphical methods. Re-write OP as:

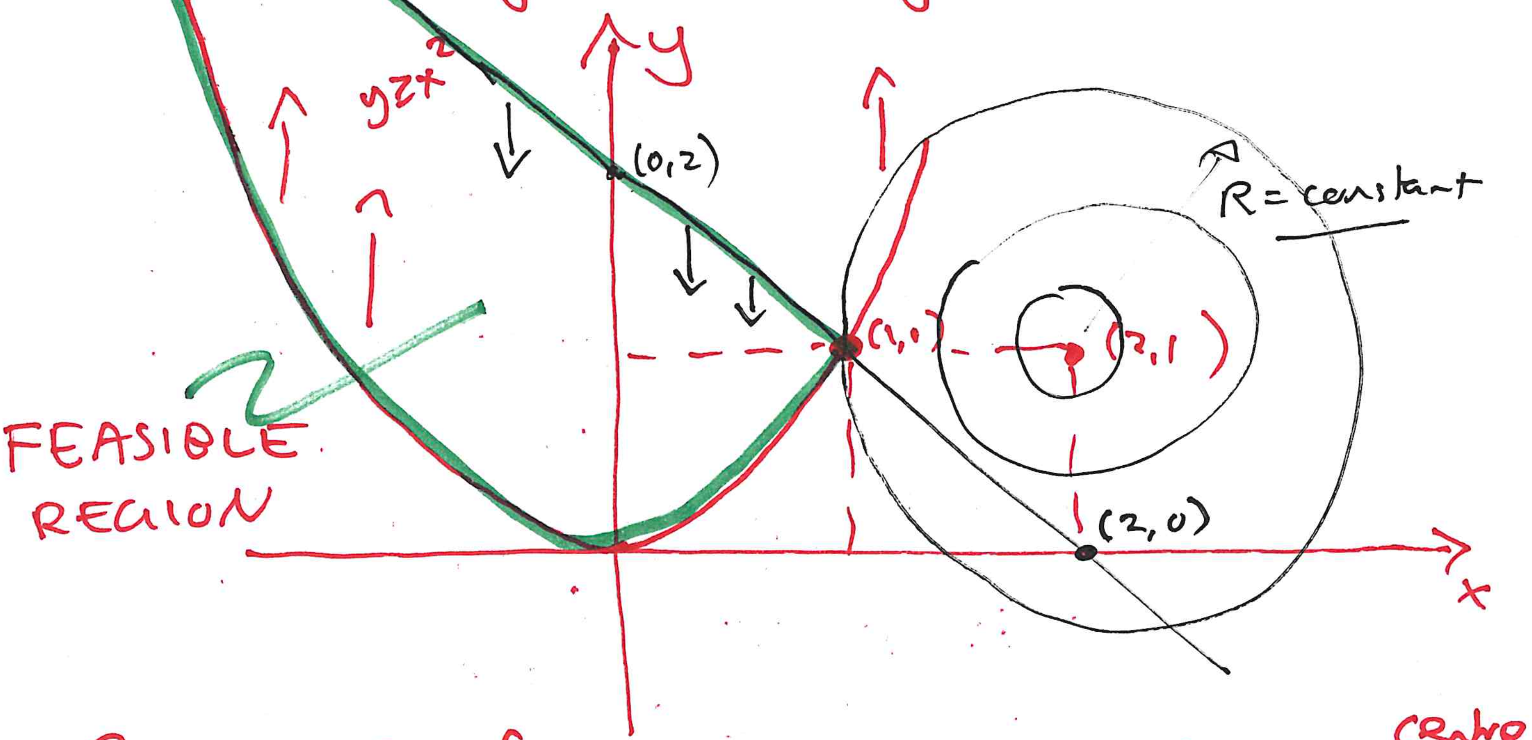
$f(x, y) = (x-2)^2 + (y-1)^2$

$y = 2 - x.$

$y = 0. \; x = 2$

c_1: $x^2 - y \leq 0 \Rightarrow y \geq x^2.$

c_2: $x + y \leq 2 \Rightarrow y \leq 2 - x.$



Contours of f are circles passing through centre (2,1)

To minimize f, we want to make the radius as small as possible.

But we have to stay in the feasible region. Graphically, the minimum is on the edge (boundary) of the feasible region, where the constraints are equality constraints.

$c_1 = 0 \implies y = x^2$.

$c_2 = 0 \implies y = 2 - x$.

<u>Solve</u>: $x^2 = 2 - x \implies x = 1$.

$y = 2 - x \implies y = 1$.

$\underline{x}_* = (1, 1)^T$.

This is checked in the notes in Matlab.

<u>Definition</u>: The feasible region is the set of all points satisfying the constraints.

For inequality constraints, the feasible region is the ~~region~~ area between the constraint boundaries.

<u>Notation</u>: We use $\underline{x}_*$ to denote a solution of the optimization problem.

<u>Why numerical optimization?</u>

- 2D problems are easy to solve with pen and paper.
- But not 10D problems or 100D problems – hence a numerical approach is necessary.

We will look at key numerical algorithms:

- Line Search (Steepest Descent)
- Newton, Quasi-Newton
- Trust-Region Methods.

More introduction on Thursday.