DM545

Linear and Integer Programming

Lecture 3
The Simplex Method: Exception Handling

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

1. Tableaux and Dictionaries

2. Exception Handling

Outline

Tableaux and Dictionaries

$$\begin{aligned} \max \ \sum_{j=1}^{n} c_j x_j & \\ \sum_{j=1}^{n} a_{ij} x_j & \le b_i, \ i = 1, \ldots, m \\ x_j & \ge 0, \ \ j = 1, \ldots, n \end{aligned}$$

$$\begin{aligned} x_{n+i} &= b_i - \sum_{j=1}^{n} a_{ij} x_j, \quad i = 1, \ldots, m \\ z &= \sum_{j=1}^{n} c_j x_j \end{aligned}$$

Tableau

$$\left[\begin{array}{c:c:c:c} I & \bar{A}_N & 0 & \bar{b} \\ \hdashline 0 & \bar{c}_N & 1 & -\bar{d} \end{array}\right]$$

$\bar{c}_N$ reduced costs

Dictionary

$$\begin{aligned} x_r &= \bar{b}_r - \sum_{s \notin B} \bar{a}_{rs} x_s, \quad r \in B \\ z &= \bar{d} + \sum_{s \notin B} \bar{c}_s x_s \end{aligned}$$

pivot op.:
choose col with r.c. >0
choose row with negative sign
update: express entering variable
and substitute in other rows

Example

$$\begin{array}{rrrrl}
\max & 6x_1 & + & 8x_2 & \\
 & 5x_1 & + & 10x_2 & \leq 60 \\
 & 4x_1 & + & 4x_2 & \leq 40 \\
 & & & x_1, x_2 & \geq 0
\end{array}$$

	x_1	x_2	x_3	x_4	$-z$	b
x_3	5	10	1	0	0	60
x_4	4	4	0	1	0	40
	6	8	0	0	1	0

$$\begin{array}{rcrcrcr}
x_3 & = & 60 & - & 5x_1 & - & 10x_2 \\
x_4 & = & 40 & - & 4x_1 & - & 4x_2 \\
\hline
z & = & & + & 6x_1 & + & 8x_2
\end{array}$$

...

	x_1	x_2	x_3	x_4	$-z$	b
x_2	0	1	$1/5$	$-1/4$	0	2
x_1	1	0	$-1/5$	$1/2$	0	8
	0	0	$-2/5$	-1	1	-64

$$\begin{array}{rcrcrcr}
x_1 & = & 2 & - & 1/5x_3 & + & 1/4x_4 \\
x_2 & = & 8 & + & 1/5x_3 & - & 1/2x_4 \\
\hline
z & = & 64 & - & 2/5x_3 & - & 1x_4
\end{array}$$

Outline

Exception Handling

1. Unboundedness
2. More than one solution
3. Degeneracies
   - benign
   - cycling
4. Infeasible starting

1. $F = \emptyset$
2. $F \neq \emptyset$ and $\exists$ solution
   1. one solution
   2. infinite solution
3. $F \neq \emptyset$ and $\nexists$ solution

Unboundedness

$$\begin{aligned} \max \ 2x_1 \ + \ & x_2 \\ x_2 & \leq 5 \\ -x_1 + x_2 & \leq 1 \\ x_1, x_2 & \geq 0 \end{aligned}$$

- Initial tableau

```
|    | x1 | x2 | x3 | x4 | -z | b |
|----+----+----+----+----+----+---|
| x3 |  0 |  1 |  1 |  0 |  0 | 5 |
| x4 | -1 |  1 |  0 |  1 |  0 | 1 |
|----+----+----+----+----+----+---|
|    |  2 |  1 |  0 |  0 |  1 | 0 |
```

- x_2 entering, x_4 leaving

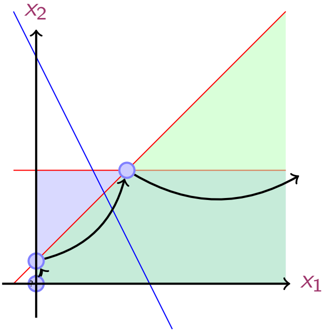
```
|             | x1 | x2 | x3 | x4 | -z |  b |
|-------------+----+----+----+----+----+----|
| II'=II-I'   |  1 |  0 |  1 | -1 |  0 |  4 |
| I'=I        | -1 |  1 |  0 |  1 |  0 |  1 |
|-------------+----+----+----+----+----+----|
| III'=III-I' |  3 |  0 |  0 | -1 |  1 | -1 |
```

