Linear Programming

(5531)

Week 01 Introduction

Prof. Sheng-I Chen

http://www.iem-omglab.nctu.edu.tw/OMG_LAB/

September 17, 2020

Agenda

- Teaching plan
  - Syllabus / contents / schedule
  - Textbooks
  - Grading and class rules
- Introduction of Linear Programming
  - Purposes
  - History
  - LP formulations
  - Solvers

Course Units

Units	Contents
Preliminary	Set theory, linear algebra, and polyhedral theory
Basic Theory	Simplex method, degeneracy, duality, sensitivity analysis, LU-factorization, and implementation issues
Interior-point Methods	Convex analysis, Farkas' lemma, central path, barrier problem, Lagrange multipliers, path-following method, and KKT system
Extensions	Network flow problems, Integer programming and etc.
Solver tutorial	IBM CPLEX installations, the procedure of solve LPs in CPLEX, and the Concert API

Tentative Schedule

Week	Contents
1	Introduction
2	Set theory, linear algebra Visual studio & IBM CPLEX installation CPLEX LP file format & Sample code (TA)
3	Simplex methods
4	Degeneracy
5	Simplex method in matrix notation
6	Midterm exam I
7	Duality theory
8	Efficiency of the simplex algorithm
9	Computational exercise
10	Sensitivity analysis

Week	Contents
11	Convex analysis
12	Midterm exam II
13	Interior-point methods KKT system
14	Decomposition principle Network flow problems
15	Network flow problems Integer programming
16	Final Exam
17	Optional (T.B.D.)
18	Optional (T.B.D.)

Textbook and Reference

- **Textbook:** Linear Programming: Foundations and Extensions 3rd Edition, Robert J. Vanderbei., Springer, 2008 (Available at the online library)
- **Reference:** Linear Programming and Network Flows 4th Edition, Mokhtar S. Bazaraa, John J. Jarvis, and Hanif D. Sherali, Wiley, 2009

Finding the Textbook at Webpac

Grading

- Homework: 25%
- In-class performance: 15%
- Midterm exams: 40% (Each worth 20%)
- Final exam: 20%

Class Rules

- All assignments must be done **independently** and returned on time
- Don't copy any work from the internet or other students. **Plagiarism will receive very serious consequences**
- Cellphone or any electronic device (except with my permission) is not allowed in the class
- Everyone is responsible for maintaining high quality of the learning environment. **Chatting or discussing topics not related are strictly prohibited.** You should raise your hands if any question.

Learning Tips

- Reading (theory)
- Coding (realize your understanding into practice)
- Thinking and answering my questions in the class

Linear Programming

- What is Linear Programming?
  - Objective function and constraints are linear in the variables
- What is the Linear Programming Problem?
  - To find a best solution that maximize or minimize a linear objective function subjects to linear constraints
  - Decision variables:

$$x_j \qquad , j = 1,2,\ldots,n$$

  - Linear objective function:

$$z = \sum_{j=1}^{n} c_j x_j$$

  - Linear constraints:

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \qquad i = 1,..,m$$

Purposes of Linear Programming

- To make decisions on operations, product design, organization, manufacturing, transportation, finance, service industry, and etc.
- To save money and time through modeling and solving LPs
- To provide insights into the optimal solution as a benchmark for validation processes in practical applications

History of Linear Programming

- 1939, L.V. Kantorovich
  - Algorithm for solving certain classes of LP problems
- 1947, G.B. Dantzig
  - The Simplex method for solving general LP problems
- 1979, L. Khachiyan
  - The first algorithm to solve LP in polynomial running time, Ellipsoid algorithm
- 1984, N. Karmarkar
  - Solve LP even faster, Interior point method

The Karmarkar Breakthrough, 1984

Breakthrough in Problem Solving

By JAMES GLEICK

A 28-year-old mathematician at A.T.&T. Bell Laboratories has made a startling theoretical breakthrough in the solving of systems of equations that often grow too vast and complex for the most powerful computers.

