Introduction to Linear Dynamical Systems EE263, Summer 2019
August 16-17 2019

Final Exam

This is a 12 hour take-home exam with 3 problems. Please turn it in on Gradescope 12 hours after it is emailed to you.

- You may use any books, notes, or computer programs (*e.g.*, matlab), but you may not discuss the exam with others until August 17, after everyone has taken the exam. The only exception is that you can ask the course staff for clarification, by emailing to the staff email address `{gorish, lucasfv, amomenis} @stanford.edu`. We've tried pretty hard to make the exam unambiguous and clear, so we're unlikely to say much. Please do not post any exam related questions on Piazza.
- Since you have 12 hours, we expect your solutions to be legible, neat, and clear. Do not hand in your rough notes, and please try to simplify your solutions as much as you can. We will deduct points from solutions that are technically correct, but much more complicated than they need to be.
- Please check your email and canvas a few times during the exam, just in case we need to send out a clarification or other announcement. It's unlikely we'll need to do this, but you never know.
- Assemble your solutions to the problems in order, *i.e.*, problem 1, problem 2, problem 3. Start each part of the problems on a new page.
- Please make a copy of your exam before handing it in. We have never lost one, but it might occur.
- If a problem asks for some specific answers, make sure they are obvious in your solutions. You might put a box around the answers, so they stand out from the surrounding discussion, justification, plots, etc.
- When a problem involves some computation (say, using matlab), we do not want just the final answers. We want a clear discussion and justification of exactly what you did, the matlab source code that produces the result, and the final numerical result. Be sure to show us your verification that your computed solution satisfies whatever properties it is supposed to, at least up to numerical precision. For example, if you compute a vector x that is supposed to satisfy $Ax = b$ (say), show us the matlab code that checks this, and the result. (This might be done by the matlab code `norm(A*x-b)`; be sure to show us the result, which should be very small.) *We will not check your numerical solutions for you, in cases where there is more than one solution.*

- In the portion of your solutions where you explain the mathematical approach, you *cannot* refer to matlab operators, such as the backslash operator. (You can, of course, refer to inverses of matrices, or any other standard mathematical construct.)
- Some of the problems are described in a practical setting, such as Robotics or Mechanical systems. *You do not need to understand anything about the application area to solve these problems.* We've taken special care to make sure all the information and math needed to solve the problem is given in the problem description.
- The zip file for the datasets required for the exam have been emailed to you alongside the exam pdf.
- Please respect the honor code. Although we encourage you to work on homework assignments in small groups, *you cannot discuss the final exam with anyone*, with the exception of EE263 course staff, until August 17 when everyone has taken it and the solutions are posted online.
- Finally, a few hints:
  - Problems may be easier (or harder) than they might at first appear.
  - None of the problems require long calculations or any serious programming.

1. *Bow Tie Launch [40 marks].*
For the launch of its new product, *Bow Tie Intl* is planning on organizing a drone show[1]. They hired **you** as their chief controls engineer to design the choreography of the fleet. They want you to design a controls sequence that would enable this fleet to draw a sketch of their latest model.

To simplify the problem, we consider here one drone only. It is equipped with thrusters exerting forces in the **x** and **z** directions with 100% efficiency. We will assume here there is no perturbation in the **y** direction. The discretized equations of motion for the drone are

$$x(t+1) = \underbrace{\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{A} x(t) + \underbrace{\begin{bmatrix} \frac{1}{2} & 0 \\ 1 & 0 \\ 0 & \frac{1}{2} \\ 0 & 1 \end{bmatrix}}_{B} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \frac{1}{2} f_g \\ f_g \end{bmatrix}}_{B_g},$$

where x_1 and x_2 are the position and velocity in the **x**-direction, and x_3, x_4 are the position and velocity in the **z**-direction. Here

$$u(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}$$

is the force acting on the drone for time in the interval $[t, t+1)$, while f_g is the gravitational force. For this exercise, and for the sake of simplicity, we will take $f_g = -1$. Let the position of the vehicle at time t be $q(t) \in \mathbb{R}^2$.

(a) (10 marks) In order to draw a nice bow tie in the air, you first specify a few points p that the drone has to fly towards at 11 given time instants T, given below:

```
p=[2 2 0 -2 -2 0 0 0 0 0 0; 1/2 -2 0 1/2 -2 0 0 0 0 0 0];
T=[10 20 30 40 50 60 61 62 63 64 65];
```

In this case, the drone starts at the origin. You require the drone to fly through different points at 10 seconds intervals, and then require it to stabilize at the origin for 5 seconds. We will therefore pilot the drone for 65 seconds. We would like to apply a sequence of inputs $u(0), u(1), \ldots, u(64)$ to make the drone go through the aforementioned positions at specified times.

