EE263 Homework 8 Solutions

Fall 2023

7.1080. Hovercraft with limited range. We have a hovercraft moving in the plane with two thrusters, each pointing through the center of mass, exerting forces in the **x** and **y** directions with 100% efficiency. The hovercraft has mass 1. The discretized equations of motion for the hovercraft are

$$x(t+1) = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} x(t) + \begin{bmatrix} \frac{1}{2} & 0 \\ 1 & 0 \\ 0 & \frac{1}{2} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}$$

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 **y**-direction. Here

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

is the force acting on the hovercraft for time in the interval $[t, t+1)$. Let the position of the vehicle at time t be $q(t) \in \mathbb{R}^2$.

a) The hovercraft starts at the origin. We'd like to apply thrust to make it move through points p_1, p_2, p_3 at times t_1, t_2, t_3, where

$$p_1 = \begin{bmatrix} 1 \\ -\frac{1}{2} \end{bmatrix} \qquad p_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad p_3 = \begin{bmatrix} -\frac{3}{2} \\ 0 \end{bmatrix}$$

$$t_1 = 6 \qquad t_2 = 40 \qquad t_3 = 50$$

We will run the hovercraft on the time interval $[0, 70]$. We'd like to apply a sequence of inputs $u(0), u(1), \ldots, u(70)$ to make the hovercraft position pass through the above sequence of points at the specified times.

We would like to find the sequence of inputs that drives the hovercraft through the desired points which has the minimum cost, given by the sum of the squares of the forces:

$$\sum_{t=0}^{70} \|u(t)\|^2$$

To do this, pick A_{hov} and y_{des} to set this problem up as an equivalent minimum-norm problem, where we would like to find the minimum-norm u_{seq} which satisfies

$$A_{\text{hov}} u_{\text{seq}} = y_{\text{des}}$$

where u_{seq} is the sequence of force inputs

$$u_{\text{seq}} = \begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(70) \end{bmatrix}$$

Plot the trajectory of the hovercraft using this input, and the way-points $p_1, \dots, p_3$. Also plot the optimal u against time.

b) 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

$$J_1 = \sum_{i=1}^{3} \|q(t_i) - p_i\|^2 = \|A_{\text{hov}} u_{\text{seq}} - y_{\text{des}}\|^2$$

and

$$J_2 = \sum_{t=0}^{70} \|u(t)\|^2$$

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. This above trade-off curve shows how we can trade-off between how accurately the hovercraft passes through the waypoints and how much input energy is used.

c) For each of the following values of μ

$$\{\, 10^{\frac{p}{2}} \mid p = -2, 0, 2, \dots, 10 \,\}$$

plot the trajectories all on the same plot, together with the waypoints.

d) Now suppose we are controlling the hovercraft by radio control, and the maximum range possible between the transmitter and receiver is 2 (in whatever units we are using for distance.) Notice that, if we use the minimum-norm input then the hovercraft passes out of range, both when making its first turn and on the final stretch (between times 50 and 70).

We'd like to do something about this, but trading off the input norm as above doesn't do the right thing; if μ is large then the hovercraft stays within range, but misses the waypoints entirely; if μ is small then it comes close to the waypoints, but goes out of range. Notice that this is particularly a problem on the final stretch between times 50 and 70; explain why this is.

e) One remedy for this problem is to solve a *constrained multiobjective least-squares* problem. We would like to impose the constraint that

$$A_{\text{hov}} u_{\text{seq}} = y_{\text{des}}$$

that is, achieve zero waypoint error $J_1 = 0$. We can attempt to keep the hovercraft in range by trading off the sum of the squares of the *position*

$$J_3 = \sum_{t=0}^{70} \|q(t)\|^2$$

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

$$\begin{array}{ll} \text{minimize} & J_3 + \gamma J_2 \\ \text{subject to} & A_{\text{hov}} u_{\text{seq}} = y_{\text{des}} \end{array}$$

First, find the matrix W so that the cost function is given by

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

f) Now we have a problem of the form

$$\begin{array}{ll} \text{minimize} & \|Wu\|^2 \\ \text{subject to} & Au = y_{\text{des}} \end{array}$$

This is called a *weighted minimum-norm solution*; the only difference from the usual minimum-norm solution to $Au = y_{\text{des}}$ is the presence of the matrix W, and when $W = I$ the optimal u is just given by $u_{\text{opt}} = A^{\dagger} y_{\text{des}}$. Show that the solution for general W is

$$u_{\text{opt}} = \Sigma^{-1} A^T (A \Sigma^{-1} A^T)^{-1} y_{\text{des}}$$

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

g) For each of the following values of γ

$$\{ 10^{\frac{p}{2}} \mid p = 0, 2, 4, \ldots, 20 \}$$

Plot the trajectories all on the same plot, together with the waypoints. Explain what you see.

h) By trying different values of γ, you should be able to find a trajectory which just keeps the hovercraft within range. Plot the trajectory of the hovercraft; what is the corresponding value of γ? Is this the smallest-norm input u that just keeps the hovercraft within range, and drives the hovercraft through the waypoints? Explain why, or why not.

