



Norwegian University of
Science and Technology

TTK4135 – Lecture 5

Solving LPs – the Simplex method

Lecturer: Lars Imsland

Purpose of lecture

- Brief recap previous lecture
- The geometry of the feasible set
- Basic feasible points, “The fundamental theorem of linear programming”
- The simplex method
- Example 13.1
- Some implementation issues

Reference: N&W Ch.13.2-13.3, also 13.4-13.5

Linear programming, standard form and KKT: recap

$$\text{LP:} \quad \min_{x \in \mathbb{R}^n} c^T x \quad \text{subject to} \quad \begin{cases} a_i x = b_i, & i \in \mathcal{E} \\ a_i x \geq b_i, & i \in \mathcal{I} \end{cases}$$

$$\text{LP, standard form:} \quad \min_{x \in \mathbb{R}^n} c^T x \quad \text{subject to} \quad \begin{cases} Ax = b \\ x \geq 0 \end{cases}$$

$$\text{Lagrangian:} \quad \mathcal{L}(x, \lambda, s) = c^T x - \lambda^T (Ax - b) - s^T x$$

KKT-conditions (LPs: necessary *and* sufficient for optimality):

$$\begin{aligned} A^T \lambda^* + s^* &= c, \\ Ax^* &= b, \\ x^* &\geq 0, \\ s^* &\geq 0, \\ x_i^* s_i^* &= 0, \quad i = 1, 2, \dots, n \end{aligned}$$

Duality

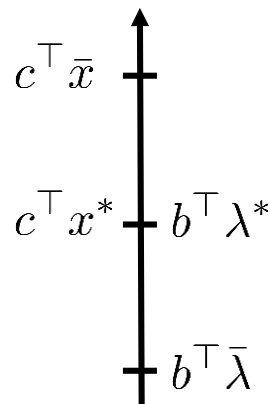
Primal problem

$$\begin{aligned} \min_x \quad & c^\top x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

Dual problem

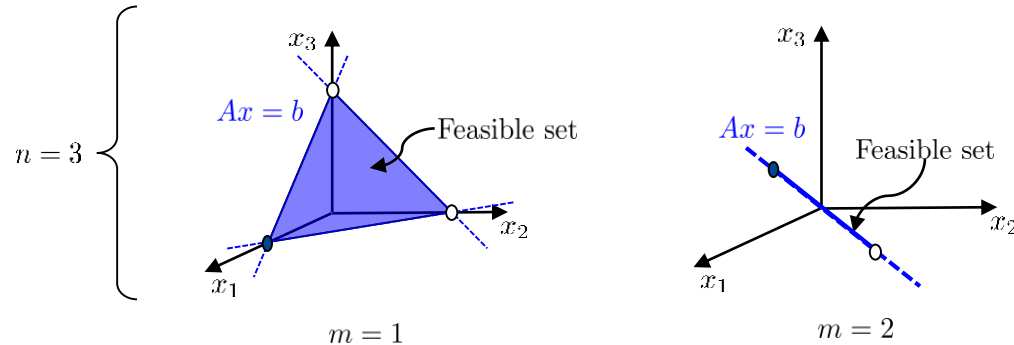
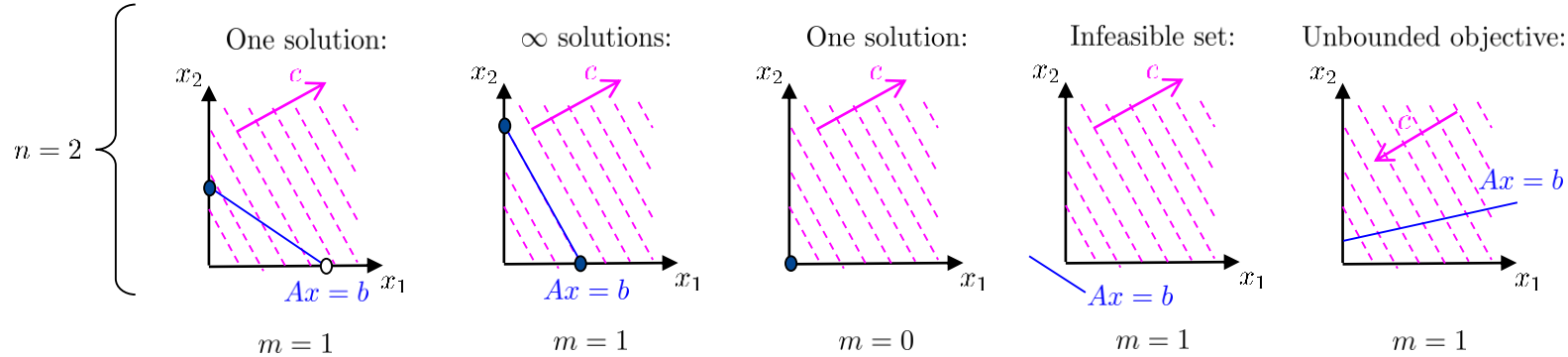
$$\begin{aligned} \max_{\lambda, s} \quad & b^\top \lambda \\ \text{s.t.} \quad & A^\top \lambda + s = c \\ & s \geq 0 \end{aligned}$$

