ORF 522

Linear Programming and Convex Analysis

Introduction

Marco Cuturi

Practical

- My office is #115, down the corridor.
- Office hours: Thursday 9AM:11AM.
- Come anytime though, in the worst case I'll tell you I am busy then.
- Check the blackboard website, I will add stuff regularly.
- TA: Jamol Pender.

Today

- A few words about *mathematical* programming and *linear* programming.
- A few examples:
  - Diet problem,
  - Network Flow.
- Review of the course.

Mathematical programming,
a bit of terminology

Mathematical & Linear Programming

- The term *programming* in *mathematical programming* is actually **not** related to computer programs.
- Dantzig explains the jargon in 2002 (document available on BB)
  - `The military refer to their various plans or proposed schedules of training, logistical supply and deployment of combat units as a program. When I first analyzed the Air Force planning problem and saw that it could be formulated as a system of linear inequalities, I called my paper Programming in a Linear Structure. Note that the term program was used for linear programs long before it was used as the set of instructions used by a computer. In the early days, these instructions were called codes.`

Mathematical & Linear Programming

- In the summer of 1948, Koopmans and I visited the Rand Corporation. One day we took a stroll along the Santa Monica beach. Koopmans said: Why not shorten Programming in a Linear Structure to Linear Programming? I replied: Thats it! From now on that will be its name. Later that day I gave a talk at Rand, entitled Linear Programming; years later Tucker shortened it to Linear Program.

- The term Mathematical Programming is due to Robert Dorfman of Harvard, who felt as early as 1949 that the term Linear Programming was too restrictive.

Mathematical & Linear Programming

- Today mathematical programming is synonymous with **optimization**. A relatively **new discipline** and one that has had significant impact.
  - `What seems to characterize the pre-1947 era was lack of any interest in trying to optimize. T. Motzkin in his scholarly thesis written in 1936 cites only 42 papers on linear inequality systems, none of which mentioned an objective function.`
- Partly a local product: optimization theory starts here with the work of Von Neumann, Khun and Tucker, *etc.*

Origins & Success

- **Monge**'s 1781 memoir is the earliest known anticipation of Linear Programming type of problems, in particular of the transportation problem (moving piles of dirt into holes).
- In the early 40's significant work can also be attributed to **Kantorovich** in the USSR (Nobel 75) on transport planning as well. More dramatic application: *road of life* from & to Leningrad during WW2.
- **Dantzig** proposed a general method to solve LP's in 1947, the **simplex method**, which ranks among `the top 10 algorithms with the greatest influence on the development and practice of science and engineering in the 20th century` according to the journal *Computing in Science & Engineering*.
- You can check the other ones in BB. All of them are after WW2 and the simplex is the second oldest.
- Other laureates: metropolis, FFT, quicksort, Krylov subspaces, QR decomposition *etc.*

Mathematical Programming

- A general formulation for a mathematical programming problem is that of defining the unknown variables $x_1, x_2, \cdots, x_n \in \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_n$ such that

$$\begin{array}{ll} \text{minimize (or mazimize)} & \boldsymbol{f(x_1, x_2, \cdots, x_n)}, \\ \text{subject to} & \boldsymbol{g_i(x_1, x_2, \cdots, x_n)} \left\{ \begin{array}{c} <,> \\ = \\ \leq,\geq \end{array} \right\} \boldsymbol{b_i}, i = 1, 2, \cdots, m; \end{array}$$

  where the b_i's are real constants and the functions $\boldsymbol{f}$ (the objective) and $\boldsymbol{g_1, g_2, \cdots, g_m}$ (the constraints) are real-valued functions of $\mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_n$.

- the sets $\mathcal{X}_i$ need not be the same, as $\mathcal{X}_i$ might be
  - **R** scalar numbers,
  - **Z** integers,
  - $\mathbf{S}_n^+$ positive definite matrices,
  - strings of letters,
  - *etc.*

- When the $\mathcal{X}_i$ are different, the adjective *mixed* usually comes in.

The Focus in this course: Linear Programs in $\mathbf{R}^n$

- the general form of linear programs in $\mathbf{R}^n$:

$$
\begin{array}{ll}
\text{max } \textit{or} \text{ min} & z = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n, \\
\text{subject to} & \left\{ \begin{array}{ccccccccc}
a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & \left\{ \begin{array}{c} <,> \\ = \\ \leq,\geq \end{array} \right\} & b_1, \\
\vdots & & & & & & & & \vdots \\
a_{m1}x_1 & + & a_{m2}x_2 & + & \cdots & + & a_{mn}x_n & \left\{ \begin{array}{c} <,> \\ = \\ \leq,\geq \end{array} \right\} & b_m,
\end{array} \right. \\
\text{where} & x_1 \geq 0,\ x_2 \geq 0,\ \cdots,\ x_n \geq 0.
\end{array}
$$

- linear objective, linear constraints... simple.
- yet powerful for many problems and one of the first classes of mathematical programs that was solved.

