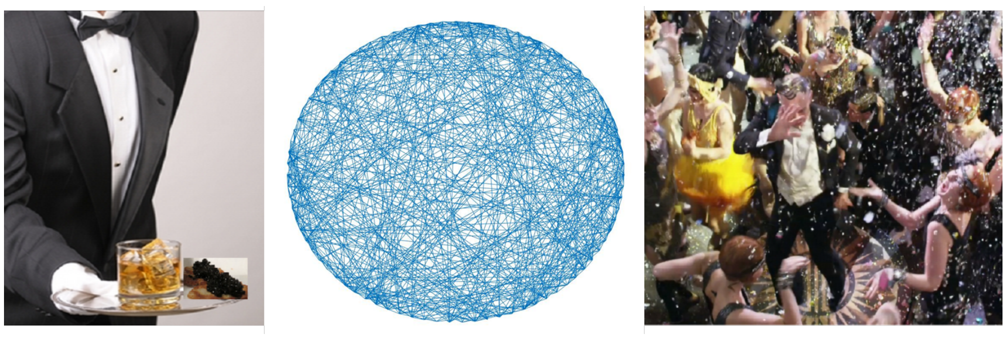
ORF 363/COS 323	Problem set 8	Fall 2023, Princeton University
Instructor: A.A. Ahmadi		
TAs: Beneventano, Chaudhry, Hua, Li, Lok		
Due on Wednesday, December 13, 2023, at 1:00pm EST, on Gradescope		

For all problems that involve coding, please include your code.

Problem 1: End-of-semester party at AAA's

The semester is coming to an end and Amirali wants to invite his wonderful ORF 363 students to his place for a party. Knowing that the students have a refined taste, he wants to offer caviar and single malt scotch to all the guests. There is just a small problem: Amirali's place is fairly small and his salary, being a Professor, is modest at best. So he decides to come up with a lame excuse:
"All students are welcome to the party. But because I want the guests to have a great time, everyone who attends needs to have known everyone else for at least two years."



In the figure above, you can see a graph with ORF 363 students as nodes and with an edge between two nodes if and only if the two students have known each other for at least two years. (This information has been provided to Amirali by the members of ORF 363 Secret Service—Abraar, Jackie, Pier, Silu, and Yixuan—who in return have gotten themselves invited to the party.) In preparation for the party, our goal in this problem is to figure out the maximum number of students N who can possibly show up. The adjacency matrix A of the graph is provided to you in the file `Party_people_in_the_house_tonight.mat`. In MATLAB, you can load the file using the `load` command. In Python, you can use the following code.

```
from scipy.io import loadmat
```

```
A = loadmat('Party_people_in_the_house_tonight.mat')['A']
```

(a) In lecture, we saw an LP relaxation for the STABLE SET problem. Show how this can be used to find an upper bound on N. Solve the LP in CVX or CVXPY and report your upper bound.

(b) We also saw an SDP relaxation for the STABLE SET problem (due to Lovász). Show how this can be used to find an upper bound on N. Solve the SDP in CVX or CVXPY and report your upper bound. Is the LP bound tight? Is the SDP bound tight? (A tight bound means an upper bound that equals N. You may want to answer the next part first.)

(c) Determine a set of N people who can show up to the party by reporting the index of their nodes in the graph. (Hint: look at the eigenvector associated with the largest eigenvalue of the optimal SDP matrix. Guess the indices and verify your guess.)

(d) Consider a decision problem of interest to certain types of hosts:
CHEAPHOST: Given a positive integer k and an undirected graph on n nodes with edges representing friendships among individuals, determine if there are at least k individuals who are all friends with each other.
Show that CHEAPHOST is NP-complete. (Hint: You are allowed to use the fact that STABLE SET is NP-hard.)

Problem 2: Doodle & appointments for academic advising

ORFE has recently asked Amirali to serve as the academic advisor of 19 undergraduates, helping them with course selection for the Spring semester (true story). Amirali is setting aside 5 hours for this task and wants to schedule 15-minute meetings with each student. He has sent out a Doodle (`http://doodle.com/`) to the 19 students and asked them to specify the time slots in which they are available for a meeting. The result is the Doodle sheet that appears on the last page of the problem set. (We have slightly modified the names of the students for privacy reasons.) Your task in this problem is to use linear programming (LP) to decide if it is possible to assign non-overlapping meetings to each student, and if so produce such a schedule.

(a) Recall the maximum flow problem from Lecture 1 (the problem about shipping oil from a source to a target). Reduce the scheduling problem described above to a maximum flow problem. Carefully argue how your reduction results in the correct yes/no answer to the decision part of the scheduling question. (Hint: you are allowed to use the following fact: In a maximum flow problem, if the edge costs are integer, then there exists an optimal integer flow.)

