Optimization Methods in Finance

PART 1
WELCOME

Welcome!

- My name is Friedrich Eisenbrand
- Assistants of the course: Thomas Rothvoß, Nicolai Hähnle

How to contact me:

- Come to see me during office ours: Tuesdays 11-12
- Send me an e-mail `friedrich.eisenbrand@epfl.ch` at least one day ahead, announcing your visit
- Apart from personal contact: I cannot guarantee to answer all your e-mails.
- **Highly appreciated:** E-mails about errors/typo's on my slides

Course webpage

See `http://disopt.epfl.ch` and follow the Teaching link!

Syllabus

Topics

- Linear Programming: Simplex Method, Computing a dedicated bond portfolio, asset pricing
- Quadratic Programming: Portfolio Optimization (Markowitz model)
- Integer Programming: Constructing an index fund, Combinatorial Auctions
- Dynamic Programming: Option Pricing, Structuring asset backed securities
- Stochastic Programming: Asset/Liability management
- Convex Programming: Market Equilibria Problems

Prerequisites and literature

Prerequisites

- Linear Algebra
- Basic programming skills: C++, Matlab
- Basic knowledge in Algorithms: O-Notation, Pseudocode, …
- Basic probability theory

Literature

- G. Cornuéjols and R. Tütüncü, Optimization Methods in Finance, Cambridge (main reference)
- D. Luenberger, Investment Science, Oxford University Press (Reference on basic mathematics and terminology in finance)

Main skills

- Learning of basic optimization methods and how to apply them in the world of finance: Modeling, practical problem solving (case studies) using software packages
- Mathematical rigor: Proofs, analysis of running time, correctness of methods

Organization

Exercises

- Every two weeks there is an assignment sheet
- For the i-th exercise sheet the students are supposed to solve the assignments in the tutorial in week $2i + 1$. The solutions will be discussed in week 2i+2.
- First assignment is available in the 2nd week

Organization

Grading

- A total amount of 100 points can be reached as follows
-
  - 30 points from the midterm
  - 10 points from the presentation of a theoretical exercise
  - 20 points for practical exercises
  - 40 points from the final exam
- The 10 points can already be gained by presenting a single theoretical exercise
- 60 points suffice to pass the course
- This is a 4-credit course

PART 2
LINEAR PROGRAMMING

Primary objectives:

- Linear programming: Example, Definition
- The standard form
- Basic feasible solutions
- simplex algorithm
- Along the way: Develop your own toy-solver in C++
- Use of modeling languages and efficient LP-solvers
- Constructing dedicated bond-portfolio

Example: Modelling Cashflow Management

Our company has net cash flow requirements (in 1000 CHF)

Month	Jan	Feb	Mar	April	May	June
Net cash flow	-150	-100	200	-200	50	300

For example in January we have to pay CHF 150k, while in March we will get CHF 300k. Initially we have 0 CHF, but the following possibilities to borrow/invest money

1. We have a credit line of CHF 100k per month. For a credit we have to pay 1% interest per month.
2. In the first 3 months we may issue a 90-days commercial paper (to borrow money) at a total interest of 2% (for the whole 3 months).
3. Excess money can be invested at an interest rate of 0.3% per month.

Task: We want to maximize our wealth in June, while we have to fulfill all payments. How shall we invest/borrow money?

Setting up a Linear Program

Introduce decision variables

$$
\begin{aligned}
x_i &= \text{credit in month } i \in \{1,\dots,6\} \\
y_i &= \text{amount of money due to commercial paper } i \in \{1,2,3\} \\
z_i &= \text{fund excess in month } i \in \{1,\dots,6\}
\end{aligned}
$$

Define $b = (-150, -100, 200, -200, 50, 300)$ as the vector of net cash flow. Then the following LP determines the optimal investment strategy:

$$
\begin{aligned}
\max z_6 - x_6 & & & \\
x_1 + y_1 - z_1 &= -b_1 & & \\
x_i + y_i - 1.01x_{i-1} + 1.003z_{i-1} - z_i &= -b_i & \forall i &= 2,3 \\
x_i - 1.02y_{i-3} - 1.01x_{i-1} + 1.003z_{i-1} - z_i &= -b_i & \forall i &= 4,\dots,6 \\
0 \leqslant x_i &\leqslant 100 & \forall i &= 1,\dots,6 \\
z_i &\geqslant 0 & \forall i &= 1,\dots,6 \\
y_i &\geqslant 0 & \forall i &= 1,2,3
\end{aligned}
$$

Notation

Let $A \in \mathbb{R}^{m\times n}$, $v \in \mathbb{R}^n$ and $J = \{j_1, \dots, j_k\} \subseteq \{1, .., n\}$ with $j_1 \leqslant j_2 \leqslant \dots \leqslant j_k$.