- Primal and dual have same KKT conditions!
- Equal optimal value: $c^\top x^* = b^\top \lambda^*$
- Weak duality: $c^\top \bar{x} \geq c^\top x^* = b^\top \lambda^* \geq b^\top \bar{\lambda}$
- Duality gap: $c^\top \bar{x} - b^\top \bar{\lambda}$
- Strong duality (Thm 13.1):
 - i) If primal or dual has finite solution, both are equal
 - ii) If primal or dual is unbounded, the other is infeasible



LP: Geometry of the feasible set

$$\begin{array}{ll} \min_x & c^\top x \\ \text{s.t.} & Ax = b \\ & x \geq 0 \end{array}$$



- Basic optimal point (BOP)
- Basic feasible point (BFP) (if they exist)

In general, the BFP has at most m non-zero components

Basic feasible point (BFP)

A point x is a basic feasible point if

- x is feasible
- There is an index set $\mathcal{B}(x) \subset \{1, \dots, n\}$ such that
 - $\mathcal{B}(x)$ contains m indices
 - $i \notin \mathcal{B}(x) \Rightarrow x_i = 0$
 - $B = [A_i]_{i \in \mathcal{B}(x)}$ is non-singular
- $\mathcal{B}(x)$ is called a basis for the LP

Facts about Simplex method

The Simplex method generates iterates that are BFP, and converge to a solution if

- 1) there are BFPs, and
- 2) one of them is a solution (Basic Optimal Point, BOP)

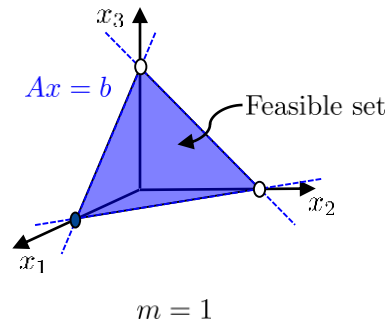
Theorem 13.2 (Fundamental theorem of Linear Programming): For standard for LP

- 1) If there is a feasible point, there is a BFP
- 2) If the LP has a solution, one solution is a BOP
- 3) If LP is feasible and bounded, there is a solution

Theorem 13.3: All vertices of the feasible polytope

$$\{x \mid Ax = b, x \geq 0\}$$

are BFPs (and all BFPs are vertices)



Facts about Simplex method, cont'd

Degeneracy: A LP is *degenerate* if there is a BFP x with $x_i = 0$ for some $i \in \mathcal{B}(x)$

Theorem 13.4: If an LP is bounded and non-degenerate, the Simplex method terminates at a BOP

Simplex definitions

LP KKT conditions (necessary&sufficient)

- Simplex method iterates BFPs until one that fulfills KKT is found.

$$A^T \lambda + s = c, \quad (\text{KKT-1})$$

$$Ax = b, \quad (\text{KKT-2})$$

$$x \geq 0, \quad (\text{KKT-3})$$

$$s \geq 0, \quad (\text{KKT-4})$$

$$x_i s_i = 0, \quad i = 1, 2, \dots, n \quad (\text{KKT-5})$$

- Each step is a move from a vertex to a neighboring vertex (*one change in the basis*), that decreases the objective

