Optimisation and Operations Research

Lecture 23: Revision

Matthew Roughan
<matthew.roughan@adelaide.edu.au>
http://www.maths.adelaide.edu.au/matthew.roughan/notes/OORII/

School of Mathematical Sciences,
University of Adelaide

August 13, 2019

Section 1

Revision

Revision

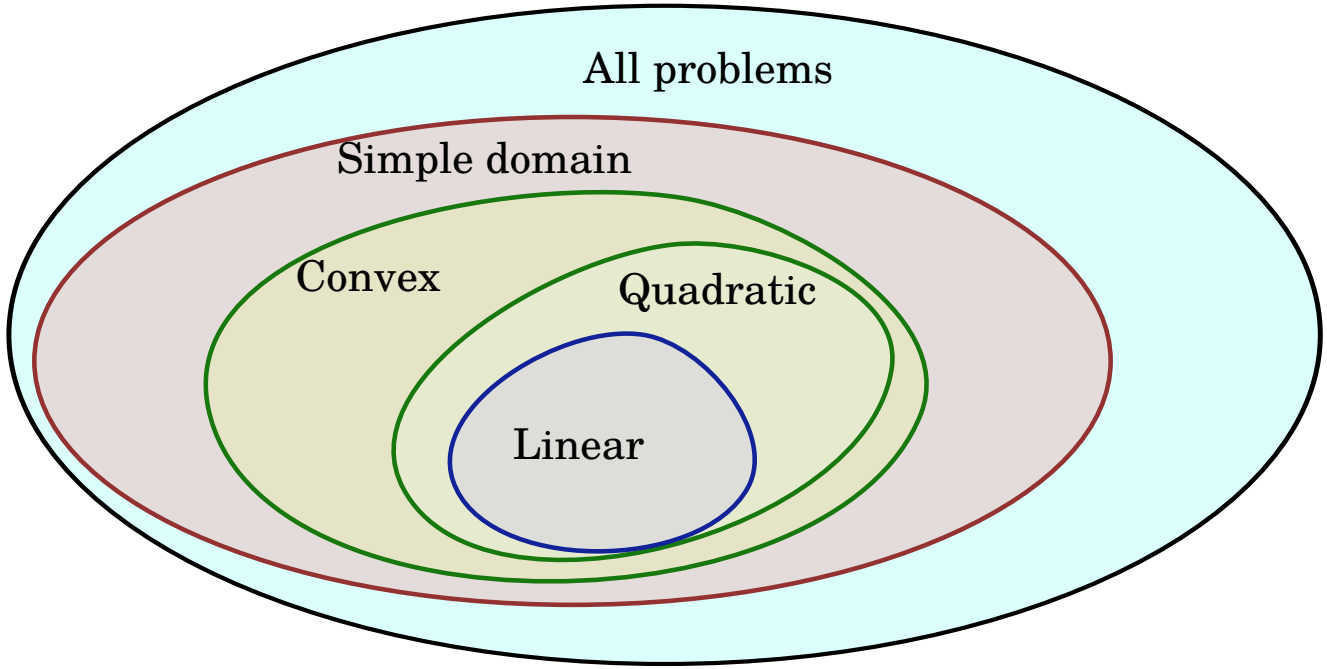
Translation

The hardest part of optimisation is often translating a messy real-world problem into mathematics

- Break it down
  - What are my variables?
    - ★ what can I control?
    - ★ what are the decisions I make?
  - What is my objective?
    - ★ express what you want to achieve in terms of a function of the variables
  - What are my constraints?
    - ★ what are the limits on the variables?

Problem Classification

- First job is often to work out what type of problem you are solving



- Integer or Continous
  - sometimes becomes integer when we introduce extra artificial variables
- Other classifications we haven't covered in this course in detail

Translation

We have been dealing with LPs and ILPs

- Variables are numbers
  - we put them in a vector $\mathbf{x}$
- The objective is a linear function of the variables
  - we can always write it as max or min of

$$z = \sum_i c_i x_i$$

    where the x_i are the variables, and c_i are some numbers
- The constraints are linear inequalities or equalities of the variables
  - can always be written into standard form $A\mathbf{x} = \mathbf{b}$
  - don't forget non-negativity

Approximation

- All real problems have a tradeoff between
  - realism
  - simplicity

  We need to balance these
- We are doing linear programming
  - sometimes the problem will be non-linear
  - often easier to approximate, than to try non-linear methods
- You need to learn "tricks"
  - linear segment approximation
  - what parts of a curve "matter"
  - introducing extra variables

Solution Methods

- Simplex (for LPs)
  - plus duality and compementary slackness
  - sensitivity analysis
- Heuristics (for ILPs)
  - greedy
  - GAs
- Branch and Bound (for ILPs)

Complexity

- An important part of using any algorithm is understanding its computational complexity
  - how long it will take to run
- Often we describe this with big-O notation
  - know how to derive
  - know the limitations

Coding

You need to be able to program to be able to deal with real problems

- Matlab
  - very good general purpose tool
  - shouldn't be the only language you know!
- AMPL (with lpsolve)
  - specific to optimisation
  - much more powerful than I have shown
- There are many others, but the above are the ones we have used

Exam notes

- You are allowed to take in some notes
  - 2 pages
  - double-sided
  - hand-written
- Standard restriction on calculators
  - Calculators without remote communications facilities are permitted.
- English and foreign-language dictionaries may be used

Further reading I