- For $i \in \{1, \dots, m\}$ and $j \in \{1, \dots, n\}$:
  - a^j: j-th column of A
  - a_i: i-th row of A
  - $A(i, j)$: Element of A in i-th row and j-th column
  - $v(i)$ (also v_i): i-th component of v
- A_J: Matrix $\left(a^{j_1}, \dots, a^{j_k}\right)$
- v_J: Vector $(v(j_1), \dots, v(j_k))^T$

Example

- $A = \left(\begin{smallmatrix} 2 & 3 & 4 & 5 \\ 1 & 3 & 2 & 7 \end{smallmatrix}\right)$, $v^T = \left(1\ 2\ 3\ 4\right)$, $J = \{2, 4\}$
- $a^3 = \left(\begin{smallmatrix} 4 \\ 2 \end{smallmatrix}\right)$
- $A_J = \left(\begin{smallmatrix} 3 & 5 \\ 3 & 7 \end{smallmatrix}\right)$, $v_J = \left(\begin{smallmatrix} 2 \\ 4 \end{smallmatrix}\right)$

Definition: Linear Program

Let $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, $I_\geqslant, I_\leqslant, I_= \subseteq \{1, \dots, m\}$ and $J_\geqslant, J_\leqslant \subseteq \{1, \dots, n\}$
Linear Program (LP) consists of:

- Objective function:

$$\max c^T x$$
$$\text{or } \min c^T x$$

- Constraints

$$a_i^T x \geqslant b(i), i \in I_\geqslant$$
$$a_j^T x \leqslant b(j), j \in I_\leqslant$$
$$a_k^T x = b(k), k \in I_=$$

- Bounds on Variables

$$x_{J_\geqslant} \geqslant 0$$
$$x_{J_\leqslant} \leqslant 0$$

Definitions

feasible LP, bounded LP

- $x^* \in \mathbb{R}^n$ is feasible of feasible solution of LP, if x^* satisfies all constraints and bounds on variables
- LP is feasible if there exist feasible solutions of LP. Otherwise LP is infeasible
- LP is bounded, if LP is feasible and there exits constant $M \in \mathbb{R}$ with $c^T x < M$ for all feasible solutions $x \in \mathbb{R}^n$ if LP is a maximization problem or $c^T x > M$ for all feasible $x \in \mathbb{R}^n$ if LP is minimization problem

Transformations

1) $\max c^T x$ can be replaced by $\min -c^T x$
2) $a_i^T x \leqslant b(i)$ can be replaced by $-a_i^T x \geqslant -b(i)$
3) Bound $x(j) \geqslant 0$ or $x(j) \leqslant 0$ can be replaced by constraint $e_j^T x \geqslant 0$ or $e_j^T x \leqslant 0$ respectively, where e_j is j-th unit vector
4) $a_i^T x \leqslant b(i)$ can be replaced by $a_i^T x + s = b(i)$, where s is a new variable which is bounded by $s \geqslant 0$
5) Conversion of an unbounded variable $x(j)$ in a bounded variable: Introduce new bounded variables $p_+(j), p_-(j) \geqslant 0$ and replace each occurrence of $x(j)$ by $p_+(j) - p_-(j)$
6) Conversion of a variable $x(j)$ bounded by $\leqslant 0$ into a variable which is bounded by $\geqslant 0$: Replace bound $x(j) \leqslant 0$ by $x(j) \geqslant 0$ and replace each other occurrence of $x(j)$ by $-x(j)$ (also in objective function!)

Standard forms

- Inequality standard form :

$$\begin{aligned} \max c^T x \\ Ax \leqslant b \end{aligned}$$

- Equality standard form or simply standard form :

$$\begin{aligned} \min c^T x \\ Ax = b \\ x \geqslant 0 \end{aligned}$$

Example

$$\begin{aligned} \max 2 \cdot x(1) - 4 \cdot x(2) \\ 3 \cdot x(1) + 2 \cdot x(2) \geqslant 4 \\ 2 \cdot x(1) + 4 \cdot x(2) \geqslant 9 \\ x(2) \leqslant 0 \end{aligned}$$

is to be transformed into standard form.
Application of 1) and 2)

$$\begin{aligned} \min -2 \cdot x(1) + 4 \cdot x(2) \\ -3 \cdot x(1) - 2 \cdot x(2) \leqslant -4 \\ -2 \cdot x(1) - 4 \cdot x(2) \leqslant -9 \\ x(2) \leqslant 0 \end{aligned}$$