One step of Simplex-algorithm

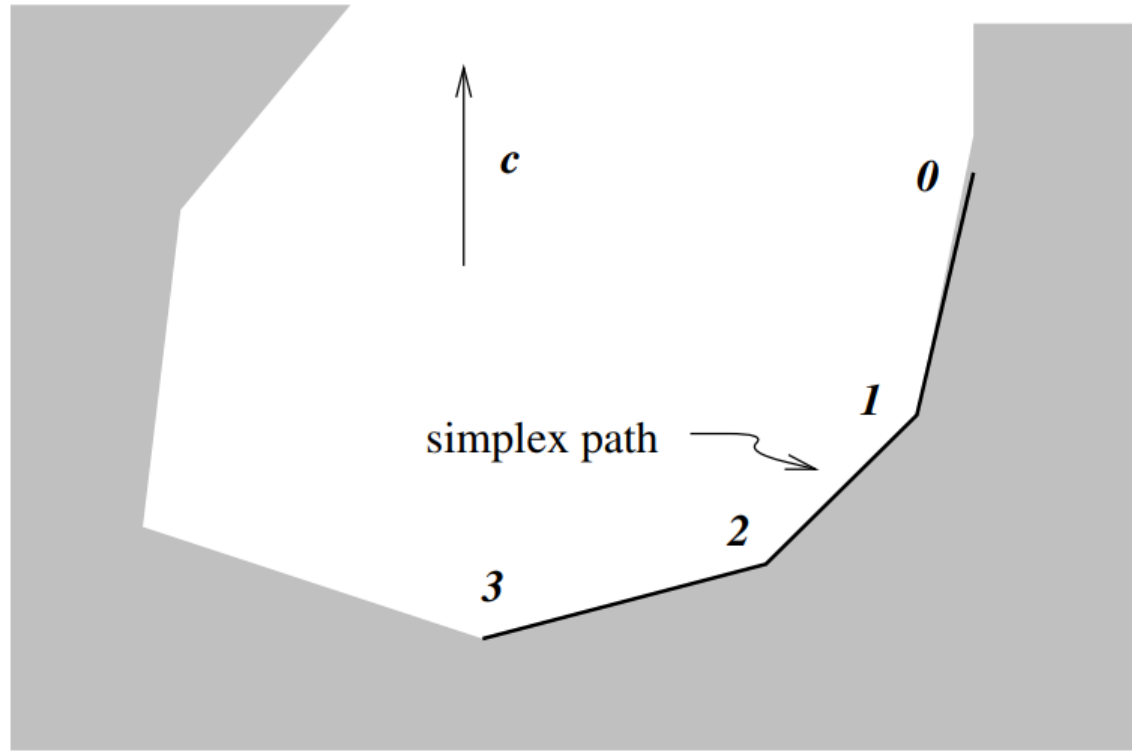
$$A^T \lambda + s = c, \quad (\text{KKT-1})$$

$$Ax = b, \quad (\text{KKT-2})$$

$$x \geq 0, \quad (\text{KKT-3})$$

$$s \geq 0, \quad (\text{KKT-4})$$

$$x_i s_i = 0, \quad i = 1, \dots, n \quad (\text{KKT-5})$$



Check KKT-conditions for BFP

- Given BFP x , and corresponding basis $\mathcal{B}(x)$. Define

$$\mathcal{N}(x) = \{1, 2, \dots, n\} \setminus \mathcal{B}(x)$$

- Partition x , s and c :

$$x_B = [x_i]_{i \in \mathcal{B}(x)} \quad x_N = [x_i]_{i \in \mathcal{N}(x)}$$

- KKT conditions

$$\text{KKT-2: } Ax = Bx_B + Nx_N = Bx_B = b \quad (\text{since } x \text{ is BFP})$$

$$\text{KKT-3: } x_B = B^{-1}b \geq 0, \quad x_N = 0 \quad (\text{since } x \text{ is BFP})$$