Linear Programs, landmarks in history

- First solution by Dantzig in the late 40's, the simplex.
- At the time, programs were solved by hand, the algorithm reflects this.
- In 1972, Klee and Minty show that the simplex has an exponential worst case complexity.
- Low complexity of linear programming proved (in theory) by Nemirovski, Yudin and Khachiyan in the USSR in 1976.
- First efficient algorithm with provably low complexity discovered by Karmarkar at Bell Labs in 1984.

Mathematical Programming Subfields

- **convex** programming: f is a **convex** function and the constraints g_i, if any, form a **convex** set. see A. d'Aspremont class in the second semester.
  - **Linear programming**.
  - Second order cone programming (SOCP).
  - Semidefinite Programming, that is linear programs in $\mathbf{S}_n^+$.
  - Conic programming, with more general cones.
- Quadratic programming (QP), with quadratic objectives and linear constraints,
- Nonlinear programming,
- Stochastic programming,
- Combinatorial programming: discrete set of feasible solutions. **integer programming**, that is LP's with integer variables, is a subfield.

Recurrent topics of the course

- The set of candidates is convex $\Rightarrow$ will need **convex analysis** to describe it.
- Everything is linear $\Rightarrow$ constant use of **linear algebra**.
- The **simplex algorithm** itself is a series of **elementary algebraic manipulations**, not much calculus.
- **LP is a powerful tool**. As is the case in statistics, **linearity can be cast in different models** $\Rightarrow$ cookbook of applications in stats, ML, finance, networks, compressed sensing.
- Assess the tool's efficiency $\Rightarrow$ study its **complexity**, worst case scenarios and *average* ones.
- LP is a first **introduction to convex optimization**, so we will dwell on more advanced **convexity topics** later in the course.

Some examples

The Diet Problem

- Most introductions to LP start with the diet problem.
- The reason: historically, one of the first large scale LP's that was computed. More on this later.
- You're a (bad) cook obsessed with numbers trying to come up with a new **cheap** dish that **meets nutrition standards**.
- You summarize your problem in the following way:

Ingredient	Carrot	Cabbage	Cucumber	Required per dish
Vitamin A [mg/kg]	**35**	**0.5**	**0.5**	**0.5mg**
Vitamin C [mg/kg]	**60**	**300**	**10**	**15mg**
Dietary Fiber [g/kg]	**30**	**20**	**10**	**4g**
Price [$/kg]	**0.75**	**0.5**	**0.15**	-

The Diet Problem

- Let x_1, x_2 and x_3 be the amount in kilos of carrot, cabbage and cucumber in the new dish.
- Mathematically,

$$\begin{array}{lrl} \text{minimize} & \mathbf{0.75}\boldsymbol{x_1} + \mathbf{0.5}\boldsymbol{x_2} + \mathbf{0.15}\boldsymbol{x_3}, & \textbf{cheap,} \\ \text{subject to} & \mathbf{35}\boldsymbol{x_1} + \mathbf{0.5}\boldsymbol{x_2}\mathbf{0.5}\boldsymbol{x_3} \geq \mathbf{0.5}, & \boldsymbol{nutritious,} \\ & \mathbf{60}\boldsymbol{x_1} + \mathbf{300}\boldsymbol{x_2} + \mathbf{10}\boldsymbol{x_3} \geq \mathbf{15}, & \\ & \mathbf{30}\boldsymbol{x_1} + \mathbf{20}\boldsymbol{x_2} + \mathbf{10}\boldsymbol{x_3} \geq \mathbf{4}, & \\ & \boldsymbol{x_1, x_2, x_3} \geq \mathbf{0}. & \textbf{reality.} \end{array}$$

- The program can be solved by standard methods. The optimal solution yields a price of 0.07$ pre dish, with 9.5g of carrot, 38g of cabbage and 290g of cucumber...

The Diet Problem

- The first large scale experiment for the simplex algorithm: 77 variables (ingredients) and 9 constraints (health guidelines)
- The solution, computed by hand-operated desk calculators took 120 man-days.
- The first recommendation was to drink several *liters* of vinegar every day.
- When vinegar was removed, Dantzig obtained *200 bouillon cubes* as the basis of the diet.
- This illustrates that a clever and careful mathematical **modeling** is always important before **solving** anything.

