CORK INSTITUTE OF TECHNOLOGY
INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

Computing@CIT

Decision Analytics

Lecture 3: Mathematical Optimisation

Mathematical Optimisation

Given a domain D and an objective function

$$f: D \to \mathbb{R}$$

find $\hat{x} \in D$ that minimises f, i.e. $\forall x \in D: f(\hat{x}) \leq f(x)$

(for maximisation problems replace f with $-f$)

Mathematical Optimisation

- Different optimisation techniques are applicable depending on
  - The properties of the domain D
  - The shape of the objective function f

Mathematical Optimisation

- Special case 1:
  - Domain is $\mathrm{D} = \mathbb{R}^n$
  - Objective function f is differentiable

- Then a necessary condition for a local minimum/maximum of the objective function f is

$$\frac{\partial}{\partial x} f = 0$$

- This condition can be either solved directly or by neighbourhood search algorithms (*gradient descent, simulated annealing*)

Mathematical Optimisation

- Special case 2:
  - Domain $\mathrm{D} = \{x \in \mathbb{R}^n \,|\, g(x) = 0\}$ is a manifold
  - Objective function f and implicit constraint g is differentiable

- Then a necessary condition for a local minimum/maximum of f is

$$\frac{\partial}{\partial x}(f + \lambda^T g) = 0$$
$$\frac{\partial}{\partial \lambda}(f + \lambda^T g) = 0$$

- This conditions essentially add the Lagrange parameter λ to the search domain and reduce the problem to the previous case

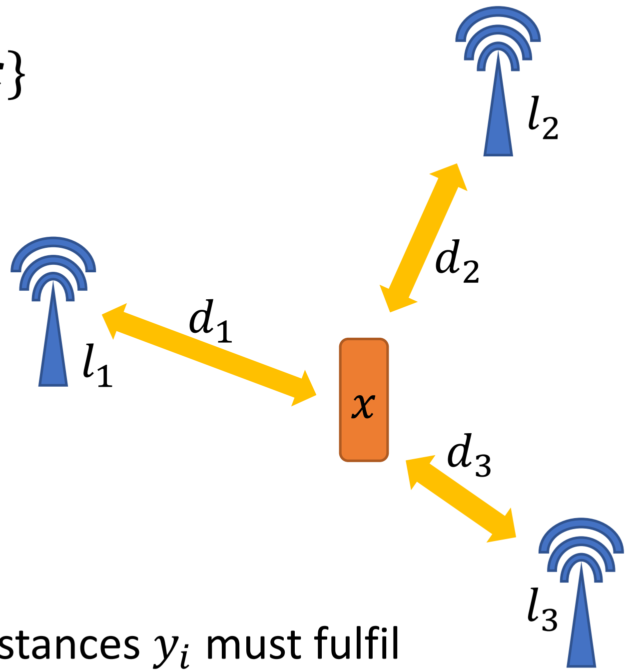
Mathematical Optimisation

- Special case 3:
  - Domain $\mathrm{D} = \{x, y \in \mathbb{R}^{n+m} | \, Ax + By = c\}$
  - Objective function $f = (y - \bar{y})^2$

- This is a special case of the previous problem with a quadratic objective function and a linear manifold as domain
- The Lagrange function in this case is
$$L(x, y, \lambda) = (y - \bar{y})^2 + \lambda^T (Ax + By - c)$$
- Its derivatives form a system of linear equations, that can be solved for (x, y, λ)
$$\frac{\partial}{\partial x} L(x, y, \lambda) = \lambda^T A = 0$$
$$\frac{\partial}{\partial y} L(x, y, \lambda) = 2(y - \bar{y}) + \lambda^T B = 0$$
$$\frac{\partial}{\partial \lambda} L(x, y, \lambda) = Ax + By - c = 0$$

Mathematical Optimisation

- Special case 3:
  - Domain $\mathrm{D} = \{x, y \in \mathbb{R}^{n+m} | \, Ax + By = c\}$
  - Objective function $f = (y - \bar{y})^2$



- Example: Radio location
  - Given the cell-tower locations $l_1, \dots, l_n$
  - and distance measurements $\bar{d}_1, \dots, \bar{d}_n$
  - Where is the mobile device x?

