Name:_____

PRINCETON UNIVERSITY

ORF 363/COS 323
Final Exam, Fall 2018

JANUARY 16, 2019

Instructor:
A.A. Ahmadi

AIs:
Dibek, Duan, Gong, Khadir,
Mirabelli, Pumir, Tang, Yu, Zhang

1. Please write out and sign the following pledge on top of the first page of your exam:
   "I pledge my honor that I have not violated the Honor Code or the rules specified by the instructor during this examination. I have not spent more than 48 hours total on this exam."
2. Don't forget to write your name on the exam. Make a copy of your solutions and keep it.
3. The exam is not to be discussed with *anyone* except possibly the professor and the TAs. You can only ask *clarification questions*, and only as *public* (and preferably non-anonymous) questions on Piazza. No emails.
4. You are allowed to consult the lecture notes, your own notes, the reference books of the course as indicated on the syllabus, the problem sets and their solutions (yours and ours), the midterm and its solutions (yours and ours), the practice midterm and final exams and their solutions, all Piazza posts, but *nothing else*. You can only use the Internet in case you run into problems related to MATLAB or CVX.
5. You are allowed to refer to facts proven in the notes or problem sets without reproving them.
6. For all problems involving MATLAB or CVX, show your code. The MATLAB output that you present should come from your code.
7. Unless you have been granted an extension because of overlapping finals, the exam is to be turned in on Friday (January 18, 2018) at 10 AM in the instructor's office (Sherrerd 329). If you cannot make it on Friday and decide to turn in your exam sooner, or if your deadline is different under the rules of the exam, you have to drop your exam off in the ORF 363 box of the ORFE undergraduate lounge (Sherrerd 123). If you do that, you need to *write down the date and time on the first page of your exam and sign it.* You can also submit the exam electronically on Blackboard as a single PDF file.
8. Good luck!

Grading

Problem 1	20 pts	
Problem 2	20 pts	
Problem 3	20 pts	
Problem 4	20 pts	
Problem 5	20 pts	
TOTAL	100	

Problem 1: Deciphering a secret

Amirali has a secret message to convey to his ORF 363 students during the final exam. To protect its secrecy against students outside of class, he decides to first write the message in binary, as a vector $x^{secret} \in \{0,1\}^{360}$. He then encrypts this binary message by generating a random matrix $A \in \mathbb{R}^{200\times 360}$ (using the command `A=randn(200,360)` in MATLAB) and then computing a vector $y \in \mathbb{R}^{200}$ as $y = Ax^{secret}$. (Note that y here is much smaller in length than x so a priori one would think that y has lost part of the information contained in x.) Amirali then shares A and y in the file `encrypted_secret.mat`, hoping that his students would be able to recover the original message x^{secret} using their new-found knowledge but others would not.

1. To recover x^{secret}, consider the optimization problem

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & \sum_{i=1}^{n}(x_i - x_i^2) \\ \text{s.t.} \quad & Ax = y \\ & 0 \le x \le 1. \end{aligned} \tag{1}$$

   What is the optimal value of (1)? Justify. Is (1) a convex optimization problem? Justify.

2. Because problem (1) seems difficult to solve, we replace the objective function with its first-order Taylor approximation at the origin, ending up with the problem

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & \sum_{i=1}^{n} x_i \\ \text{s.t.} \quad & Ax = y \\ & 0 \le x \le 1. \end{aligned} \tag{2}$$

   Is (2) a convex optimization problem? Does your optimal solution to (2) allow you to recover an optimal solution to (1)?

3. Using a binary-to-text converter,[1] tell us the secret message in text.

[1] Available e.g. at `https://codebeautify.org/binary-to-text`.

Problem 2: Convex optimization applied to non-convex problems

Consider the optimization problem

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \le 0,\ i = 1,\dots,m, \end{aligned} \tag{3}$$

whose feasible set is non-empty and compact. Suppose for $i = 0,\dots,m$, the functions $f_i : \mathbb{R}^n \to \mathbb{R}$ can be written as $f_i(x) = g_i(x) - h_i(x)$, where $g_i : \mathbb{R}^n \to \mathbb{R}$ and $h_i : \mathbb{R}^n \to \mathbb{R}$ are all convex functions, h_i's are all differentiable, and g_0 is a strictly convex function. Consider the following algorithm for approximately solving (3):

Algorithm 1

Input: the functions $g_i(x), h_i(x)$ for $i = 0,\dots,m$, a vector $x_0 \in \mathbb{R}^n$ which is feasible to (3), a positive integer N.

Output: a vector $x_N \in \mathbb{R}^n$.

procedure
  $k \leftarrow 0$
  while $k < N$ **do**
    Let $f_i^k(x) := g_i(x) - \big(h_i(x_k) + \nabla h_i(x_k)^T(x - x_k)\big)$, $i = 0,\dots,m$
    Solve the optimization problem: $\min_{x\in\mathbb{R}^n} f_0^k(x)$, s.t. $f_i^k(x) \le 0$, $i = 1,\dots,m$
    Let x_{k+1} denote its optimal solution
    $k \leftarrow k + 1$
  end while

1. Show that the optimization problem solved in each iteration of Algorithm 1 is a convex optimization problem and has a unique optimal solution.