$-x_1 + x_2 + x_4 = 1$, x_1 can increase without restriction,
$\theta = \min\{\frac{b_i}{a_{is}} : a_{is} > 0, i = 1 \ldots, n\}$

- x_1 entering, x_3 leaving

```
|              | x1 | x2 | x3 | x4 | -z |   b |
|--------------+----+----+----+----+----+-----|
| I'=I         |  1 |  0 |  1 | -1 |  0 |   4 |
| II'=II+I'    |  0 |  1 |  1 |  0 |  0 |   5 |
|--------------+----+----+----+----+----+-----|
| III'=III-3I' |  0 |  0 | -3 |  2 |  1 | -13 |
```

  x_4 was already in basis but for both I and II ($x_3 + 0x_4 = 5$), x_4 can increase arbitrarily



∞ solutions

$$
\begin{array}{lrcrcl}
\max & x_1 & + & x_2 & & \\
 & 5x_1 & + & 10x_2 & \leq & 60 \\
 & 4x_1 & + & 4x_2 & \leq & 40 \\
 & & & x_1, x_2 & \geq & 0
\end{array}
$$

- Initial tableau

	x1	x2	x3	x4	-z	b
x3	5	10	1	0	0	60
x4	4	4	0	1	0	40
	1	1	0	0	1	0

- x_2 enters, x_3 leaves

	x1	x2	x3	x4	-z	b
I'=I/10	1/2	1	1/10	0	0	6
II'=II-4Ix4	2	0	-2/5	1	0	16
III'=III-I	1/2	0	-1/6	0	1	-6

- x_1 enters, x_4 leaves

	x1	x2	x3	x4	-z	b
I'=I-II'/2	0	1	1/5	-1/4	0	2
II'=II/2	1	0	-1/5	1/2	0	8
III'=III-II'/2	0	0	0	-1/4	1	-10

$\vec{x} = (8, 2, 0, 0), z = 10$
nonbasic variables typically have reduced costs $\neq 0$. Here x_3 has r.c. $= 0$. Let's make it enter the basis

- x_3 enters, x_2 leaves

	x1	x2	x3	x4	-z	b
I'=5I	0	5	1	-5/4	0	10
II'=II+I'/5	1	1	0	4	0	10
III'=III	0	0	0	-/14	1	-10

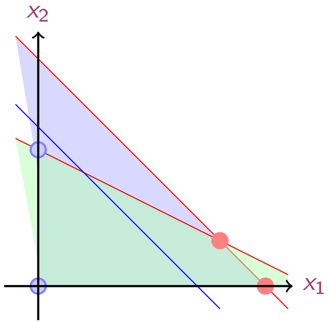
$\vec{x} = (10, 0, 10, 0), z = 10$
There are 2 optimal solutions $\rightsquigarrow$ all their convex combinations are optimal solutions:

$$\vec{x} = \sum_i \alpha_i \vec{x}_i \qquad \vec{x}^1 = (8, 2, 0, 0) \qquad x_1 = 8\alpha + 10(1-\alpha)$$

$$\alpha_i \geq 0 \qquad \vec{x}^2 = (10, 0, 10, 0) \qquad x_2 = 2\alpha$$

$$\sum_i \alpha_i = 1 \qquad x_3 = 10(1-\alpha)$$

$$x_4 = 0$$



Degeneracy

$$\begin{array}{rcrcl} \max & & x_2 & & \\ -x_1 & + & x_2 & \leq & 0 \\ x_1 & & & \leq & 2 \\ & & x_1, x_2 & \geq & 0 \end{array}$$

- Initial tableau

```
|    | x1 | x2 | x3 | x4 | -z | b |
|----+----+----+----+----+----+---|
| x3 | -1 |  1 |  1 |  0 |  0 | 0 |
| x4 |  1 |  0 |  0 |  1 |  0 | 2 |
|----+----+----+----+----+----+---|
|    |  0 |  1 |  0 |  0 |  1 | 0 |
```