In this first attempt, you look for the sequence that minimizes the control effort, *i.e.* that minimizes

$$\sum_{t=0}^{64} \|u(t)\|^2$$

[1]See https://www.youtube.com/watch?v=KhDEEN4gcpI for an example

To do this, pick $\hat{A}$ and $\hat{y}$ to set this problem as an equivalent minimum-norm problem, where we would like to find the minimum-norm u_{seq} which satisfies

$$\hat{A}u_{\text{seq}} = \hat{y}$$

where u_{seq} is the sequence of force inputs

$$u_{\text{seq}} = \begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(64) \end{bmatrix}$$

Plot the trajectory of the drone using this input, and the way-points p. Also plot the optimal u against time.

(b) (6 marks) Now we would like to compute the trade-off curve between the accuracy with which the mass passes through the waypoints and the norm of the force used. Let our two objective functions be

$$J_1 = \sum_{i=1}^{11} \|q(t_i) - p_i\|^2 ,$$

where $q(t_i)$ notes the x and z direction positions of the drone at prescribed times t_i and p_i the goal positions, and

$$J_2 = \sum_{t=0}^{64} \|u(t)\|^2$$

By minimizing the weighted sum

$$J_1 + \mu J_2$$

for a range of values of μ, plot the trade-off curve of J_1 against J_2 showing the achievable performance. To generate suitable values of μ, you may find the `logspace` command useful in Matlab; you'll need to pick appropriate maximum and minimum values.

(c) (5 marks) Plot up to 5 drone trajectories for different values of μ. Make sure those trajectories explore the a range of different behaviors. Explain the influence of μ on the resulting trajectories.

(d) (5 marks) After showing the results of the first simulations to your boss, you are told that the sketch should be skinnier. The wings of the bow tie are indeed much wider than the actual product. One possibility to solve this problem would be to specify additional way points. However, that would amount to specifying the entire trajectory of the drone and therefore losing flexibility.

Another soultion is to solve a *constrained multiobjective least-squares* problem. We would like to impose the constraint that the drone passes through all the waypoints that is, achieve zero waypoint error $J_1 = 0$, while penalizing the drone when it goes too far out. This will hopefully result in the wings being skinnier. We can attempt to keep the drone close to the origin by trading off the sum of the squares of the *position*

$$J_3 = \sum_{t=0}^{65} \|q(t)\|^2$$

against input cost J_2 subject to this constraint. To do this, we'll solve

$$\begin{array}{ll} \text{minimize} & J_3 + \gamma J_2 \\ \text{subject to} & J_1 = 0 \end{array}$$

First, find the matrices F and G so that the cost function is given by

$$J_3 + \gamma J_2 = \|F u_{\text{seq}} + G\|^2$$

(e) (6 marks) Now we have a problem of the form

$$\begin{array}{ll} \text{minimize} & \|F u_{\text{seq}} + G\|^2 \\ \text{subject to} & \hat{A} u_{\text{seq}} = \hat{y} \end{array}$$

This is called a *weighted minimum-norm solution*; the only difference from the usual minimum-norm solution to $\hat{A} u_{\text{seq}} = \hat{y}$ is the presence of the matrices F and G, and when $F = I$ and $G = 0$ the optimal u is just given by $u_{\text{opt}} = A^{\dagger}\hat{y}$. Show that the solution for general F and G is

$$u_{\text{opt}} = \Sigma^{-1} A^T (A \Sigma^{-1} A^T)^{-1} (\hat{y} + \hat{A} \Sigma^{-1} F^T G) - \Sigma^{-1} F^T G$$

where $\Sigma = F^T F$. (One way to do this is using Lagrange multipliers.) Use this to solve the remaining parts of this problem.

(f) (5 marks) Plot the trajectories for $\gamma = 1, 10, 100, 1000$. How do you explain the behavior at $\gamma = 1$? How do you explain the evolution between the different trajectories? Among those four cases, which one do you think matches best the requirements of your boss?

(g) (3 marks) For a range of values of γ, plot the trade-off curve of J_3 against J_2 showing the achievable performance.