Flow of packets in Networks

We follow with an example in networks:

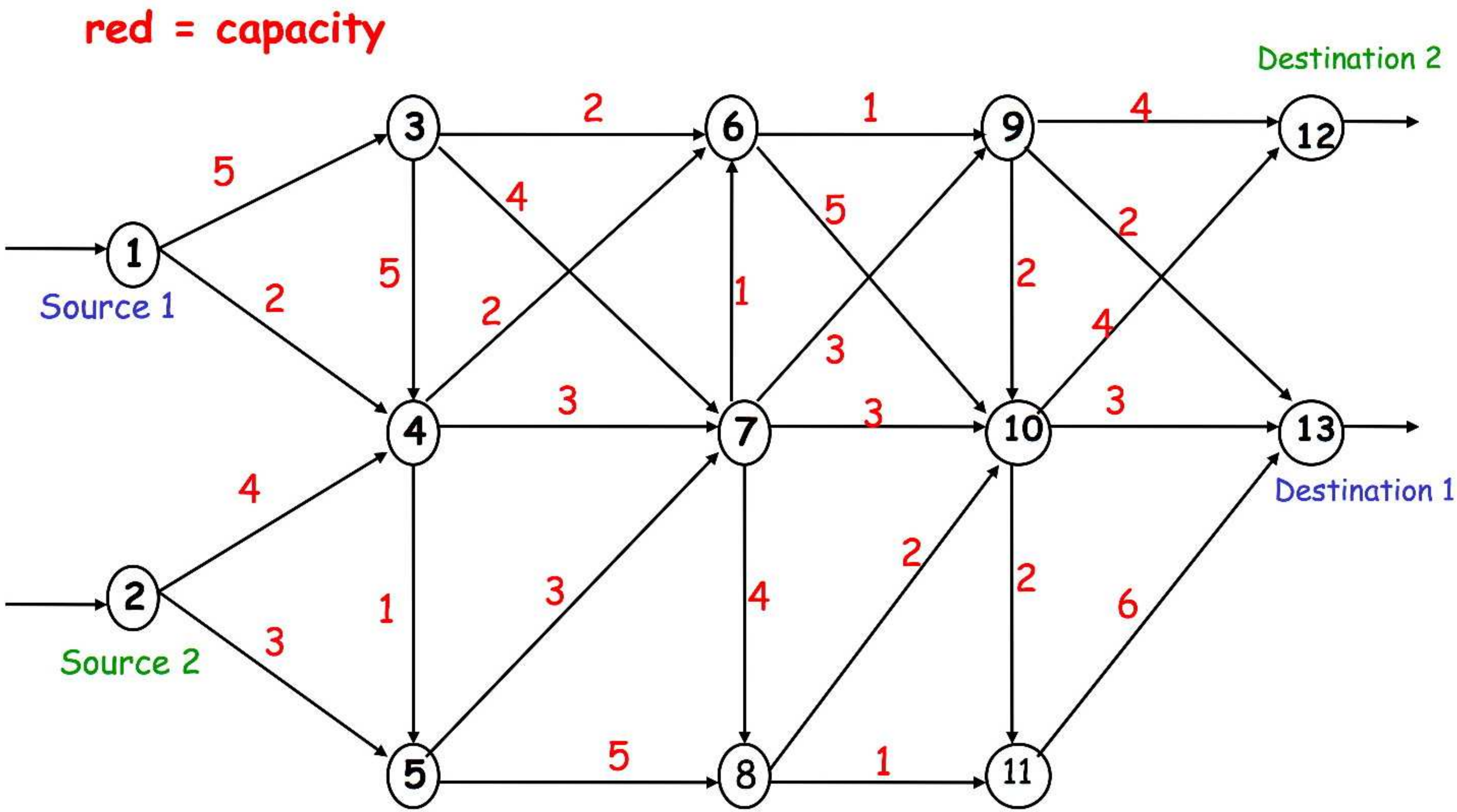
- We use the internet here, but this could be any network (factory floor, transportation, etc).
- Transport data packets from a source to a destination.
- For simplicity: two sources, two destinations.
- Each link in the network has a fixed capacity (bandwidth), shared by all the packets in the network.

Networks: Routing

- When a link is saturated (congestion), packets are simply dropped.
- Packets are dropped at random from those coming through the link.
- Objective: choose a routing algorithm to maximize the **total bandwidth** of the network.

