INSTITUT FÜR MATHEMATIK UND
WISSENSCHAFTLICHES RECHNEN

Optimierung I

Übungsblatt 4

Bearbeitung bis 8. April 2014

Aufgabe 4.1: [Application of the simplex algorithm]

Consider the following problem:

$$\min_{x\in\mathbb{R}^6} \; -3x_1 - x_2 - 3x_3, \text{ subject to: } \begin{cases} 2x_1 + x_2 + x_3 + x_4 = & 2 \\ x_1 + 2x_2 + 3x_3 + x_5 = & 5 \\ 2x_1 + 2x_2 + x_3 + x_6 = & 6 \\ x \geq & 0 \end{cases} \tag{1}$$

The problem is already in a canonical form, with the basic variables x_4, x_5, x_6. Solve it with the simplex algorithm, by introducing in the basis the following variables, in this order: x_2, x_3, x_1. We give here a first iteration of the algorithm.

Initial table:

a_1	a_2	a_3	a_4	a_5	a_6	b
2	1	1	1	0	0	2
1	2	3	0	1	0	5
2	2	1	0	0	1	6
−3	−1	−3	0	0	0	0

- Basic variables: x_4, x_5, x_6, basic solution: $(0, 0, 0, 2, 5, 6)$
- Cost (as a function of non-basic variables): $-3x_1 - x_2 - 3x_3$
- New basic variable: x_2, removed variable: x_4 (since $2/1 < 5/2 < 6/2$).
- Operations of the rows: $r_2 \leftarrow r_2 - 2r_1$, $r_3 \leftarrow r_3 - 2r_1$, $r_4 \leftarrow r_4 + r_1$ (r_1 means row 1, "$\leftarrow$" means "replaced by").

New table:

a_1	a_2	a_3	a_4	a_5	a_6	b
2	1	1	1	0	0	2
−3	0	1	−2	1	0	1
−2	0	−1	−2	0	1	2
−1	0	−2	1	0	0	2

Realize the next two iterations of the algorithm and use the same scheme as the one described above.

Aufgabe 4.2: [Non-degeneracy condition]

Let the matrix $A \in \mathbb{R}^{m\times n}$ and the vector $b \in \mathbb{R}^m$ reflect the equality constraints in a linear program in standard form. Show that if each $m \times m$ submatrix of $[A|b]$ has full rank, then the linear program satisfies the non-degeneracy condition. Give a non-trivial example.

Aufgabe 4.3: [Proving the non-boundedness with the Simplex algorithm]

Consider the following linear problem:

$$\min_{x\in\mathbb{R}^3} \; x_1 - 3x_2 + x_3, \text{ subject to: } \begin{cases} 5x_1 - x_2 - 4x_3 \le & 9 \\ 3x_1 + x_2 - 4x_3 \le & 4 \\ x \ge 0. \end{cases} \tag{2}$$

Put the problem into a canonical form and show, by applying the Simplex algorithm, that the optimal value is $-\infty$. Use the scheme as in the first exercise to describe the pivoting operations. Finally, describe a half-line of the feasible space which is such that the cost decreases along.

Aufgabe 4.4: [New variable selection]
For a linear program in standard form given by A, b, and c, let the simplex tableau in some step of the simplex algorithm reads as:

$$\begin{array}{cccccccc|c}
a_1 & a_2 & \dots & a_{m-1} & a_m & a_{m+1} & \dots & a_n & b \\
\hline
1 & 0 & \dots & \dots & 0 & y_{1,m+1} & \dots & y_{1,n} & y_{1,0} \\
0 & 1 & \ddots & & \vdots & y_{2,m+1} & & y_{2,n} & y_{2,0} \\
\vdots & \ddots & \ddots & \ddots & \vdots & \vdots & & \vdots & \vdots \\
\vdots & & \ddots & 1 & 0 & y_{m-1,m+1} & \dots & y_{m-1,n} & y_{m-1,0} \\
0 & \dots & \dots & 0 & 1 & y_{m,m+1} & \dots & y_{m,n} & y_{m,0} \\
\hline
0 & \dots & \dots & \dots & 0 & r_{m+1} & \dots & r_n & -z_0.
\end{array} \tag{3}$$

We denote, for all $k = m+1,...,n$,

$$\mu_k = \min\left\{\frac{y_{i,0}}{y_{i,k}}, \forall i = 1,...,m, \text{ such that } y_{i,k} > 0\right\}. \tag{4}$$

We finally denote:

$$\lambda = \min_{k=m+1,...,n} \{r_k \mu_k\}. \tag{5}$$

Note that if for some k, there is no coefficient i such that $y_{i,k} > 0$, then we consider that $\mu_k = \infty$ and that

$$r_k\mu_k = \begin{cases} 0 & \text{if } r_k = 0, \\ +\infty & \text{if } r_k > 0 \\ -\infty & \text{if } r_k < 0. \end{cases} \tag{6}$$

i) What is the maximum number of needed operations (divisions and comparison operations) to compute λ ?

ii) What can we say if $\lambda > 0$? What if $\lambda = -\infty$?

iii) We assume that $-\infty < \lambda \le 0$. Let k^* be such that

$$\lambda = r_{k^*}\mu_{k^*}. \tag{7}$$

Show that k^* leads to the greatest reduction of the objective function.

Aufgabe 4.5: [The simplex algorithm] Programming Exercise
Write an implementation of the simplex algorithm which takes the problem data A, b and c as well as a set of basic variables s associated with a feasible basic solution and returns either a solution of the linear program or stops with an error in case the minimization problem has no solution. Use a pivoting strategy which always selects the variable associated with a minimal relative cost coefficient.
The implementation may be structured as follows:

- `T = simplex_tableau_init(A, b, c, s)`

  which returns the initial simplex tableau T associated with A, b, c and the basic variables selected by s (s can, for instance, be a vector of length n indicating with 1 and 0, respectively, whether a variable is basic or not),

- `[T, e] = simplex_tableau_update(T)`

  which updates the tableau T according to the simplex algorithm if possible or indicates, with the help of e, that the current solution is already optimal or that there is no solution (also, it should give a warning if the current basic solution is degenerate),

- `[x, e] = simplex_algorithm(A, b, c, s)`

  which assembles the above two subroutines to return either a solution x or indicates an error with e.

Test the implementation with the linear program of the Problems 4.1 and 4.3 and verify that it yields the same results.

Remark concerning programming exercises:

- All Programs should be written for Matlab/Octave and send to the group leader at least 2 hours before class in the following form:
  - Put all code together in one single file named by your surname followed by the first letter of your first name, for example: *hollerm.m* or *pfeifferl.m.*
  - The file should be programmed as a function, following a predefined input-output structure.

    For the present exercise 4.5, it should take the matrix A, the vector b, the cost c and a boolean vector s, indicating the basic variables, as input and give a solution x and an error indicator e as output, where e equals 0 if the problem has been solved correctly, 1 if one of the basic solutions became degenerate and 2 if the problem has no solution. An example structure is: [x,e] = hollerm(A,b,c,s).
  - Send the file to your group leader by mail (as attachment), having exactly *Optimierung 1, Abgabe x* in the subject, where x stands for the exercise number, i.e. 4.5 in this case.
- The programs will be automatically tested. In the corresponding exercise we will bring the code of one student and ask him or her to present the code on the beamer.
- It is mandatory to hand in and cross at least 50% of the programming exercises (marked as programming exercise on the right). There will be roughly 5 to 6 programming exercises during the semester. For each successfully completed programming exercise that goes beyond 50% you will get an additional point.