CSU Math 510

Mini-project 2

Due Thursday, December 8

Overview. Your second of two mini-projects this semester is due on Thursday, December 8. A great way to know if you have learned something is to try to communicate it with others. Therefore, I am asking you to make your mini-project publicly available (typically meaning that it has a url), with the goal of helping teach folks from outside of CSU. The medium for your mini-projects is wide-open: it could be a blog post, a video tutorial, a small webpage explaining a topic, an expository note on a topic, a small report on a research topic, a summary of an academic paper, a computer program you have written using linear programming, an interview with somebody who uses linear programming in their work, or a presentation for a local data science club, etc! The possible topics for your mini-projects are also wide-open: it could be on some aspect of the history of linear programming, an explanation of the solution to an exercise, an explanation of the proof of a theorem, a topic from a lecture, a topic that wasn't covered in any lectures (but that you perhaps wish was), an academic paper you are interested in, or a description of how linear programming relates to your research area, etc. I am excited to see what mini-projects you all come up with and propose!

I want your mini-project to be something that is interesting and beneficial to you! You could, for example:

- Do and explain as cool problem of your choosing.
- Relate the class to your research.
- Learn and explain a side-topic related to linear programming or optimization.
- Code something up and share it.
- Make a blog post.
- Make a YouTube video.
- etc.

Submission. To submit your mini-project, please post it to the discussion board "Mini-project 2" on Canvas. If you have any trouble doing this, or if you prefer not to, then you could also submit this mini-project by emailing it to `henry.adams@colostate.edu`.

What follows is a few "homework" problems from the 2020 version of this class. You do *NOT* need to do or even look at any of these problems. Nevertheless, if you have no ideas for a mini-project, then maybe these problems will help inspire an idea for you.

The blue text below contains html links.

1. Read the 13 page article *Linear Programming: The Story about How it Began* by George Dantzig. Is there any history here that you're interested in sharing, or exploring further?

2. Solve a linear programming problem of your choosing in a software package of your choosing (perhaps python, Matlab, etc).

   There should be built-in commands for doing this.

3. Find and describe a cool example application of linear programming.

4. A set C in $\mathbb{R}^n$ is *convex* if given any two points $x, y \in C$, the entire line-segment connecting x and y is also contained in C. Prove that if C_1 and C_2 are two convex sets in $\mathbb{R}^n$, then their intersection $C_1 \cap C_2$ is also convex.

5. Find the vertices of the polytope in $\mathbb{R}^3$ that is defined by the following system of linear inequalities:

$$\begin{aligned} x_1 + x_2 + x_3 &\leq 4 \\ x_1 &\leq 2 \\ x_3 &\leq 3 \\ 3x_2 + x_3 &\leq 6 \\ x_1 + 2x_2 + 2x_3 &\leq 9 \\ x_1 &\geq 0 \\ x_2 &\geq 0 \\ x_3 &\geq 0 \end{aligned}$$

   Are any of the above linear inequalities redundant, meaning they can be removed without changing the polytope?

   Remark: You have not yet *been taught an algorithm for doing this. But, c'mon, it feels like we might be able to play around and figure it out, right? Exploring in mathematics builds your intuition and leads to good ideas!*

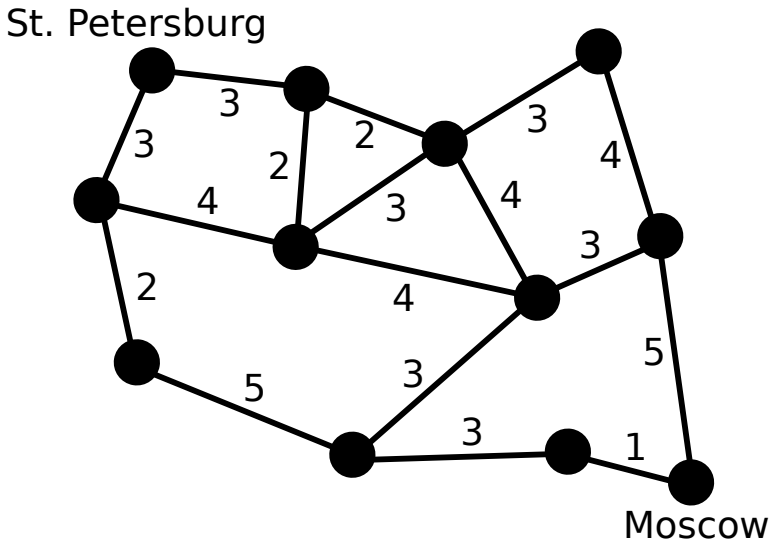
6. Find the half-spaces which define the polytope in $\mathbb{R}^3$ that is defined as the convex hull of the following set of points:

$$\{(0,0,0), (-2,1,-3), (3,-4,2), (1,-2,5), (-1,-3,4), (-2,-2,-2), \\ (3,5,2), (1,2,-2), (2,1,-3), (-1,1,3), (-4,2,-1), (2,-2,-1)\}.$$

Are any of the points above redundant, meaning they can be removed without changing the polytope?