The discovery, which is to be formally published next month, is already circulating rapidly through the mathematical world. It has also set off a deluge of inquiries from brokerage houses, oil companies and airlines, industries with millions of dollars at stake in problems known as linear programming.

Faster Solutions Seen

These problems are fiendishly complicated systems, often with thousands of variables. They arise in a variety of commercial and government applications, ranging from allocating time on a communications satellite to routing millions of telephone calls over long distances, or whenever a limited, expensive resource must be spread most efficiently among competing users. And investment companies use them in creating portfolios with the best mix of stocks and bonds.

The Bell Labs mathematician, Dr. Narendra Karmarkar, has devised a radically new procedure that may speed the routine handling of such problems by businesses and Government agencies and also make it possible to tackle problems that are now far out of reach.

"This is a path-breaking result," said Dr, Ronald L. Graham, director of mathematical sciences for Bell Labs in Murray Hill, N.J.

"Science has its moments of great progress, and this may well be one of them."

Because problems in linear programming can have billions or more possible answers, even high-speed computers cannot check every one. So computers must use a special procedure, an algorithm, to examine as few answers as possible before finding the best one — typically the one that minimizes cost or maximizes efficiency.

A procedure devised in 1947, the simplex method, is now used for such problems,

Continued on Page A19, Column 1

THE NEW YORK TIMES, November 19, 1984

Karmarkar at Bell Labs: an equation to find a new way through the maze

Folding the Perfect Corner

A young Bell scientist makes a major math breakthrough

Every day 1,200 American Airlines jets crisscross the U.S., Mexico, Canada and the Caribbean, stopping in 110 cities and bearing over 80,000 passengers. More than 4,000 pilots, copilots, flight personnel, maintenance workers and baggage carriers are shuffled among the flights; a total of 3.6 million gal. of high-octane fuel is burned. Nuts, bolts, altimeters, landing gears and the like must be checked at each destination. And while performing these scheduling gymnastics, the company must keep a close eye on costs, projected revenue and profits.

Like American Airlines, thousands of companies must routinely untangle the myriad variables that complicate the efficient distribution of their resources. Solving such monstrous problems requires the use of an abstruse branch of mathematics known as linear programming. It is the kind of math that has frustrated theoreticians for years, and even the fastest and most powerful computers have had great difficulty juggling the bits and pieces of data. Now Narendra Karmarkar, a 28-year-old Indian-born mathematician at Bell Laboratories in Murray Hill, N.J., after only a years' work has cracked the puzzle of linear programming by devising a new algorithm, a step-by-step mathematical formula. He has translated the procedure into a program that should allow computers to track a greater combination of tasks than ever before and in a fraction of the time.

Unlike most advances in theoretical mathematics, Karmarkar's work will have an immediate and major impact on the real world. "Breakthrough is one of the most abused words in science," says Ronald Graham, director of mathematical sciences at Bell Labs. "But this is one situation where it is truly appropriate."

Before the Karmarkar method, linear equations could be solved only in a cumbersome fashion, ironically known as the simplex method, devised by Mathematician George Dantzig in 1947. Problems are conceived of as giant geodesic domes with thousands of sides. Each corner of a facet on the dome

TIME MAGAZINE, December 3, 1984

Applications

- Capacity planning
- Network flow design
- Inventory control
- Facility layout
- Scheduling
- Healthcare
- Finance

Optimization applies everywhere

Linear Programming Problem

$$\text{maximize (or minimize) } f(x)$$
$$\text{such that } x \in S$$

Linear Program in Standard Form

$$
\begin{aligned}
\text{maximize } & c_1 x_1 + c_2 x_2 + \ldots + c_n x_n \\
\text{subject to } & a_{11} x_1 + a_{12} x_2 + \ldots + a_{1n} x_n \le b_1 \\
& a_{21} x_1 + a_{22} x_2 + \ldots + a_{2n} x_n \le b_2 \\
& \\
& a_{m1} x_1 + a_{m2} x_2 + \ldots + a_{mn} x_n \le b_m \\
& x_1, x_2, \ldots, x_n \ge 0
\end{aligned}
$$

