CMPSCI 690: Optimization for Computer Science
HomeFun 1

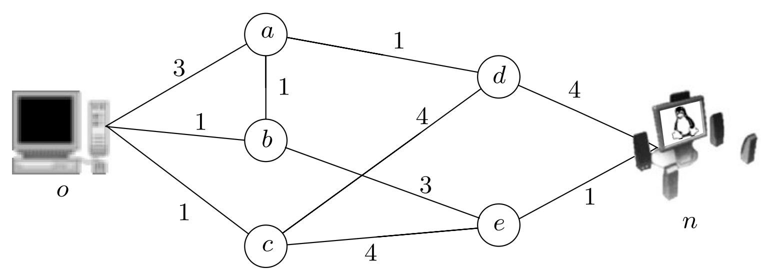
Due: Submit on Moodle by March 15 2013

Abstract

This is the first "home fun" assignment based on Lectures 1 through 6. Solutions should preferably be *computer formatted*. This is a fairly **long** assignment. It cannot be done over one weekend. Begin early. Late submissions will be penalized at 10% of the grade per day. If you cannot finish the assignment, turn in what ever you have managed to complete. A partial grade is better than no grade! You may discuss the questions with other students taking this class, but your final answers must be your own. You can also discuss these questions on the class forum on `piazza.com`. Any plagiarism will be dealt with following the Department of Computer Science policy on cheating.

Question 1 (20 points)

A graduate student in a CS lab wants to convince his/her advisor to purchase a new state of the art computer system because (secretly) h/she wants to transfer a large music collection from the older computer to the newer one using the local network in the lab. The network is shown below (o represents the older computer and n represents the newer computer).



The number on each data link specifies the maximum transfer rate of that link (in Mbytes/second, say). Assume that each link can transfer data in either direction, but not in both directions simultaneously. For example, the link from a to b can either transmit from a to b at a rate up to 1Mbyte per second or from b to a at up to this rate, but not in both directions simultaneously. Data cannot be stored at any of the intermediate nodes, so all data entering a node has to be transmitted immediately (this constraint is to avoid alerting other graduate students in the lab of this secret project). The student wants to compute the maximum transfer rate from the old computer o to the new computer n to decide how long this transfer will take (and consequently decide when to carry out this nefarious activity). Formulate this as an optimization problem.

- What are the variables in the problem?
- Write out the cost function and the constraints.
- Express this linear programming problem in standard form $\min_x c^T x$ such that $Ax = b, x \geq 0$.

Question 2 (20 points)

A refinery takes four types of raw gasoline and blends them to produce three types of fuel. The data for this process is given below.

Raw Gas Type	Octane Rating	Available Barrels per Day	Price Per Barrel
1	68	4000	\$31.02
2	88	5050	\$33.15
3	91	7100	\$36.35
4	99	4300	\$38.75

The company sells raw gasoline not used in making fuels at \$38.95 per barrel if its octane rating is over 90 and \$36.85 per barrel if its octane rating is less than 90. The demand for the various types of blended fuel is as follows:

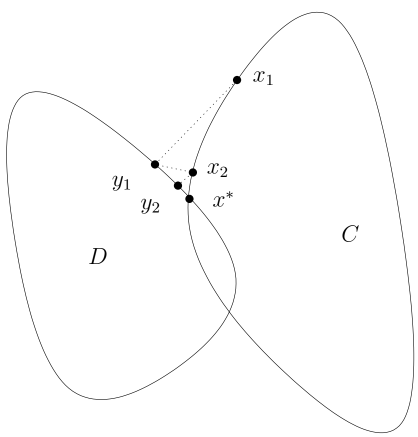
Fuel Blend Type	Minimum Octane Rating	Selling price (\$/barrel)	Demand pattern
1	95	\$45.15	At most 10,000 barrels/day
2	90	\$42.95	Any amount can be sold
3	85	\$40.99	At least 15,000 barrels/day

How can the company maximize its daily profit? Formulate this as a linear programming problem.

- What are the variables in the problem?
- Write out the reward (or cost) function and the constraints.
- Express this linear programming problem in standard form $\min_x c^T x$ such that $Ax = b, x \geq 0$.

Question 3 (20 points)

The *matrix completion problem* has attracted a great deal of interest motivated by real-world challenge problems, such as the Netflix^{TM} competition. Recall from the class lecture that the matrix completion problem involves being given a partially specified matrix M and the goal is to find its completion X such that $X(i,j) = M(i,j)$ when entry i,j is specified in M. We will investigate a special case of this problem called the *positive semidefinite matrix completion problem*, where the goal is to find a positive semidefinite matrix X from a partially specified PSD matrix M (recall that a PSD matrix A is such that $x^T Ax \geq 0$ for all nonzero vectors x, which implies that all the eigenvalues of A are nonnegative and real).