(b) Let's define a decision problem called DOODLE: Given an $m \times n$ zero/one matrix A whose (i, j)-th entry is one if and only if the i-th person is available for a meeting in the j-th time slot, decide if all individuals can get their own meeting time. Is DOODLE in P?

(c) Solve the instance of DOODLE given on the last page of this pset by formulating an LP and solving it in CVX or CVXPY. If the answer to this problem instance is yes, produce the schedule of meetings for all 19 students. You can mark this schedule with a pen on the Doodle sheet provided. For your convenience, the accompanying file `Doodle_matrix.mat` includes the zero-one matrix A encoding this problem instance.

Remark. While you are welcome to take up the challenge of finding a meeting schedule by hand, we want you to solve this problem by solving an LP. Why? Well, suppose instead of 19 meetings your task was to schedule 1000. Your paper-and-pencil strategy would likely fail miserably, while the LP could still be solved in a matter of seconds.

Problem 3: SDP relaxations for nonconvex optimization

Consider the following optimization problem:

$$\begin{aligned} \min_{x_1,x_2}. \quad & x_1^4 - 2x_1^3x_2 + x_1^2x_2^2 - 5x_1x_2^3 - x_2^4 \\ \text{s.t.} \quad & x_1^2 + x_2^2 = 1. \end{aligned} \tag{1}$$

(a) Plot the objective function. Is it a convex function? Prove your answer either way.

(b) Is the feasible set convex? Prove your answer either way.

(c) Generate 200 points uniformly at random from the feasible set. Evaluate the objective function at each point and report the smallest value you find. (Hint: consider using the function `randn` in MATLAB or `np.random.randn` in Python.)

(d) To understand how far your best current solution is from being optimal, we need a *lower bound* on (1). Denote the objective function of (1) by $p(x) := p(x_1, x_2)$. Show that if a real number γ satisfies the following relation

$$p(x) - \gamma(x_1^2 + x_2^2)^2 = z^T(x)Qz(x), \text{ (for all } x)$$

where $z(x) = (x_1^2, x_1x_2, x_2^2)^T$, and Q is any positive semidefinite matrix, then γ is a lower bound on (1).

(e) To obtain the largest such lower bound, solve the following SDP using CVX or CVXPY:

$$\begin{aligned} \max_{\gamma,Q}. \quad & \gamma \\ \text{s.t.} \quad & p(x) - \gamma(x_1^2 + x_2^2)^2 = z^T(x)Qz(x), \\ & Q \succeq 0. \end{aligned}$$

What is the optimal value? How does it compare to your best upper bound on (1)?

Remark. While you are not asked to prove this, the optimal value of the SDP in fact exactly equals the optimal value of (1). Moreover, there is a way for you to extract an optimal solution (x_1^*, x_2^*) from the eigenvectors of the optimal Gram matrix Q^*. This gives you an upper bound that exactly matches the SDP lower bound.

	Nov 16 MON 3:00 PM 3:15 PM	Nov 16 MON 3:15 PM 3:30 PM	Nov 16 MON 3:30 PM 3:45 PM	Nov 16 MON 3:45 PM 4:00 PM	Nov 17 TUE 3:00 PM 3:15 PM	Nov 17 TUE 3:15 PM 3:30 PM	Nov 17 TUE 3:30 PM 3:45 PM	Nov 17 TUE 3:45 PM 4:00 PM	Nov 18 WED 3:00 PM 3:15 PM	Nov 18 WED 3:15 PM 3:30 PM	Nov 18 WED 3:30 PM 3:45 PM	Nov 18 WED 3:45 PM 4:00 PM	Nov 19 THU 3:00 PM 3:15 PM	Nov 19 THU 3:15 PM 3:30 PM	Nov 19 THU 3:30 PM 3:45 PM	Nov 19 THU 3:45 PM 4:00 PM	Nov 20 FRI 3:00 PM 3:15 PM	Nov 20 FRI 3:15 PM 3:30 PM	Nov 20 FRI 3:30 PM 3:45 PM	Nov 20 FRI 3:45 PM 4:00 PM
19 participants +	✓3	✓3	✓2	✓2	✓2	✓1	✓1	✓3	✓1	✓3	✓2	✓3	✓2	✓1	✓1	✓1	✓3	✓0	✓2	✓1
Woodrow Wi...	✓	✓															✓			
F. Scott Fitzgerald						✓				✓										
Richard Feynman															✓					
Michelle Obama		✓										✓								
Paul Volcker								✓		✓										
John Nash				✓															✓	
Terence Tao													✓							
Ben Bernanke					✓					✓										
Paul Krugman							✓				✓									
Andrew Wiles	✓																			
Steve Forbes					✓			✓			✓									
John Milnor	✓													✓						
Kazuyo Sejima									✓											
Albert Einstein			✓										✓							
George A. Miller			✓													✓				
Alan Turing		✓		✓																
Meg Whitman												✓								
Donald Rumsfeld												✓					✓			✓
Eugene O'Neil								✓									✓		✓	