Linear Propram in Geometry

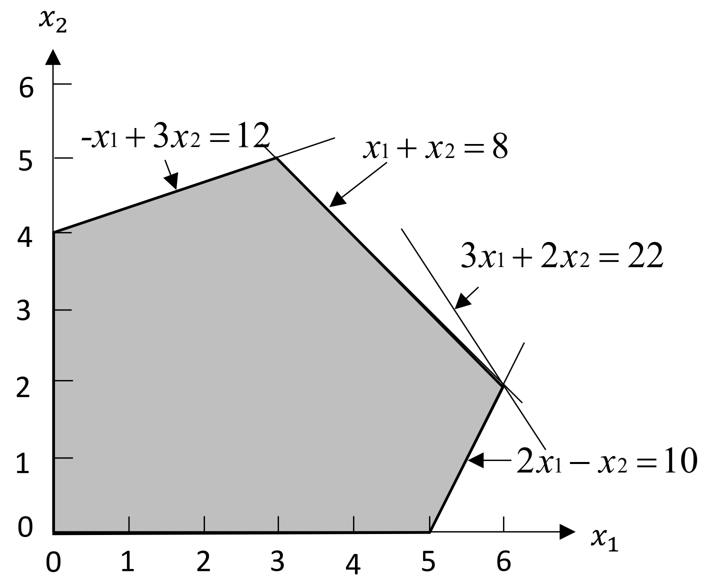
maximize $3x_1 + 2x_2$

subject to $-x_1 + 3x_2 \leq 12$

$x_1 + x_2 \leq 8$

$-2x_1 - 3x_2 \leq 12$

$x_1, x_2 \geq 0$



Types of Mathematical Programming

- Linear Programming (LP)
- Nonlinear Programming (NLP)
- Integer Programming (IP)
- Mixed Integer Programming (MIP)
- Quadratic Programming (QP)
- Quadratically Constrained Programming (QCP)
- Mixed-Integer Quadratic Programming (MIQP)
- Mixed-Integer Quadratically Constrained Programming (MIQCP)
- Etc.

Linear Programming and Integer Programming

- Most real-world problems are formulated as *integer programming* (IP) and *mixed-integer linear programming* problems (MILP is equivalent to *mixed integer programming,* MIP)
- One may think that IP or MIP problems wouldn't be harder than linear programming (LP) problems
- In fact, IPs are much harder than LPs. When solving IP problems, we cannot expect to obtain optimal integer solutions within a reasonable time frame

Example of Linear Programming Problem

- A steel company must decide how to allocate next week's time on a rolling mill, which is a machine that takes unfinished slabs of steel as input and can produce either of two semi-finished products: bands and coils. The mill's two products come off the rolling line at different rates:
    - Bands 200 tons/hr
    - Coils 140 tons/hr
- They also produce different profits:
    - Bands $ 25/ton
    - Coils $ 30/ton
- Based on currently booked orders, the following upper bounds are placed on the amount of each product to produce:
    - Bands 6000 tons
    - Coils 4000 tons
- Given that there are 40 hours of production time available this week, the problem is to decide how many tons of bands and how many tons of coils should be produced to yield the greatest profit. Formulate this problem as a linear programming problem. Solve this problem using of Excel Solver. Can you solve this problem by inspection?

Methods for Solving Linear Programming Problems

- Iterative approach:
  - Step 1. Start from an initial solution
  - Step 2. Check if the current solution is good enough
  - Step 3. If not, find a better solution and then return to Step 2
- Simplex methods (based on algebra)
- Inter point methods (based on calculus)