This randomization is not a simplification. TCP/IP, the protocol behind the internet, works according to similar principles.. . .

Networks: Routing



Networks: Routing

A model for the network routing problem: let $N = \{1, 2, \ldots, 13\}$ be the set of network nodes and $L = \{(1,3), \ldots, (11,13)\}$ the set of links.

Variables:

- x_{ij} the flow of packets with origin 1 and destination 1, going through the link between nodes i and j.
- y_{ij} the flow of packets with origin 2 and destination 2, going through the link between nodes i and j.

Parameters:

- u_{ij} the maximum capacity of the link between nodes i and j.

Networks: Routing

In EXCEL. . .

Routing problem: Modeling

Write this as an optimization problem.

Consistency constraints:

- Flow coming out of a node must be less than incoming flow:

$$\sum_{j:\ (i,j)\in L} x_{ij} \leq \sum_{j:\ (j,i)\in L} x_{ij}, \quad \text{for all nodes } i$$

  and

$$\sum_{j:\ (i,j)\in L} y_{ij} \leq \sum_{j:\ (j,i)\in L} y_{ij}, \quad \text{for all nodes } i$$

- Flow has to be positive:

$$x_{ij}, y_{ij} \geq 0, \quad \text{for all } (i,j) \in L$$

Routing problem: Modeling

Capacity constraints:

- Total flow through a link must be less than capacity:

$$x_{ij} + y_{ij} \le u_{ij}, \quad \text{for all } (i,j) \in L$$

- No packets originate from wrong source:

$$x_{2,4},\ x_{2,5},\ y_{1,3},\ y_{1,4} = 0$$

Objective:

- Maximize total throughput at destinations:

$$\text{maximize } x_{9,13} + x_{10,13} + x_{11,13} + y_{9,12} + y_{10,12}$$

Routing problem: Modelling

The final program is written:

$$
\begin{array}{ll}
\text{maximize} & x_{9,13} + x_{10,13} + x_{11,13} + y_{9,12} + y_{10,12} \\
\text{subject to} & \displaystyle\sum_{j:\ (i,j)\in L} x_{ij} \leq \sum_{j:\ (j,i)\in L} x_{ij} \\
& \displaystyle\sum_{j:\ (i,j)\in L} y_{ij} \leq \sum_{j:\ (j,i)\in L} y_{ij} \\
& x_{ij} + y_{ij} \leq u_{ij} \\
& x_{2,4}, x_{2,5}, y_{1,3}, y_{1,4} = 0 \\
& x_{ij}, y_{ij} \geq 0, \quad \text{for all } (i,j) \in L
\end{array}
$$

Constraints and objective are linear: this is a **linear program**.

Routing problem: Solving

- In this case, the model was written entirely in EXCEL
- EXCEL has a rudimentary linear programming solver (which does not work very well for macs unfortunately)
- This is how the optimal solution was found here. In general, specialized solvers are used (more later).

- Original solution, : network capacity of 3.7
- Optimal capacity: **14 !!**

Outline of the Course

Syllabus

- 4 lectures of **basic convex analysis** in the LP context.
- 4 lectures on the **simplex**: the theory, practical implementations, issues (cycling, degeneracy) and efficiency.
- 4 lectures on **duality** and **other algorithms** (IPM, Ellipsoid, Integer programming)
- 4 lectures on **applications** of LP's, notably network flows & transportation, finance, machine learning, compressed sensing.
- 4 lectures on the study of **canonical polytopes & cones**, including the cone of PSD matrices.
- 4 lectures on **combinatorial** aspects of convex analysis.

Ressources and Material

- All lectures will be online. All important proofs will be in the slides.
- **Linear Programming, Foundations and Extensions** by Robert Vanderbei, Springer.
- We will use some of the tools proposed online by Prof. Vanderbei for the simplex.
- As a general rule, you will find that a lot of material is online.
- Online `http://www.stanford.edu/~boyd/cvxbook/`,and quite cheap as a book, **Convex Optimization** by Stephen Boyd and Lieven Vandenberghe, Cambridge Univ.Press, 2004.
- For convexity, **A course in convexity** by A. Barvinok, AMS GSM vol.54, 2002.

Grading & Various

- 4 homework assignments (roughly every 4-5 lectures), total 50%
- final exam, total 50%.

- Won't be here (NIPS conference) on Tuesday 8th and Thursday 10th of December.
- Thanksgiving 26th of November.

Any questions?