$$\text{KKT-5: } x^\top s = x_B^\top s_B + x_N^\top s_N = 0 \quad \text{if we choose } s_B = 0$$

$$\text{KKT-1: } \begin{bmatrix} B^T \\ N^T \end{bmatrix} \lambda + \begin{bmatrix} s_B \\ s_N \end{bmatrix} = \begin{bmatrix} c_B \\ c_N \end{bmatrix} \Rightarrow \begin{cases} \lambda = B^{-T} c_B \\ s_N = c_N - N^T \lambda \end{cases}$$

$$\text{KKT-4: Is } s_N \geq 0?$$

- If $s_N \geq 0$, then the BFP x fulfills KKT and is a solution
- If not, change basis, and try again
 - E.g. pick smallest element of s_N (index q), increase x_q along $Ax=b$ until x_p becomes zero. Move q from \mathcal{N} to \mathcal{B} , and p from \mathcal{B} to \mathcal{N} . This guarantees decrease of objective, and no “cycling” (if non-degenerate).

Ex. 13.1

Ex. 13.1, first iteration

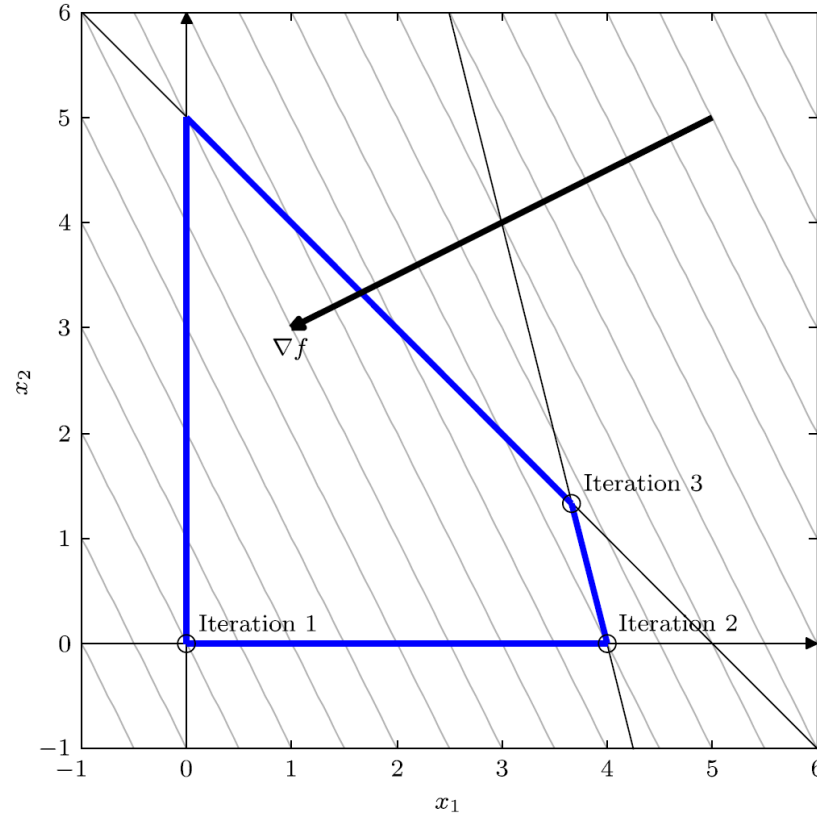
$$c^T = (-4 \quad -2 \quad 0 \quad 0), \quad A = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 2 & .5 & 0 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 5 \\ 8 \end{pmatrix}$$

Ex. 13.1, second iteration $c^\top = (-4 \quad -2 \quad 0 \quad 0)$, $A = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 2 & .5 & 0 & 1 \end{pmatrix}$, $b = \begin{pmatrix} 5 \\ 8 \end{pmatrix}$