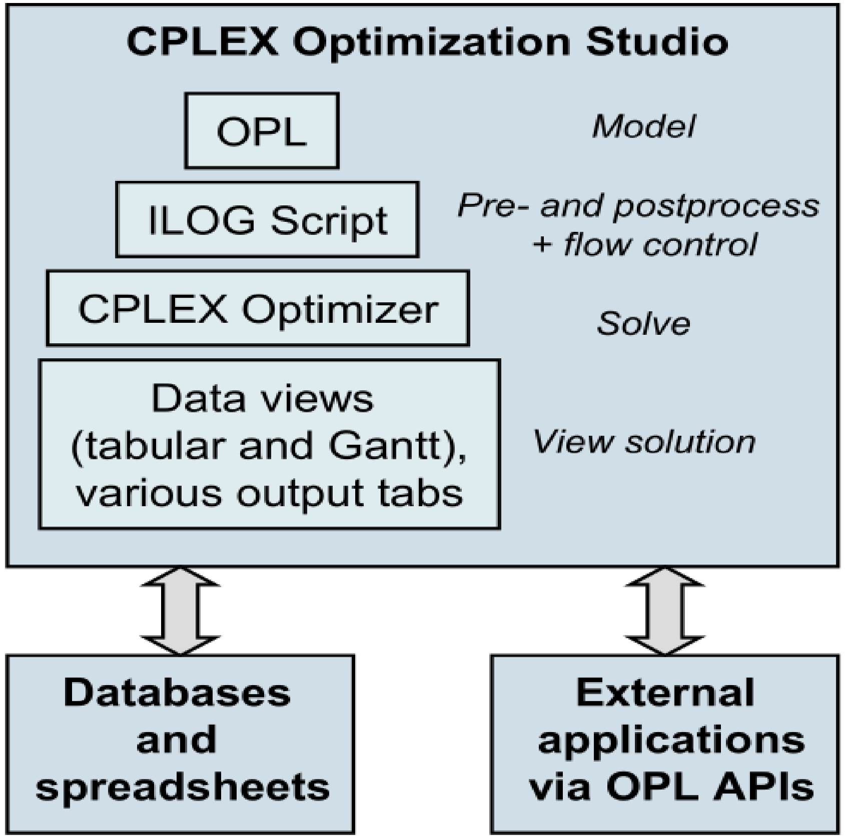
Software and Development Environment

- Solvers
  - IBM CPLEX
  - Gurobi Optimizer
  - GLPK
  - FICO Xpress
  - Lingo
- Modeling Languages
  - AMPL
  - GMPL

Modeling + Solving + Systems = Optimization

IBM CPLEX Optimization Studio

- An Integrated Development Environment (IDE) for modeling and solving mathematical programs
- Embedded an Optimization Programming Language (OPL) for modeling
- IBM ILOG Script for pre- and post processing, and flow control
- Solution approaches:
  - Several MP optimizers
  - CP Optimizers for CP problems
- Database and spreadsheet connectivity
- OPL APIs for integration with external applications



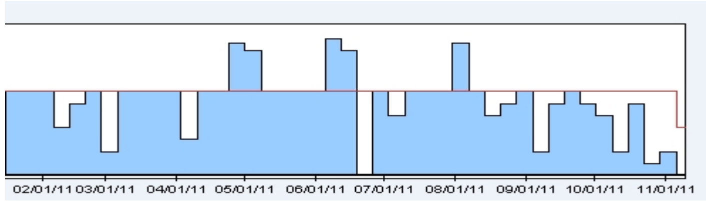
Source: IBM

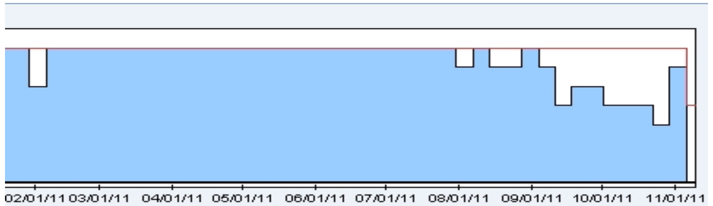
Other Approaches for Making Decisions

- Metaheuristics
- Reinforcement learning

Mathematical Programing versus Metaheuristics

- The MP enables *constrained and optimized solutions* that cannot be achieved merely through the metaheuristic algorithms
- MP provides insights over the problem via *relaxation* and *duality*
- MP is more flexible which can be adjusted easily to adapt to different problem instances
- Chromosome encoding can be a cumbersome task for evolutionary algorithms
- Solving MP problems may be challenge, but we may not solve for an exact solution every time