The goal of this programming problem is to solve the PSD matrix completion problem as a *convex feasibility problem* using the technique of *alternating projections* between two sets, one involving the set S_+^n of PSD matrices and the other being the set of all completed matrices consistent with the specified entries of M (see an example of alternating projections shown above for two sets C and D). The algorithm involves alternatively projecting from one set to the other until convergence. To solve this problem, you need to implement these two key steps:

- Given a partially specified symmetric matrix M, specify a projection method that will find the closest point to M in the set of all PSD matrices S_+^n (Hint: use the property that any symmetric matrix has real eigenvalues and can be decomposed as $M = V\Lambda V^T$ using the spectral theorem of linear algebra).

- Given a PSD matrix X, specify a projection method that will find the closest point to X in the set of all completions of the original matrix M.

Implement your program in any language of your choice, and display the convergence of your algorithm using the Frobenius norm of the difference between the current matrix X_k and its projection X_{k+1} on one of the two sets, i.e. $\|X_{k+1} - X_k\|_F$. For projections onto the space S_+^n, this value is the square root of the sum of squares of the negative eigenvalues of X_k. In the case of projecting onto the set of matrix completions of M, it is the square root of the sum of squares of the adjustments made to the fixed entries of X_k.

Question 4 (20 points)

In this programming problem you will implement gradient descent, conjugate gradient, and Newton's method. You will compare their performance on solving random systems of linear equations, $Ax = b$.

Use the attached Matlab function, GenAb, to generate 100 random systems of linear equations with 10 variables and 10 equations. GenAb takes one input, n, which specifies the number of variables and equations in the system of linear equations. You do not have to use Matlab - you may reimplement GenAb in the language of your choice. Or, if you're using a different language and do not want to translate the Matlab code, we have provided 100 A's in A.txt (each 10x10 block of numbers is an A), and 100 b's in b.txt (every 10 numbers form one b).

Next, implement gradient descent, conjugate gradient, and Newtons method. Since the algorithms are all simple to implement, you my **not** use existing packages for the algorithms. You should write them yourselves. If you have access to a package that performs one of the methods, include a comparison of its performance to the version that you wrote (e.g., Matlab has a function *pcg*).

Run each of the algorithms on the 100 sampled problems, each for 50 iterations. Submit a plot with the horizontal axis showing number of iterations of the optimization algorithm, and the vertical axis showing the mean value of $r^\intercal r$, where $r = Ax - b$. Use a logarithmic scale for the vertical axis.

After reading this problem description, but before writing any code, write a paragraph describing what you expect the results to look like. Then, after gathering your results, write a few sentences comparing your hypothesis to the observed results.

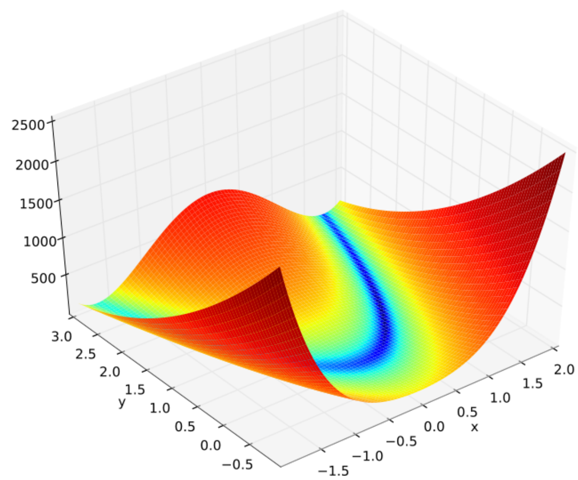
(For your own amusement, try using n=100 equations and unknown variables and see how the plots change. Note that the GenAb function fails for $n >> 1000$).

Question 5 (20 points)

In this programming problem, you will implement gradient descent and Newton's method. You will compare their performance on the Rosenbrock function, which is specified as:

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2$$

For gradient descent, you may use any of the step size schedules described in class (including a constant step size). You should ensure that initial step sizes are selected appropriately. The Rosenbrock function is pictured below[1]



You should provide two plots: the value of the Rosenbrock function squared at the current point versus iteration of the algorithm, and the error $(x - [1, 1])^2$ versus the iteration of the algorithm.[2] Provide plots starting with $x_0 = [0.01, 2.8], x_0 = [0, 3]$, and $x_0 = [0.9, 0.9]$.

Write a brief description (and justification) of your results.

[1]Image taken from Wikipedia.

[2]Notice that [1,1] is the global minimum of the Rosenbrock function.