Compulsory project 2 (2013)
INF-MAT3370 [1]

- **General information:** You should give/send your project by **April 12.** to

  Nikolai Bjørnestøl Hansen [nikolai.bjornestol.hansen@gmail.com]

  either (i) by email in a single pdf-file denoted "username.pdf" (your username!) and program files with names "username_interior.m" and "username_simplex.m", or (ii) give a paper print to Nikolai, or (iii) use the Devilry system. You should also read the general information about compulsory projects at the course web page.

This is mostly a numerical project on linear optimization algorithms. The goal is to learn more about these algorihms by implementing simplified versions. We focus on (the matrix version of) the primal simplex algorithm and an interior point algorithm, see below. But, first, there is a theoretical question.

1 A theoretical question (or two)

Consider the LP problem

$$\text{(P)} \quad \max \{c^T x : Ax \leq b\}$$

where A is an $m \times n$ matrix, and b and c are vectors of length m and n, respectively.

(i) Find the dual (D) of (P).

(ii) Show that the dual of (D) is equivalent to (P). (Hint: write (P) in the standard form used in the book by replacing x by some nonnegative variables.)

(iii) Consider the special case of (P) where $c = O$ (the zero vector). So solving (P) really means finding a feasible solution, i.e., an x satisfying $Ax \leq b$. Use the dual (D) to characterize whenever $Ax \leq b$ has a solution.

[1] Geir Dahl, UiO, geird@math.uio.no

2 Algorithms/implemention

Your task is to implement the following two LP algorithms:

- **Algorithm S**: the primal simplex algorithm.
- **Algorithm I**: the path-following interior point algorithm.

Here are further details on this:

- You may choose any programming language you like, but we recommend MATLAB (or perhaps Python) as it makes the programming more easy.
- Algorithm S is explained in detail in Section 6.2 in Vanderbei (p.93–95). This is the matrix version of the primal simplex algorithm. You need to solve some linear systems of equations and for this you may use Matlab's command $V \setminus d$ for solving $Vx = d$.
- Algorithm I is given i Fig. 18.1 (p.308) in Vanderbei and it is explained in Chapters 17, 18 and 19. (Actually, only chapter 17 is included in our syllabus, but you just need to read a few lines in chapters 18 and 19 to know what to do.) The main part is to solve the equations called the *KKT-system* (Karush-Kuhn-Tucker system, the optimality conditions for the barrier problem) to find the steps Δx, Δy, Δz and Δw. This can be done in different ways, but I suggest using the *normal equations in primal form*, see start of section 19.2, eq. (19.9) for finding Δy, then find Δx etc.
- You can concentrate on LP problems in the standard form

$$\max\{c^T x : Ax \le b,\ x \ge O\}$$

  where the vectors b and c and the matrix A are given $(m \times n)$. For simplicity we only work with matrices $A \ge O$ and assume that $b \ge O$, so $x = O$ is a feasible solution.
- Report your code and a test run, for both algorithms, based on the data file on the course web page (print a line or two with useful information for each (or maybe every second) iteration.

Good luck!