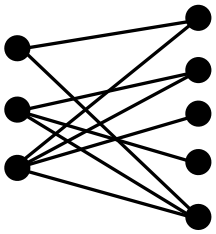
Remark: You have not yet *been taught an algorithm for doing this. But, c'mon, it feels like we might be able to play around and figure it out, right? Exploring in mathematics builds your intuition and leads to good ideas!*

7. Watch a YouTube video on the Ford-Fulkerson algorithm for finding the maximum flow in a network, and then use this algorithm to find the maximum flow from St. Petersburg to Moscow in the below network.



8. Skim Spielman and Teng's article *Smoothed Analysis of Algorithms: Why the Simplex Algorithm Usually Takes Polynomial Time*, available at https://www.di.ens.fr/~vergnaud/algo0910/Simplex.pdf, and explain what you learned.

9. Solve a linear programming problem to find a maximum matching or a minimum vertex cover in the following bipartite graph.



If you do both, then check — does König's Theorem hold on this example?

10. Read parts of Dey, Hirani, and Krishnamoorthy's article *Optimal homologous cycles, total unimodularity, and linear programming* and briefly explain what you learned. I suggest starting with the pictures in Figure 3 :).

11. Use the simplex method to find an optimal solution to the following linear program. If the problem is unfeasible or if the optimization function is unbounded from above, then use the simplex method to deduce and show this.

Maximize $x_1 + 2x_2 - x_3$
subject to

$$\begin{array}{llll} -4x_1 & +x_2 & +x_3 & = 10 \\ x_1 & & +x_3 & = 3 \\ & & x_1, x_2, x_3 & \geq 0 \end{array}$$

12. Put the following linear program into equational form (maximize $c^T x$ subject to $Ax = b$ and $x \geq 0$). Note that x_3 is not currently constrained to be nonnegative.

Minimize x_3
subject to

$$\begin{array}{llllll} -2x_1 & +x_2 & +2x_3 & & & \leq 1 \\ x_1 & & -x_3 & +2x_4 & & = 2 \\ x_1 & +x_2 & & +x_4 & & \geq 3 \\ & & & x_1, x_2, x_4 & & \geq 0 \end{array}$$

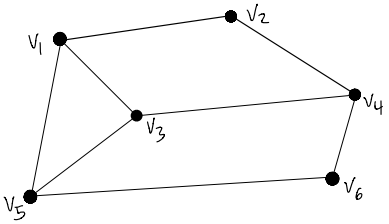
13. Use the simplex method to find an optimal solution to the following linear program. You may choose pivot variables however you like.

Maximize $x_1 + x_2 + x_3 + x_4$
subject to

$$\begin{array}{llllll} -4x_1 & +x_2 & +x_3 & & & = 10 \\ x_1 & & & +x_4 & & = 3 \\ x_1 & +x_2 & & & & \geq -1 \\ & & & x_1, x_2, x_3, x_4 & & \geq 0 \end{array}$$

14. A *maximum independent set* in a graph is a collection of vertices in the graph, no two of which are adjacent, such that the number of vertices is as large as possible.

    (a) Write down, but do not solve, an integer linear programming problem whose solution gives a maximum independent set for the following graph. Briefly explain the notation for the variables you use.



    (b) The *linear relaxation* of the above problem is obtained by dropping the integrality constraint. Explain why for general graphs, the linear relaxation may give an arbitrarily bad approximation for the size of a maximum independent set. One

way to do this is by describing a family of graphs where the linear relaxation approximation can be arbitrarily bad.

15. Argue why finding an optimal solution to a linear programming problem is in some sense no harder than finding any feasible solution. In this class we will see three such possible arguments, one of which involved a binary search, and a second of which involved the duality of linear programming.

16. Explain why under many different choices of pivot rule, it is plausible that the simplex algorithm has exponential running time in the worst case. (This is known to be true for some choices of pivot rule, and suspected to be true for most other pivot rules.) I am not asking for a proof, but instead a plausibility argument.

17. Prove that if a linear programming problem admits an optimal solution, then it also admits an optimal solution at one of the vertices of the polytope defining the feasible region.

18. Do a mini-project related to the traveling salesperson problem.

19. Do a mini-project related to convex optimization.

20. Do a mini-project related to support vector machines.

21. Do a mini-project related to the sparsity-promoting L^1 norm.

22. Do a mini-project related to polytopes, maximizing the number of a faces in a polytope, cyclic polytopes, etc.

23. Show that if you do not design your pivot rules carefully, then it is possible for the simplex algorithm to cycle.

24. Explain the relationship between chromatic numbers of graphs, fractional chromatic numbers, integer linear programs, and relaxations of integer linear programs: https://en.wikipedia.org/wiki/Fractional_coloring#Linear_Programming_(LP)_Formulation.