$b_i = 0$ (one basic var. is zero) might lead to cycling

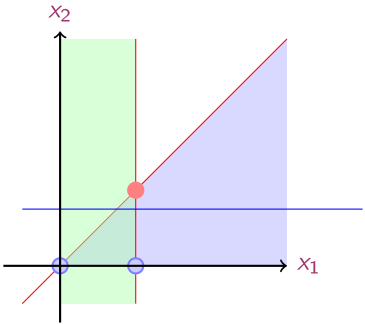
- degenerate pivot step: not improving, the entering variable stays at zero

```
|   | x1 | x2 | x3 | x4 | -z | b |
|---+----+----+----+----+----+---|
|   | -1 |  1 |  1 |  0 |  0 | 0 |
|   |  1 |  0 |  0 |  1 |  0 | 2 |
|---+----+----+----+----+----+---|
|   |  1 |  0 | -1 |  0 |  1 | 0 |
```

- now nondegenerate:

```
|   | x1 | x2 | x3 | x4 | -z |  b |
|---+----+----+----+----+----+----|
|   |  0 |  1 |  0 |  1 |  0 |  2 |
|   |  1 |  0 |  0 |  1 |  0 |  2 |
|---+----+----+----+----+----+----|
|   |  0 |  0 | -1 | -1 |  1 | -2 |
```

$x_1 = 2, x_2 = 2, z = 2$



$\geq n+1$ constraints meet at a vertex

Def: Improving variable, one with positive reduced cost

Under certain pivoting rules cycling can happen. So far we chose an arbitrary improving variable to enter.

Degenerate conditions may appear often in practice but cycling is rare and some pivoting rules prevent cycling. (Ex. 2 Sheet 2 shows the smallest possible example)

Theorem

If the simplex fails to terminate, then it must cycle.

Proof:

- there is a finite number of basis and simplex chooses to always increase the cost
- hence the only situation for not terminating is that a basis must appear again. Two dictionaries with the same basis are the same (related to uniqueness of basic solutions)

Pivot Rules

Rules for breaking ties in selecting entering improving variables (more important than selecting leaving variables)

- Largest Coefficient: the improving var with largest coefficient in last row of the tableau.
  Original Dantzig's rule, can cycle

- Largest increase: absolute improvement: $\text{argmax}_j\{c_j\theta_j\}$
  computationally more costly

- Steepest edge the improving var whose entering into the basis moves the current basic feasible sol in a direction closest to the direction of the vector $\vec{c}$ (ie, maximizes the cosine of the angle between the two vectors):

$$\mathbf{a}\cdot\mathbf{b} = \|\mathbf{a}\|\,\|\mathbf{b}\|\cos\theta \quad\Longrightarrow\quad \max \frac{c^T(x_{\text{new}} - x_{\text{old}})}{\|\, x_{\text{new}} - x_{\text{old}}\,\|}$$

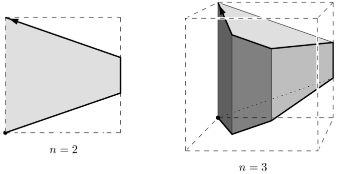
- Bland's rule choose the improving var with the lowest index and, if there are more than one leaving variable, the one with the lowest index
  Prevents cycling but is slow

- Random edge select var uniformly at random among the improving ones

- Perturbation method perturb values of b_i terms to avoid $b_i = 0$, which must occur for cycling.
  To avoid cancellations: $0 < \epsilon_m \ll \epsilon_{m-1} \ll \cdots \ll \epsilon_1 \ll 1$
  can be shown to be the same as lexicographic method, which prevents cycling

Efficiency of Simplex Method

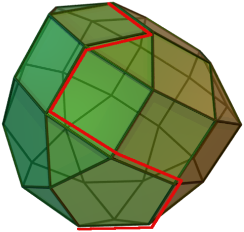
- Trying all points is $\approx 4^m$
- In practice between $2m$ and $3m$ iterations
- Klee and Minty 1978 constructed an example that requires $2^n - 1$ iterations:



- random shuffle of indexes + lowest index for entering + lexicographic for leaving: expected iterations $< e^{C\sqrt{n \ln n}}$

Efficiency of Simplex Method

- unknown if there exists a pivot rule that leads to polynomial time.
- Clairvoyant's rule: shortest possible sequence of steps
  Hirsh conjecture $O(n)$ but best known $n^{1+\ln n}$



- smoothed complexity: slight random perturbations of worst-case inputs
  D. Spielman and S. Teng (2001), *Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time*
  $O(max(n^5 \log^2 m, n^9 log^4 n, n^3 \sigma^{-4}))$