2. *Attack position of Fighter Jets [30 marks].*
   Consider you are commanding a fleet of MiG-17 fighter jets , labeled $1, \dots, n$, which move along a line with (scalar) positions $y_1, \dots, y_n$. We let $v_1, \dots, v_n$ denote the velocities of the jets, and $u_1, \dots, u_n$ the net forces applied to the jets. The motions of these fighter jets are governed by the equations

$$\dot{y}_i = v_i, \qquad \dot{v}_i = u_i - v_i.$$

   (Here we take mass of each jet to be one, and include a damping term in the equations.) We assume that $y_1(0) < \cdots < y_n(0)$, *i.e.*, the jets start out with jet 1 in the leftmost position, followed by the jet 2 to its right, and so on, with jet n in the rightmost position. Your fleet has been called to action to bomb an enemy base. In order to achieve this, you must arrange your vehicles in the following attack configuration after a sufficiently large time:

$$\lim_{t\to\infty} y_i(t) = i, \quad \lim_{t\to\infty} v_i(t) = 0, \quad i = 1, \dots, n,$$

   i.e., first jet at position 1, with unit spacing between adjacent jets, and all stationary. We call this configuration *aligned*, and the goal is to drive the jets to this configuration, *i.e.*, to align the jets. We define the spacing between jet i and $i+1$ as $s_i(t) = y_{i+1}(t) - y_i(t)$, for $i = 1, \dots, n-1$. (When the jets are aligned, these spacings are all one.) Back at the command center ,You have at your disposal the following three control schemes to achieve the given fleet configuration.

   - *Right looking control* is based on the spacing to the jet to the right. We use the control law
$$u_i(t) = s_i(t) - 1, \quad i = 1, \dots, n-1,$$
     for jets $i = 1, \dots, n-1$. In other words, we apply a force on jet i proportional to its spacing error with respect to the jet to the right (*i.e.*, jet $i+1$). The rightmost jet uses the control law
$$u_n(t) = -(y_n(t) - n),$$
     which applies a force proportional to its position error, in the opposite direction. This control law has the advantage that only the rightmost jet needs an absolute measurement sensor; the others only need a measurement of the distance to their righthand neighbor.
   - *Left and right looking control* adjusts the input force based on the spacing errors to the jet to the left and the jet to the right:
$$u_i(t) = \frac{s_i(t) - 1}{2} - \frac{s_{i-1}(t) - 1}{2}, \quad i = 2, \dots, n-1,$$
     The rightmost jet uses the same absolute method as in right looking control, *i.e.*,
$$u_n(t) = -(y_n(t) - n),$$

and the first jet, which has no jet to its left, uses a right looking control scheme,

$$u_1(t) = s_1(t) - 1.$$

This scheme requires jet n to have an absolute position sensor, but the other jets only need to measure the distance to their neighbors.

- *Independent alignment* is based on each jet independently adjusting its position with respect to its required position:

  $$u_i(t) = -(y_i(t) - i), \quad i = 1, \ldots, n.$$

  This scheme requires all jets to have absolute position sensors.

In the questions below, we consider the case where you have $n = 7$ jets in your squadron.

(a) (4*3+3 marks) Which of the three schemes work? By 'work' we mean that the jets converge to the alignment configuration, no matter what the initial positions and velocities are. Among the schemes that do work, which one gives the fastest asymptotic convergence to alignment? (If there is a tie between two or three schemes, say so.) In this part of the problem you can ignore the issue of collisions of the jets, *i.e.*, spacings that pass through zero.
Hint: Try expressing each of the control schemes as a Linear Dynamical System.

(b) (5*3 marks) *Collisions.* In this problem we analyze jet collisions, which occur when any spacing between jets is equal to zero. (For example, $s_3(5.7) = 0$ means that jets 3 and 4 collide at $t = 5.7$.) We take the particular starting configuration

$$y = (0, 1, 2, 4, 6, 7, 8), \qquad v = (0, 0, 0, 0, 0, 0, 0),$$

which corresponds to the jets with zero initial velocity, but not in the aligned positions. For each of the three schemes above (whether or not they work), determine if a collision occurs. If a collision does occur, find the earliest collision, giving the time and the jets involved in that collision. (For example, 'jets 3 and 4 collide at $t = 7.7$.') If there is a tie, *i.e.*, two pairs of jets collide at the same time, say so. If the jets do not collide, find the point of closest approach, *i.e.*, the minimum spacing that occurs, between any pair of jets, for $t \geq 0$. (Give the time, the jets involved, and the minimum spacing.) If there is a tie, *i.e.*, two or more pairs of jets have the same distance of closest approach, say so. Be sure to give us times of collisions or closest approach with an absolute precision of at least 0.1.