Reinforcement Learning

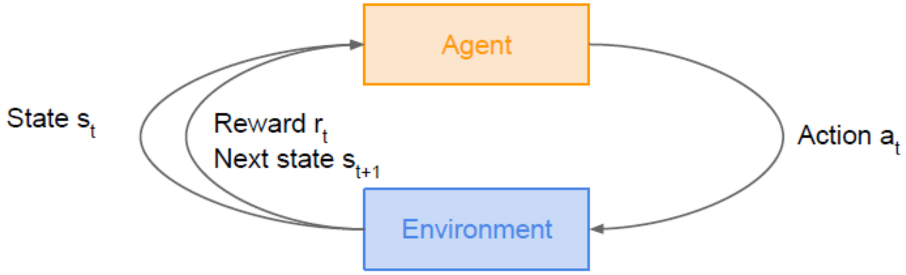
- The Q-value function is to measure how good is a state-action pair:

$$Q^{\pi}(s,a) = \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | s_0 = s, a_0 = a, \pi\right]$$

- The optimal Q-value function (aka, the Bellman equation) is :

$$Q^{*}(s,a) = \max_{\pi} \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r_t | s_0 = s, a_0 = a, \pi\right]$$

- The optimal policy π^* corresponds to taking the best action in any state as specified by Q*



CPLEX LP Format

```
Maximize obj: x1 + 2 x2 + 3 x3 + x4
Subject To
c1: - x1 + x2 + x3 + 10 x4 <= 20
c2: x1 - 3 x2 + x3 <= 30
c3: x2 - 3.5 x4 = 0
Bounds
0 <= x1 <= 40
2 <= x4 <= 3
General
x4
End
```

Solving the LP Problems

- LPs may be solved via:
  - GUI
  - Command lines
  - Programming languages

Solving the LP Problems

- GUI:

LINGO Model - Project_New_Int

```
max=(T_1/60)*x_1*0.99+(T_1/(50.5+61.6))*x_2*0.99*0.995+(T_1/(78.6+74.6))*0.99*0.995+(T_2/312)*x_4
_1+(T_2/364)*x_4_2+(T_2/389)*x_4_3;
(T_1/60)*x_1*0.99+(T_2/312)*x_4_1<=50000;
(T_1/(50.5+61.6))*x_2*0.99*0.995+(T_2/364)*x_4_2<=12500;
(T_1/(78.6+74.6))*x_3*0.99*0.995+(T_2/389)*x_4_3<=15000;
(T_2/312)*x_4_1=(T_1/60)*x_1*0.01;
(T_2/364)*x_4_2=(T_1/(50.5+61.6))*x_2*(1-0.99*0.995);
(T_2/389)*x_4_3=(T_1/(78.6+74.6))*x_3*(1-0.99*0.995);
y_1>=x_1;
2*y_2>=x_2;
2*y_3>=x_3;
y_4_1>=x_4_1;
y_4_2>=x_4_2;
y_4_3>=x_4_3;
y_1+y_2+y_3<=30;
y_4_1+y_4_2+y_4_3<=4;
x_1+x_2+x_3+x_4_1+x_4_2+x_4_3<=22;
T_1=226800;
T_2=208800;
@gin(x_1);
@gin(x_2);
@gin(x_3);
@gin(y_1);
@gin(y_2);
@gin(y_3);
```

