AMSC 607 / CMSC 764 Fall 2006
HMWK 3: Due November 2

Show all work. All work must be your own (i.e., no group efforts are allowed). If you use a reference book, cite it, or you will lose credit!

In this homework you will program a feasible direction method and an interior point method for solving linear programming problems

$$\min_x c^T x$$

$$Ax = b$$

$$x \geq 0$$

where $x \in \mathcal{R}^n$ and $b \in \mathcal{R}^m$ with $m < n$. Assume a constraint qualification.

The feasible direction method. For this method, see the feasible direction notes, pp.6-8. This is the simplex method except that we allow iterates that are not vertices of the feasible set.

Write a Matlab function `xopt = lpfeasdir(A,b,c,x)`. The parameters to your feasible direction algorithm are A, b, c, and an initial feasible point x.

- Use `qrupdate` (instead of the B and N method in the notes) to update a factorization of $\hat{A}^T$, where the rows of the basis matrix $\hat{A}$ are the rows of A and the rows of the identity matrix corresponding to the currently active inequality constraints $x \geq 0$.
- At each iteration, $\hat{A}^T$ gains one column, and it may also lose one: if there is no feasible downhill direction, remove the column corresponding to the most negative (estimated) Lagrange multiplier.
- The next point is $x + \alpha p$, where p is determined from solving the system involving a column of the identity matrix, and α defines the longest step that is possible without violating any of the constraints $x \geq 0$. The constraint that we hit becomes the added one.
- Stop when there is no feasible downhill direction.

The IPM. For this method, see the LP-IPM notes, pp. 9-10. Write a Matlab function `xopt = lpipm(A,b,c,x,y,z,eta)`. The parameters to your IPM are A, b, c, a convergence tolerance η, and an initial feasible point x, y, z.

Stop the iteration when the duality gap is less than η. Use `qr` to solve the least squares problem (or linear system) at each iteration.

Test your algorithms on the problem

$$\begin{aligned} \min -x_1 - 2x_2 & \\ -2x_1 + x_2 + x_3 &= 2 \\ -x_1 + 2x_2 + x_4 &= 7 \\ x_1 + 2x_2 + x_5 &= 3 \\ x_1, x_2, x_3, x_4, x_5 &\geq 0 \end{aligned}$$

Start the feasible direction algorithm at the point $[0, 0, 2, 7, 3]^T$. For the IPM, start at the point $x = [.5, .5, 2.5, 6.5, 1.5]^T$, $y = [-1, -1, -5]^T$, $z = [1, 11, 1, 1, 5]^T$. (Check that these starting points satisfy the required conditions.) For each algorithm, print your sequence of iterates and the value of $c^T x$ at each iteration.

Grading: 50 points total.

- 10 points for the efficient implementation of each of the algorithms as a bug-free Matlab function, with good documentation for the calling sequence and the algorithm. (20 points total)
- 5 points for the output of testing both algorithms on the problem given above. (5 points total)
- 10 points for a count of the work per iteration (in terms of m and n) for the feasible direction method and the IPM. (10 points total)
- 15 points for testing the algorithms on a larger linear programming problem (which will appear on the website soon) and discussing the results – comparing the time and operations counts for the two methods and the accuracy achieved in the solution. For the IPM, include the effects of different values of η.

Note. Let `A` and `B` be matrices, and let `c` be a vector. Make sure you understand why the statements `A*(B*c)` and `A \ (B*c)` take much less time than `A*B*c` and `A \ B * c`, and then use this knowledge in your programming.