Inmas 2021: Modeling and Optimization

Session 1: LP Formulations and Gurobi

Alexander Estes

Optimization problem I

A typical (continuous) optimization problem:

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

subject to:

$$g_1(x) \leq 0$$
$$\dots$$
$$g_k(x) \leq 0$$

Optimization problem II

Objective function

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

subject to:

$$g_1(x) \leq 0$$
$$\dots$$
$$g_k(x) \leq 0$$

Optimization problem III

Constraints

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

subject to:

$$\begin{aligned} \boldsymbol{g_1(x)} &\leq 0 \\ &\dots \\ \boldsymbol{g_k(x)} &\leq 0 \end{aligned}$$

Optimization problem IV

Terminology:

- x is *feasible* if satisfies all the constraints.
- *feasible region* is set of all feasible points.
- x is *optimal* if satisfies all constraints and $f(x) \leq f(y)$ for any other feasible solution y.

Goal:

- find an optimal solution.

Why Optimization?

Optimization:

- Provides mathematical approach to making best decisions.
- Applicable to many settings:
  - Production scheduling
  - Airline crew scheduling
  - Sports scheduling
  - Portfolio selection
  - Telecommunication network design
  - Design of radiation treatments
  - Molecular biology
  - More

Formulating optimization problems I

Real-world problems are typically broad, sometimes vague questions:

- How should I schedule my crew?
- How much of each product should I order?
- How should I design my telecommunication network?
- What is the best way to schedule a baseball season?

Formulating optimization problems II

Steps in building a formulation:

1. What decisions need to be made?
   - Identify decisions.
   - Define variables that represent decisions as numerical vectors.
2. What constraints are there on decisions?
   - Define constraints.
3. How do we measure quality of decisions?
   - Define an objective function.

Pie-eating contest I

Max is in a pie-eating contest.

- Contest lasts 1 hour.
- Max can eat a torte in 2 minutes.
- Max can eat an apple pie in 3 minutes.
- Tortes are worth 4 points.
- Pies are worth 4 points.

How can Max get the most points?

Pie-eating contest II

What decisions do we have?

1. How many tortes should Max eat?
2. How many apple pies should Max eat?

Decision variables:

- Let x be the number of tortes eaten by Max.
- Let y be the number of pies eaten by Max.

Pie-eating contest III

What constraints are there on decisions?

- Time limit:

$$2x + 3y \leq 60$$

- Max cannot eat negative quantities of tortes:

$$x \geq 0$$

- Max cannot eat negative quantities of apple pies:

$$y \geq 0$$

Pie-eating contest IV

What constraints are there on decisions?

- Time limit:

$$2x + 3y \leq 60$$

- Max cannot eat negative quantities of tortes:

$$x \geq 0$$

- Max cannot eat negative quantities of apple pies:

$$y \geq 0$$

Pie-eating contest V

How do we measure the quality of a decision?

- Number of points received in contest:

$$4x + 5y$$

Pie-eating contest VI

Putting it all together:

$$\max \quad 4x + 5y$$

s.t.

$$\begin{aligned} 2x + 3y &\le 60 \\ x &\ge 0 \\ y &\ge 0 \end{aligned}$$

Example feasible solution: $x = 10, y = 10$.
Optimal solution $x = 30$, $y = 0$.

Manufacturing example I

There is a manufacturing company.

- The company makes n different products using m different materials.
- b_i is the amount of raw material i available.
- c_j be the revenue of producing a single unit of product j.
- a_{ij} be the amount of raw material i used in one unit of product j.

How can the company maximize revenues?

Manufacturing example II

Decision variables:

- x_j - units of product j that are produced.

Constraints:

- Cannot use more resources than available:

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \text{ for } i \in [m]$$

- We cannot produce negative quantities of products:

$$x_i \geq 0$$

Objective:

$$\sum_{i=1}^{n} c_i x_i$$

Manufacturing example III

Decision variables:

- x_i - units of product i that are produced.

Constraints:

- Cannot use more resources than available (matrix form):

$$Ax \leq b$$

- We cannot produce negative quantities of products:

$$x \geq 0$$

Objective (vector form):

$$c^{\mathsf{T}} x$$

Manufacturing example IV

Altogether:

$$\max c^{\mathsf{T}} x$$

s.t.

$$\begin{aligned} Ax &\leq b \\ x &\geq 0. \end{aligned}$$

Linear programming I

Not all optimization problems are tractable.

- Need to focus on special tractable cases.

Linear programming II

Linear program:

- Special case of optimization problem
- Affine constraints; i.e. each takes the form

  $$\alpha^\intercal x \leq \beta$$

  for vector α and constant β.
- Linear objective function; i.e. takes the form

  $$c^\intercal x$$

  for some vector c.
- Both previous examples are linear programs.
- Very useful in practice and can be solved efficiently.

Optimality I

There are three possibilities:

- LP has an optimal solution.
- LP is infeasible; e.g.

$$
\begin{aligned}
\min x \\
x \leq 1 \\
x \geq 2 \\
x \geq 0
\end{aligned}
$$

- LP is *unbounded*; e.g.

$$
\begin{aligned}
\max x \\
x \geq 0
\end{aligned}
$$

Optimality II

Note:

- An LP can only be unbounded if its feasible region is unbounded.
- Unbounded feasible region does not imply unbounded:

$$\min x$$
$$x \geq 0$$

Optimality III

In more general settings, an optimization problem may be feasible and bounded while having no optimal solution, e.g.

$$\min_{x \in [1,\infty)} \frac{1}{x}$$

This can never happen in an LP!

Solving LPs with Software I

Many software options available:

- CPLEX
- COIN-OR
- FICO Xpress
- GLPK
- Gurobi
- Many others.

Vary in features, cost, open source/proprietary.

Solving LPs with Software I

Three options for using solvers:

- Solver interfaces with programming languages.
- Modeling languages.
- Formatted text files.

Solving LPs with Software II

Solver interfaces with programming languages.

- Write code in programming language of choice.
- Libraries implemented in that programming language allow you to access solver.
- Main advantage: easy to include optimization model in program.

Solving LPs with Software III

Modeling languages.

- A language for formulating models in a way that a solver can understand.
- Higher level than solver interfaces.
- Can be commercial/open source.
- Examples: OPL (bundled with CPLEX), GAMS, AMPL
- Advantage: more intuitive than interfaces with programming languages.
- Disadvantage: harder to incorporate into program.

Solving LPs with Software IV

Formatted text files.

- Formulation is saved as a text file with specific format.
- Solvers can read those text files.
- Advantage: Can avoid dependencies on specific languages/solvers.
- Disadvantages: text file formats are not entirely standardized; more difficult and less intuitive to automate.
- Example use: storing a library of optimization problems

Gurobi example

See “piecontest.py”, “productionplanning.py”

Gurobi summary I

Summary of the steps that we took:

1. Import Gurobi Python interface

```
from gurobipy import *
```

2. Create empty model.

```
m = Model("mymodel")
```

3. Add variables.

```
myvar = m.addVar(type, name, lb, ub)
```

4. Add objective.

```
m.setObjective(function, sense)
```

Gurobi summary II

5. Add constraints.

   ```
   m.addConstr(inequality)
   ```

6. Solve model.

   ```
   m.optimize()
   ```

7. Check model status.

   ```
   status = m.getAttr("Status")
   ```

8. Retrieve solution.

   ```
   sol_val = myvar.getAttr("X")
   ```