Application of 6)

$$\min -2 \cdot x(1) - 4 \cdot x(2)$$
$$-3 \cdot x(1) + 2 \cdot x(2) \leqslant -4$$
$$-2 \cdot x(1) + 4 \cdot x(2) \leqslant -9$$
$$x(2) \geqslant 0$$

Application of 5)

$$\min -2 \cdot p_+(1) + 2 \cdot p_-(1) - 4 \cdot x(2)$$
$$-3 \cdot p_+(1) + 3 \cdot p_-(1) + 2 \cdot x(2) \leqslant -4$$
$$-2 \cdot p_+(1) + 2 \cdot p_-(1) + 4 \cdot x(2) \leqslant -9$$
$$p_+(1), p_-(1), x(2) \geqslant 0$$

Application of 4)

$$\min -2 \cdot p_+(1) + 2 \cdot p_-(1) - 4 \cdot x(2) + 0 \cdot s(1) + 0 \cdot s(2)$$
$$-3 \cdot p_+(1) + 3 \cdot p_-(1) + 2 \cdot x(2) + s(1) = -4$$
$$-2 \cdot p_+(1) + 2 \cdot p_-(1) + 4 \cdot x(2) + s(2) = -9$$
$$p_+(1), p_-(1), x(2), s(1), s(2) \geqslant 0$$

Convention

In the following, we will always assume LPs to be in standard form

$$\begin{aligned} \min c^T x \\ Ax = b \\ x \geqslant 0 \end{aligned} \tag{1}$$

Reminder

- Vectors $v_1, \ldots, v_k \in \mathbb{R}^n$ are linearly independent, if one has $\sum_{i=1}^{k} x(i)\, v_i \neq 0$ for all $x \neq 0$ $\mathbb{R}^k$
- Column-rank of A is maximal number of linearly independent columns of A
- Row-rank of A is maximal number of linearly independent rows of A
- Rank of A, $\text{rank}(A)$: Maximal number of linearly independent columns of rows of A
- A has full row-rank (column-rank), if $\text{rank}(A) = m$ ($\text{rank}(A) = n$)

W.l.o.g. A has full row-rank

- Suppose $\text{rank}(A) < m$ and let the first row be in the span of the other rows

$$a_1 = \sum_{i=2}^{m} \lambda_i a_i \text{ with suitable numbers } \lambda_2, \dots, \lambda_m \in \mathbb{R}.$$

- If $\sum_{i=2}^{m} \lambda_i b(i) = b(1)$, then one has for all $x \in \mathbb{R}^n$ with $a_i^T x = b(i), i = 2, \dots, m$ also $a_1^T x = b(1)$ which means that the first equation in $Ax = b$ can be discarded.
- If $\sum_{i=2}^{m} \lambda_i b(i) \neq b(1)$, then there does not exist an $x \in \mathbb{R}^n$ with $Ax = b$ and the LP (1) is infeasible.

Convention

In the following we will assume that A has full row-rank.

Lemma 2.1

If LP (1) has an optimal solution, then there exists an optimal solution $x^* \in \mathbb{R}^n$ of (1) such that A_J has full column rank, where $J = \{j \mid x^*(j) > 0\}$.

Intuition for proof of Lemma 2.1

Consider LP

$$\min x_1 + x_2$$
$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix} x = \begin{pmatrix} 6 \\ 15 \end{pmatrix}$$
$$x \geqslant 0$$

$x = (1,1,1)^T$ is feasible but columns of A corresponding to positive entries of x (matrix A itself) are not linearly independent.
Vector $d = (1,-2,1)^T$ satisfies $Ad = 0$.
$A(x + 1/2d) = Ax = b$, $x + 1/2d = (3/2, 0, 3/2) \Rightarrow x + 1/2d$ is feasible, with fewer nonzero entries and smaller objective value than x.