  - Using squared distances $\bar{y}_i = \bar{d}_i^2$ the "true" distances y_i must fulfil

$$\left| l_i - x \right|^2 = y_i$$

  - Approximating $x^T x \approx x_0^T x_0 + 2x_0^T(x - x_0)$ these can be re-phrased as follows

$$2\left(x_0^T - l_i^T\right)x - y_i = x_0^T x_0 - l_i^T l_i$$

  - Minimising the residual between measured distances and "true" distances $(y - \bar{y})^2$ has the structure of the optimisation problem stated above

Mathematical Optimisation

- Special case 4:
  - Domain $\mathrm{D} = \{x \in \mathbb{R}^n | x \geq 0 \wedge Ax \leq b\}$ is bound by linear constraints
  - The objective function is linear $f = c^T x$

- This can be solved by Linear Programming (LP)
- Very efficient algorithms exist to solve LP problems, the Google OR Tools provide a front-end for various solvers, e.g.:

```
from ortools.linear_solver import pywraplp

solver = pywraplp.Solver('LP',
                         pywraplp.Solver.GLOP_LINEAR_PROGRAMMING)
```

Mathematical Optimisation

- Special case 4:
  - Domain $\mathrm{D} = \{x \in \mathbb{R}^n | x \geq 0 \wedge Ax \leq b\}$ is bound by linear constraints
  - The objective function is linear $f = c^T x$

- Example: Combined Heat and Power (CHP) optimisation
  - A CHP can be set to run at $0 \leq p_i \leq 1$ percent of its total capacity
  - Its maximum thermal output is T_i and its maximum electrical output is E_i
  - The goal is to satisfy thermal demand $\sum_i p_i T_i \geq D_T$ as well as electrical demand $\sum_i p_i E_i \geq D_E$ while minimising fuel cost $\sum_i p_i c_i$
  - Excess thermal energy can be easily dumped, however the grid imposes a maximum amount of electrical energy it can take, i.e. $\sum_i p_i E_i - D_E \leq M_E$

Mathematical Optimisation

- Special case 5:
  - Domain $\mathrm{D} = \{x \in \mathbb{N}^n | Ax \leq b\}$ is integer only and bound by linear constraints
  - The objective function is linear $f = c^T x$

- Despite its similarity to case 4 (LP), mixed integer linear programs (MIP) are NP-hard
- MIP algorithms are available, the Google OR Tools provide a unified front-end for various solvers, e.g.:

```
from ortools.linear_solver import pywraplp

solver = pywraplp.Solver(MIP',
                         pywraplp.Solver.CBC_MIXED_INTEGER_PROGRAMMING)
```

Mathematical Optimisation

- Special case 5:
  - Domain $\mathrm{D} = \{x \in \mathbb{N}^n | Ax \leq b\}$ is integer only and bound by linear constraints
  - The objective function is linear $f = c^T x$

- Examples: Combined Heat and Power (CHP) optimisation
  - Same as before, but what happens if the CHP output cannot be adjusted from 0-100%, but only has an ON/OFF switch?
  - More important, if this is integrated into a bigger grid management problem, where some generators and/or consumers can only be switched on and off altogether

Mathematical Optimisation

- Special case 6:
  - Domain $\mathrm{D} \subset \mathbb{N}^n$ is integer only and subject to constraints
  - The objective function $f = 1$ is constant, i.e. all feasible solutions are equally valid

- In general, Constraint Satisfaction Problems (CSP) are also NP-hard
- The goal is (only) to find a feasible solution, not to find the best (e.g. n-queens problems)
- Constraint Programming (CP) can be used to find a solution
- The Google OR Tools provide the CP-SAT solver for these problems:

```python
from ortools.sat.python import cp_model
```

Mathematical Optimisation

- Special case 7:
  - Domain $\mathbb{D} \subset \mathbb{N}^n$ is integer only and subject to constraints
  - A generic objective function f is defined to quantify the *cost* of each feasible solution

- Even more generic than case 6, hence also NP-hard
- Can be used to solve Combinatorial Optimisation Problems, but be aware that sometimes more efficient solutions exist (e.g. Max-Flow)
- Again, Constraint Programming can often be used for finding a good solution
- The Google OR Tools provide the CP-SAT solver for these problems:

```
from ortools.sat.python import cp_model
```

Mathematical Optimisation

- Special case 7:
  - Domain $\mathrm{D} \subset \mathbb{N}^n$ is integer only and subject to constraints
  - A generic objective function f is defined to quantify the *cost* of each feasible solution

- Example: Renewable integration into the energy grid
  - Issues: Wind turbines can be switched on and off, solar panels often cannot, backup power generation is expensive
  - Depending on the weather conditions and energy demand, different combinations of renewables need to be integrated into the grid in order to maximise profit while keeping the grid stable

Mathematical Optimisation

- Direct solutions exist for some (important) special cases, including
  - Gauss-Helmert model (special case 3)
  - Linear Programming (special case 4)
- Integer domain problems tend to be more difficult than real domain problems
- Constraint Programming (special cases 6 & 7) is a very versatile tool for solving many difficult integer optimisation problems
- However: CP is **NOT** the method of choice for every problem, in particular if the problem can be solved more efficiently using one of the other specialised approaches
- Neighbourhood search algorithms can be used where a meaningful neighbourhood relation in the parameter space can be established (special cases 1 & 2)

Thank you for your attention!