2. *Preserving feasibility.* Show that the points $x_1,\dots,x_N$ generated by Algorithm 1 are all feasible to (3).

3. *The descent property.* Show that the points $x_1,\dots,x_N$ generated by Algorithm 1 satisfy $f_0(x_{k+1}) \le f_0(x_k)$ for $k = 0,\dots,N-1$.

4. Show that any quadratic function $f(x) = x^TQx + c^Tx + b$ can be written as $f(x) = g(x) - h(x)$, where g and h are strictly convex functions. (Hence, at least when $f_0,\dots,f_m$ in (3) are all quadratic functions, Algorithm 1 is applicable.)

Problem 3: How bad can one iteration of gradient descent be for convex optimization?

We saw in lecture that the negative of the gradient is always a descent direction for a differentiable function, in fact the direction of steepest descent. However, if the step size of a descent algorithm is chosen too large, moving along this direction can potentially increase the function. In this problem, we would like to see how bad this can get on a family of convex polynomials.

Suppose our goal is to apply the gardient descent algorithm to minimize a univariate degree-4 polynomial

$$p(x) = c_4x^4 + c_3x^3 + c_2x^2 + c_1x,$$

which is known to satisfy the following three constraints:

1. p is a convex function,
2. $p'(2) = 1$,
3. $|c_i| \leq 10$ for $i = 1, \ldots, 4$.

Suppose we apply one iteration of gradient descent starting from $x_0 = 2$ and with a step size of one. Observe that with this initial point, the next iterate will be $x_1 = x_0 - p'(2) = 1$. What is the largest value of $p(x_1) - p(x_0)$, as p varies over all degree-4 polynomials that satisfy the three properties above? (You can write down the value that CVX returns with three digits after the decimal point.)

Write down a degree-4 polynomial with the above properties that leads to worst-case performance for one step of the gradient descent algorithm with unit step size.

Hint: You can use the fact that a quadratic polynomial is nonnegative if and only if it is a sum of squares (no need to prove this fact).

Problem 4: Solving easy linear programs quickly

Let $c, y_1, \ldots, y_m \in \mathbb{R}^n$ be given. Describe an algorithm for finding the optimal value of the linear program

$$\begin{aligned} \min_{x \in \mathbb{R}^n,\, \lambda \in \mathbb{R}^m} \quad & c^T x \\ \text{s.t.} \quad & x = \sum_{i=1}^m \lambda_i y_i \\ & \sum_{i=1}^m \lambda_i = 1 \\ & \lambda_i \geq 0,\ i = 1, \ldots, m, \end{aligned} \tag{4}$$

which involves at most mn scalar-scalar multiplications, mn scalar-scalar additions, and m scalar-scalar comparisons. (Here, the comparison of two scalars means to decide which one is less than or equal to the other.) Carefully justify why your algorithm gives the correct optimal value.

Problem 5: When does a population go extinct?

Consider the linear dynamical system

$$\begin{pmatrix} x_{k+1}^1 \\ x_{k+1}^2 \\ x_{k+1}^3 \\ \vdots \\ x_{k+1}^n \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & 0 & 0 & \dots & 0 \\ 0 & a_{32} & 0 & \dots & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \dots & \dots & a_{n,n-1} & 0 \end{pmatrix} \begin{pmatrix} x_k^1 \\ x_k^2 \\ x_k^3 \\ \vdots \\ x_k^n \end{pmatrix},$$

which models the growth dynamic of a population. In this model, the variable x_k^i denotes the number of individuals in age group i (i.e., those who are $i-1$ to i years old), in year k of the history of the population. We have $i \in \{1, \dots, n\}$ as we are assuming that no individual stays alive for more than n years. The index k of time, however, runs forever. Note that the structure of the square matrix governing the dynamics (let us call it A) is quite intuitive: at each time stage, only a fraction $a_{(i+1)i}$ of people in age group i make it to the age group $i+1$. At the same time, each age group i contributes a fraction a_{1i} to the newborns in the next stage.

Consider an instance of this dynamical system given by

$$A = \begin{pmatrix} 0.1 & 0.2 & 0.3 & 0.4 & a_{15} \\ 0.9 & 0 & 0 & 0 & 0 \\ 0 & 0.8 & 0 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0 \\ 0 & 0 & 0 & 0.7 & 0 \end{pmatrix}.$$

What is the largest value of a_{15} for which the population will eventually go extinct, regardless of the initial population that the dynamical system starts from? Give this cutoff value to two digits after the decimal point.

Hint 1: You can use the following mathematical fact (no need to prove it): The population will go extinct if and only if the linear dynamical system admits a Lyapunov function of the type $V(x) = x^T Q x$, where Q is a symmetric 5×5 matrix. Try different values of a_{15} and use semidefintie programming to see if such a Lyapunov function exists.

Hint 2: You may want to use the following inequality (no need to prove it) to show certain properties of your Lyapunov function: $x^T Q x \geq \lambda_{min}(Q)||x||^2, \forall x \in \mathbb{R}^5$. Here, $\lambda_{min}(Q)$ denotes the smallest eigenvalue of Q.

Hint 3: To have a valid semidefinite programming formulation, if you ever need to impose a constraint of the type $E \succ 0$ for some matrix E, instead impose the constraint $E \succeq I$ and argue why this does not affect feasibility.