# Computational Optimization - MSc AIDA UoM

Professor Nikolaos Samaras

Academic Year: 2021-2022

The discovery of the simplex algorithm led to a series of important questions and results. Borgwardt [1], [2] proved that the expected number of iterations to solve a linear problem using a simplex-type algorithm is polynomial when applied to real problems. Earlier, Klee-Minty [3] showed that the worst-case behavior of the simplex is exponential. Since then, many efforts have been made to find better (faster) ways to solve linear problems. These efforts led to the discovery of efficient interior point methods [4], and [5]. These methods are faster than the simplex algorithm for large-scale problems. However, the development of pivoting algorithms that are faster than the simplex remains a subject of great scientific interest. One way to improve the computational behavior of simplex-type algorithms is by moving to non-adjacent improving vertices. This movement can be achieved by avoiding the boundary of the polyhedron

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

and constructing basic solutions that are not feasible. Such an algorithm is called the exterior point simplex algorithm.

**Tasks:**

**[A.]** Write code in the Python programming language that implements the exterior points algorithm, as presented in lecture 5. Specifically, you should only program Phase II of the exterior points algorithm. If a feasible partition needs to be computed for a problem, then use scipy.optimize.linprog and specify that you want only Phase I to be executed. Then, your code will accept the Phase I problem and execute Phase II.

**[B.]** Run a mini computational study on the following random sparse (approximately 10% density) optimal linear problems. First, you need to convert them to matrix form with your work from week 1. For each problem, you should record the CPU time and the number of iterations made by the algorithm. There is no need to use Phase I from the scipy library.

| ID | Name | Constraints | Variables | Optimal Value |
|---|---|---|---|---|
| 1 | sdata1_100x100 | 500 | 500 | -143,4570 |
| 2 | sdata1_200x200 | 200 | 200 | -81,2455 |
| 3 | sdata1_300x300 | 300 | 300 | -98,9077 |
| 4 | sdata1_400x400 | 400 | 400 | -73,8266 |
| 5 | sdata1_500x500 | 500 | 500 | -80,5277 |

**Note:** The inverse can be computed either using a ready-made function in Python or using the eta-matrix.

**References**

1. Borgwardt, H.K. (1982). "Some distribution independent results about the asymptotic order of the average number of pivot steps in the simplex method", Mathematics of Operations Research, Vol. 7(3), pp. 441-462.

2. Borgwardt, H.K. (1982). "The average number of pivot steps required by the simplex method is polynomial", Zeitschrift fur Operational Research, Series A: Theory, Vol. 26(5), pp. 157-177.

3. Klee, V., Minty, G. (1972). "How good is the simplex algorithm?", In Inequalities III, Shisha, O. (Ed.), Academic Press, New York, pp. 159-179.

4. Gondzio, J. (1992). "Splitting dense columns of constraint matrix in interior point methods for large-scale linear programming", Optimization, Vol. 24, pp. 285-297.

5. Gondzio, J. (1996). "Multiple centrality corrections in a primal-dual method for linear programming", Computational Optimization and Applications, Vol. 6, pp. 137-156.