

## MCS 471 Project Three: Numerical Aspects of Linear Programming

The goal of the project is to investigate some numerical aspects of linear programming, and in particular the simplex algorithm.

In our experiments we will use MATLAB or Octave. MATLAB is available in all computer labs on campus, but is commercial and expensive. Octave is similar to MATLAB (for our work, Octave runs just as fine), and is free to download from <http://www.octave.org/>.

### 0. Matrices in MATLAB or Octave

Matrices are the basic data structure in MATLAB and Octave. To select rows 2 to 4 from a matrix  $A$ , type  $A(2:4, :)$ . Notice the right quote to perform a transpose:  $x'$  turns the column (respective row) vector into a row (respective column) vector. The normal arithmetical operations are naturally extended to matrices and vectors.

We can group commands into  $.m$  files, defining functions. The name of the function should match the file name. For this project, we need the function “simplex” (and six subroutines: “initialize”, “entering”, “leaving”, “diagonalize”, “passing”, and “solution”) with its definition in the file “simplex.m” (download this file and also “initialize.m”, “entering.m”, “leaving.m”, “diagonalize.m”, “passing.m”, and “solution.m” from the class web site). If the path is set correctly (do **help path** otherwise to see how to set the search path), then you can call this function just as a regular command.

### 1. The Linear Programming Problem

The problem we consider is to maximize a linear function, subject to linear equalities, e.g.:

$$\begin{aligned} & \max_{x,y} 143x + 60y \\ & \text{subject to } \begin{cases} x + y \leq 75 \\ 110x + 30y \leq 4,000 \\ 120x + 210y \leq 15,000 \\ x \geq 0, y \geq 0 \end{cases} \end{aligned} \quad (1)$$

Using compact matrix vector notation, this problem is denoted as

$$\begin{aligned} & \max_z f^T z \\ & \text{subject to } \begin{cases} Az \leq b \\ z \geq 0 \end{cases} \quad \text{with } z = \begin{bmatrix} x \\ y \end{bmatrix}, f = \begin{bmatrix} 143 \\ 60 \end{bmatrix}, A = \begin{bmatrix} 1 & 1 \\ 110 & 30 \\ 120 & 210 \end{bmatrix}, b = \begin{bmatrix} 75 \\ 4000 \\ 15000 \end{bmatrix}. \end{aligned} \quad (2)$$

The 7th edition of our textbook describes linear programming in section 7.3, starting on page 428.

### 2. Numerical Experiments

In the following assignments we will first study the geometry of the problem, before investigating the numerical stability and numerical conditioning.

#### 2.1 A geometric interpretation

For two variables – and to some extent also for three variables – we can solve a linear programming problem graphically. The purpose of the first assignment is to create a linear programming problem for which we can graph the execution of the simplex algorithm.

**Assignment One.** Draw a septagon in the plane. Two of the edges must be the coordinate axes  $x_1 = 0$  and  $x_2 = 0$ . Choose the other edges of the septagon at random. Generate coordinates for the

vertices of the septagon, using floating-point numbers of at least 4 significant decimal places long. The mantissa of each coordinate (except the (0,0) of course) must be at least 1000. Compute the equations for the lines defining the seven edges of the septagon. Scale the coordinates so that the largest coefficient of the system  $Ax = b$  equals one.

Choosing the proper orientation of the right-hand side vector, we define a system of linear inequalities  $Ax \leq b, x \geq 0$  whose solution set is the septagon. Use **rand(2,1)** to generate a random objective function  $f$  and use the simplex routine to solve the problem. Do this experiment *ten* times (each time generating a new objective function  $f$ ) and record the number of pivoting steps the algorithm executes.

What is the average number of steps? Draw one typical trajectory of the vertices visited while going to the optimal solution. Draw the line defined by  $f$  through the optimal solution.

### 2.2 Numerical Stability of Pivoting

The choice of the pivot is guided by the optimization criterion, i.e.: we bring in that variable which will give the largest increase in the objective function (as done by the entering function).

**Assignment Two.** Give a concrete example (e.g., modify the septagon) where the choice of a small pivot creates huge numbers. Demonstrate in detail how the calculations go wrong. Could one improve the numerical stability? Illustrate.

### 2.2 Numerical Conditioning

In the lectures we gave a geometrical interpretation of the condition of a linear system in two variables, i.e.: when the two equations are almost parallel to each other, the linear system is ill conditioned. The purpose of this assignment is to see the conditioning of a linear programming problem.

**Assignment Three.** Give an example of a linear programming problem (e.g. modify the septagon) where the optimal solution is the solution of an ill conditioned linear system. Use the command **cond** to estimate the condition number of the linear system at the optimal solution. Make sure the condition number is at least  $10^8$ .

Run the simplex method on this example and report what happens. Try to find a critical value for the condition number, i.e.: after the condition number passes this critical value, the solution can no longer be trusted. Justify and illustrate.

Can you think of a geometrical transformation to remedy the numerical conditioning?

## 3. The deadline is Friday 11 March 2005 at 10AM

Bring *your* solution to the project to class. The *your* is emphasized to stress that your solution is the result of an *individual* effort. Collaborations are **not** permitted.

Your solution should contain the following:

1. Answers to the questions in the assignments. Please write complete grammatically correct sentences.
2. Tables summarizing the numerical experiments you have done. Use the **format short e** command when generating data, so your numbers in the tables will also be in scientific notation.
3. Sequences of commands used to generate the data for the experiments. It is good practice to store these commands in little .m files, e.g., in `assignment_one.m`, `assignment_two.m`, etc.
4. Output of your sessions with MATLAB or Octave as *an appendix*. Typing **diary** followed by the name of a file in a session creates a new file with the given name which will contain everything you see on the screen during the session. You may edit out mistakes from the output of diary.

The solution to the project is essentially a report on paper.

If you have questions or difficulties with the assignments, feel free to come to my office for help.