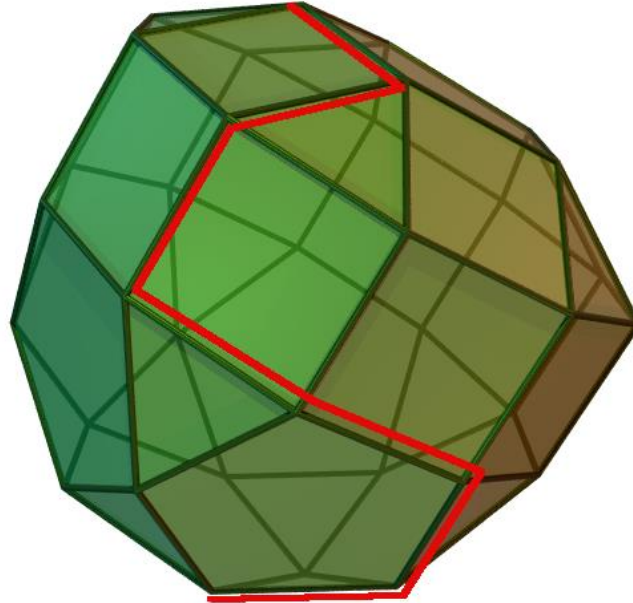
Ex. 13.1, third iteration

$$c^T = (-4 \quad -2 \quad 0 \quad 0), \quad A = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 2 & .5 & 0 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 5 \\ 8 \end{pmatrix}$$

Example 13.1 – figure



Simplex in 3D



[wikipedia.org](https://en.wikipedia.org/wiki/Simplex)

Linear algebra – LU factorization

- Two linear systems must be solved in each iteration:
 - $B^\top \lambda = c_B$
 - $Bd = A_q$ (to find the direction to check when increasing x_q)
 - We also have $Bx_B = b$. Since x_B is not needed in the iterations, we don't need to solve this (apart from in the final iteration)
 - This is the major work per iteration of simplex, efficiency is important!
- B is a general, non-singular matrix
 - Guaranteed a solution to the linear systems
 - LU factorization is the appropriate method to use (same for both systems)
 - Don't use matrix inversion!
- In each step of Simplex method, one column of B is replaced:
 - Can update ("maintain") the LU factorization of B in a smart and efficient fashion
 - No need to do a new LU factorization in each step, save time!

Other practical implementation issues (Ch. 13.5)

- Selection of “entering index” q
 - Dantzig’s rule: Select the index of the most negative element in s_N
 - Other rules have proved to be more efficient in practice
- Handling of degenerate bases/degenerate steps (when a positive x_q is not possible)
 - If no degeneracy, each step leads to decrease in objective and convergence in finite number of iterations is guaranteed (Thm 13.4)
 - Degenerate steps lead to no decrease in objective. Not necessarily a problem, but can lead to cycling (we end up in the same basis as before)
 - Practical algorithms use perturbation strategies to avoid this
- Starting the simplex method
 - We assumed an initial BFP available – but finding this is as difficult as solving the LP
 - Normally, simplex algorithms have two phases:
 - Phase I: Find BFP
 - Phase II: Solve LP
 - Phase I: Design other LP with trivial initial BFP, and whose solution is BFP for original problem

$$\min e^\top z \text{ subject to } Ax + Ez = b, \quad (x, z) \geq 0$$

$$e = (1, 1, \dots, 1)^\top, \quad E \text{ diagonal matrix with } \begin{cases} E_{jj} = 1 & \text{if } b_j \geq 0 \\ E_{jj} = -1 & \text{if } b_j < 0 \end{cases}$$

- Presolving (Ch. 13.7)
 - Reducing the size of the problem before solving, by various tricks to eliminate variables and constraints. Size reduction can be huge. Can also detect infeasibility.

Simplex – an active set method

- Complexity:
 - Typically, at most $2m$ to $3m$ iterations
 - Worst case: All vertices must be visited (exponential complexity in n)
 - Compare interior point method: Guaranteed polynomial complexity, but in practice hard to beat simplex on many problems
- Active set methods (such as simplex method):
 - Maintains explicitly an estimate of the set of inequality constraints that are active at the solution (the set \mathcal{N} for the simplex method)
 - Makes small changes to the set in each iteration (a single index in simplex)
- Next time: Active set method for QP