3. *UN General Assembly voting [30 marks].*
CNN News network has decided to analyse the voting patterns of countries in the UN General Assembly which comprises of both capitalist and communist nations along with socialist nations which essentially follow a stance between the two hard-core ideologies. As the journalist assigned to this study, you collect the results of votes on the key economic issues raised in the Assembly for the year 2018-19. This data is then compiled by your team into the file *un_voting_patterns.m*, which defines the following variables.

- `countries`, a 102×2 cell array. There are 102 rows because Tanzania exited during its term, and was replaced by Belgium. Also Albania, has been switching its stance through the year, recording both pro-capitalism and socialists votes
  - `countries{i,1}` is the name of the ith country
  - `countries{i,2}` is the economy type of the ith country
- `votes`, a 102×633 matrix where `votes(i,j)` is the vote cast by country i in vote j ($+1$ if country i voted "Yea" on vote j, -1 if country i voted "Nay" on vote j, and 0 if country i did not participate in vote j)

(a) (5 marks) Make a stem plot of the singular values of `votes`. How many significant singular values are there? The fraction of the variation in the voting data associated with the first two singular values is

$$f_2 = \frac{\sigma_1^2 + \sigma_2^2}{\sum_{i=1}^{r} \sigma_i^2}.$$

What is f_2 for the Assembly voting data?

(b) (9 marks) Missing votes will complicate our subsequent analysis, so we use a low-rank model of the voting matrix to guess the missing votes. Initialize A to the observed voting matrix; repeat the following steps until convergence:

- replace A with its best rank-two approximation;
- replace the entries of A corresponding to the known votes with their true values;
- replace the entries of A corresponding to the unknown votes by setting non-negative entries to $+1$, and negative entries to -1.

Report the numbers of $+1$'s and -1's in your final matrix A.

(c) (3*2+2 marks) Let $A = U\Sigma V^T$ be the singular-value decomposition of A. We want to find interpretations for the first two left singular vectors, u_1 and u_2.

i. The function `spatial_plot`, which is available in the data zip file, can be used to generate color-coded scatter plots. Use this function to make a scatter plot of $(u_1)_i$ and $(u_2)_i$ where Capitalists, Communists and Socialists are represented by red, blue, and green markers, respectively. You can make such a plot using the command

```
spatial_plot(u1 , u2 , z , 3 , eye(3));
```

where `u1` and `u2` are the first two left singular vectors of A, and `z` is a vector where `z(i)` is 0 if country i is a Capitalist economy, 1 if country i is a Socialist economy, and 2 if country i is a Communist economy.

ii. Make a scatter plot showing the fraction of votes in which each country voted with the majority of the other countries. You can make such a plot using the command

```
spatial_plot(u1 , u2 , z , 10);
```

where `u1` and `u2` are the first two left singular vectors of A, and `z` is a vector where `z(i)` is the fraction of votes j for which

$$A_{ij} = sgn(\sum_{\tilde{i}=1}^{102} A_{\tilde{i}j}).$$

(By default, `spatial_plot` uses the `cool` color map, which is pink for large values, and blue for small values. Also, sgn refers to the sign function.)

Based on these plots, give intuitive interpretations of $(u_1)_i$ and $(u_2)_i$.

(d) (3*2+2 marks) We can also find interpretations for the first two right singular vectors.

i. Make a scatter plot of $(v_1)_j$ and $(v_2)_j$ where the colors indicate the total support received by vote j. In particular, the color should correspond to

$$z_j = \sum_{i=1}^{102} A_{ij}.$$

ii. Make a scatter plot of $(v_1)_j$ and $(v_2)_j$ where the colors indicate the partisan support received by vote j. In particular, let $Cap \subset \{1, \dots, 102\}$ be the set of Capitalists, and let $Com \subset \{1, \dots, 102\}$ be the set of Communists. The color in your plot should correspond to

$$z_j = \frac{1}{|Cap|} \sum_{i \in Cap} A_{ij} - \frac{1}{|Com|} \sum_{i \in Com} A_{ij}.$$

Based on these plots, give intuitive interpretations of $(v_1)_i$ and $(v_2)_i$.

The data given in the above question is artificially generated and should not be taken as an indicator for the world reality in any way whatsoever.

Congratulations! You've finished EE263! We hope this was a great learning experience for you all and we wish you the best in all of your future endeavors.