Solution Report - Project_New_Int

```
Global optimal solution found at step:            7
Objective value:                           62759.58
Branch count:                                     1

                Variable           Value        Reduced Cost
                     T_1        226800.0           0.0000000
                     X_1        13.00000           -3780.000
                     X_2        6.000000           -2023.194
                     T_2        208800.0           0.0000000
                   X_4_1       0.7342759           0.0000000
                   X_4_2       0.3163740           0.0000000
                   X_4_3       0.4123297E-01       0.0000000
                     X_3        1.000000           -22.13225
                     Y_1        13.00000           0.0000000
                     Y_2        3.000000           0.0000000
                     Y_3        1.000000           0.0000000
                   Y_4_1       0.7342759           0.0000000
                   Y_4_2       0.3163740           0.0000000
                   Y_4_3       0.4123297E-01       0.0000000

                     Row    Slack or Surplus      Dual Price
                       1        62759.58            1.000000
                       2        860.0000           0.0000000
                       3        360.8385           0.0000000
                       4        13519.58           0.0000000
                       5       0.0000000            1.000000
                       6       0.0000000            1.000000
```

Solving the LP Problems

- Command lines:

C:\Windows\system32\cmd.exe

```
C:\Super Size>glpsol --cpxlp supersize.lp.txt
lpx_read_cpxlp: reading problem data from `supersize.lp.txt'...
lpx_read_cpxlp: 899 rows, 1653 columns, 4721 non-zeros
lpx_read_cpxlp: 1653 integer columns, 812 of which are binary
lpx_read_cpxlp: 2557 lines were read
lpx_simplex: original LP has 899 rows, 1653 columns, 4721 non-zeros
lpx_simplex: presolved LP has 690 rows, 1444 columns, 3340 non-zeros
lpx_adv_basis: size of triangular part = 690
*     0:   objval =  0.000000000e+000   infeas =  0.000000000e+000 (0)
*   200:   objval =  0.000000000e+000   infeas =  0.000000000e+000 (0)
*   400:   objval =  0.000000000e+000   infeas =  0.000000000e+000 (0)
*   600:   objval =  4.000000000e+000   infeas =  0.000000000e+000 (0)
*   693:   objval =  4.316666667e+000   infeas =  0.000000000e+000 (0)
OPTIMAL SOLUTION FOUND
Integer optimization begins...
Objective function is integral
+   693: mip =     not found yet <=              +inf        (1; 0)
+   732: mip =  4.000000000e+000 <=  4.000000000e+000   0.0% (6; 0)
+   732: mip =  4.000000000e+000 <=     tree is empty   0.0% (0; 11)
INTEGER OPTIMAL SOLUTION FOUND
Time used:   0.0 secs
Memory used: 1.8M (1869924 bytes)

C:\Super Size>
```

Solving the LP Problems

- Programming languages:

```
// Create decision variables for the foods to buy
buy = model.addVars(0, 0, cost, 0, Foods, nFoods);

// The objective is to minimize the costs
model.set(GRB_IntAttr_ModelSense, 1);

// Update model to integrate new variables
model.update();

// Nutrition constraints
for (int i = 0; i < nCategories; ++i)
{
  GRBLinExpr ntot = 0;
  for (int j = 0; j < nFoods; ++j)
  {
    ntot += nutritionValues[j][i] * buy[j];
  }
  model.addConstr(ntot == nutrition[i], Categories[i]);
}

// Solve
model.optimize();
printSolution(model, nCategories, nFoods, buy, nutrition);
```

Challenges for Solving Linear Programming

- Complexity of the real-world problems
- Explosion of the Big Data
- Algorithm implemented in “black box”
- Runtime
- Optimal or sub-optimal solution? How good is the solution?

Future Operations Research

- More platforms to choose:
  - Mobile
  - Cloud
- More efficient solvers:
  - Presolve techniques (reformulation, probing, and etc.)
  - Decompositions
  - Advanced search algorithms
  - Distributed and parallel computing

Next Week

- Read chapter 1
- Homework 1 (due at the beginning of the class next week)
  - Exercises 1.1, 1.2 and 1.3
- Install MS Visual Studio (recommended version 2010) and IBM CPLEX on your laptop
  - Visual studio is available at the NCTU *Information Technology Service Center* website
  - Register CPLEX Academic Initiative

Remember to bring your PC to the class on next week!