Proof of Lemma 2.1

- Let x^* be an optimal solution and suppose that columns of A_J are linearly dependent
- Idea: Compute new optimal solution $\widetilde{x}$ with $J' = \{j \mid \widetilde{x}(j) > 0\} \subset J$
- After finite number of repetitions of this step, one has optimal solution which satisfies the condition of the theorem
- Let $\overline{J} = \{1, \dots, n\} \setminus J$
- A_J does not have full column rank $\Longrightarrow \exists\, d \in \mathbb{R}^n$ with $d \neq 0$, $Ad = 0$ and $d(j) = 0\ \forall j \in \overline{J}$
- $\Longrightarrow x^* \pm \varepsilon d$ is feasible for $\varepsilon > 0$ sufficiently small.

Proof cont.

- $c^T(x^* \pm \varepsilon d) = c^T x^* \pm \varepsilon c^T d$; since x^* is optimal it follows that $c^T d = 0$
- Via eventually changing d to $-d$ we can assume that there exists a $j \in J$ with $d(j) < 0$
- Consider $x^* + \varepsilon d$. Goal: Choose $\varepsilon > 0$ in such a way that $x^* + \varepsilon d$ still feasible ($\iff x^* + \varepsilon d \geqslant 0$) but also $(x^* + \varepsilon d)(j) = 0$ for some $j \in J$
- How large can we choose $\varepsilon > 0$ without getting infeasible?
- Infeasibility can be caused by indices j with $d(j) < 0$; Let $K \subseteq J$ be the set of these indices
- We need for all $k \in K$ the condition:

$$\begin{aligned} & x^*(k) + \varepsilon\, d(k) \geqslant 0 \\ \iff \quad & \varepsilon \leqslant -x^*(k)/d(k) \end{aligned}$$

Proof cont.

- Let $k' \in K$ be an index with

$$-x^*(k')/d(k') = \min_{k \in K} -x^*(k)/d(k)$$

  and let $\varepsilon' = -x^*(k')/d(k')$
- Then $x^* + \varepsilon' d$ is feasible and $J' = \{j \mid (x^* + \varepsilon' d)(j) > 0\} \subset J$. □

Definitions

Basic solution, basic feasible solution, associated basis

- $x^* \in \mathbb{R}^n$ is called a basic solution, if $Ax^* = b$ and $\text{rank}(A_J) = |J|$, where $J = \{j \mid x^*(j) \neq 0\}$; A basic solution is a feasible basic solution, if $x^* \geqslant 0$
- Basis is index set $B \subseteq \{1, \ldots, n\}$ with $\text{rank}(A_B) = m$ and $|B| = m$
- $x^* \in \mathbb{R}^n$ with $A_B x_B^* = b$ and $x^*(j) = 0$ for all $j \notin B$ is basic solution associated to basis B.

Lemma 2.2

Each basic solution x^* is associated to (at least) one basis B

Proof

- Let $J = \{j \mid x^*(j) \neq 0\}$
- Columns of A_J are linearly independent
- Augment J to index set $B \supseteq J$ such that A_B is invertible
- One has $A_B x^*_B = b$ and $x^*(j) = 0$ for all $j \notin B$

Exercise

Show that a basic solution can be associated to two different bases.

A naive algorithm for linear programming

Let $\min\{c^T x \mid Ax = b, x \geqslant 0\}$ be a bounded LP

- Enumerate all bases $B \subseteq \{1, \ldots, n\}$ $\mathbf{O}(\binom{n}{m}) = \mathbf{O}(\mathbf{n}^{\mathbf{m}})$ many
- Compute associated basic solution x^* with $x^*_B = A_B^{-1} b$ and $x^*_{\overline{B}} = 0$
- Return the one which has smallest objective function value among the feasible basic solutions
- Running time $O(n^m \cdot m^3)$
- **Are there more efficient algorithms?**

Programming exercise (2P)

- Download the boost-library and install it (it is important that the directory "boost" that contains all the header files is in your include path; read installation notes!)
- Download all files from this directory
- Compile and run the file "toto.cc".
- Now implement the naive algorithm for linear programming and apply it to find an optimal solution of the Cashflow Management problem from the beginning of this lecture. (You have to transform the LP into standard form first!)
- Send your code, together with detailed compile-instructions, to Thomas Rothvoss (thomas.rothvoss@epfl.ch)
- Due date: Sept 30, 2009, By then we must have received your code and we have been able to compile and run it. It is a good idea to send us your code before this date.

Summary of todays lecture

Primary objectives:

- Linear programming: Example, Definition ✔
- The standard form ✔
- Basic feasible solutions ✔
- simplex algorithm
- Along the way: Develop your own toy-solver in C++ first steps
- Use of modeling languages and efficient LP-solvers
- Constructing dedicated bond-portfolio