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

Solution.

a) Setting

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

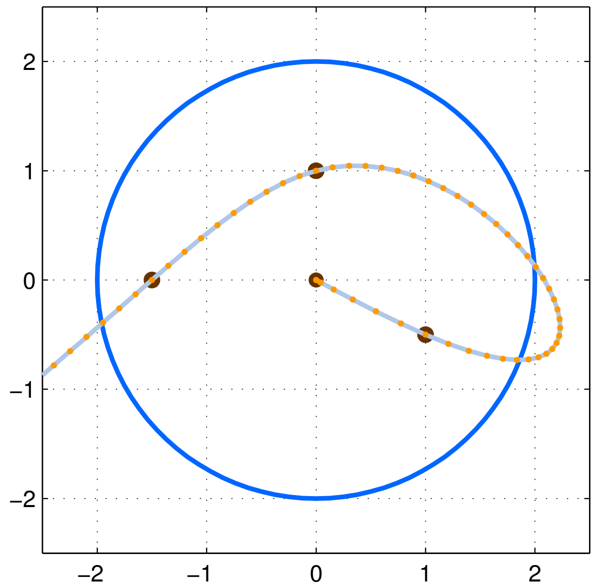
gives the position of the hovercraft at time t as

$$y(t) = \sum_{\tau=0}^{t-1} C A^{t-1-\tau} B u(\tau)$$

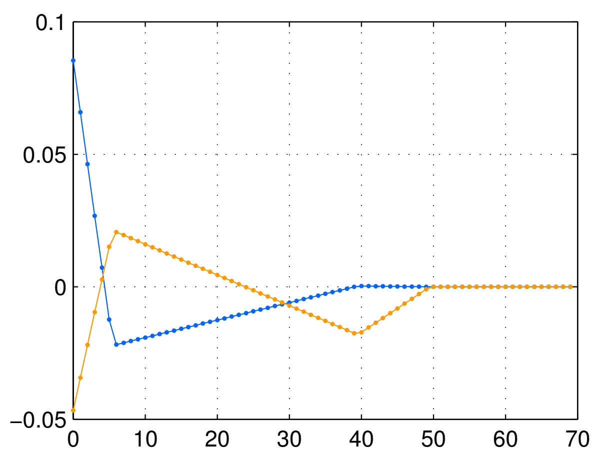
The parameters for the least-squares problem are therefore

$$A_{\text{hov}} = \begin{bmatrix} CA^{t_1-1}B & CA^{t_1-1} & \cdots & CB & 0 & 0 & \ldots & 0 \\ CA^{t_2-1}B & CA^{t_2-2}B & & & \cdots & & & 0 \\ CA^{t_3-1}B & CA^{t_3-2}B & & & \cdots & & & 0 \end{bmatrix} \qquad y_{\text{des}} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

Solving this least squares problem gives optimal trajectory



The corresponding optimal input sequence is below.



b) The weighted sum objective is

$$J_1 + \mu J_2 = \left\| \begin{bmatrix} A_{\text{hov}} \\ \sqrt{\mu} I \end{bmatrix} u_{\text{seq}} - \begin{bmatrix} y_{\text{des}} \\ 0 \end{bmatrix} \right\|^2$$

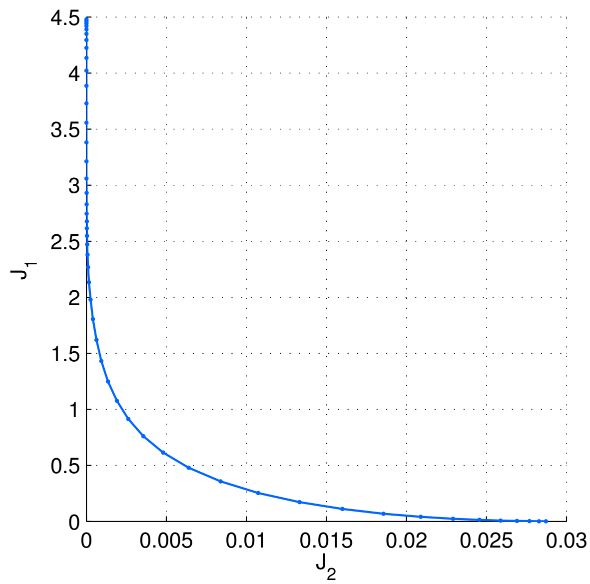
where

$$u_{\text{seq}} = \begin{bmatrix} u(0) \\ \vdots \\ u(69) \end{bmatrix}$$

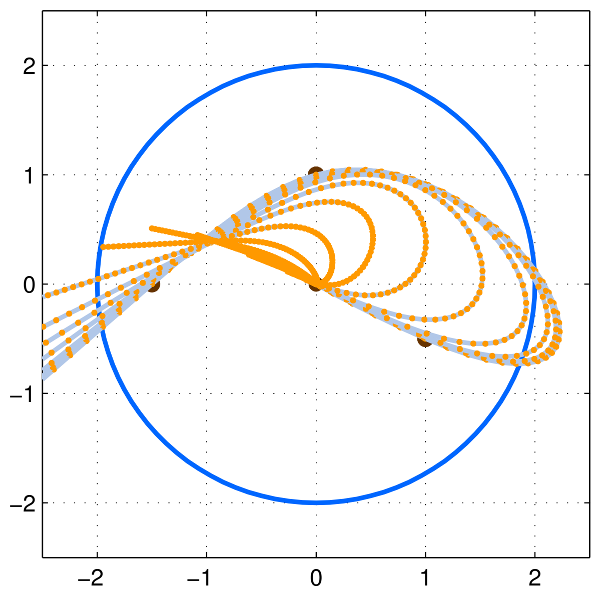
and so the optimal input sequence is given by

$$u_{\text{seq}} = \begin{bmatrix} A_{\text{way}} \\ \sqrt{\mu} I \end{bmatrix}^{\dagger} \begin{bmatrix} y_{\text{des}} \\ 0 \end{bmatrix}$$

Choosing values of μ between 1 and 10^7 using `mus=logspace(0,7,50)`, the trade-off curve is shown below.



c) All of the trajectories together are



We can see clearly that increasing μ reduces the accuracy with which the trajectory passes through the waypoints.

d) On the final stretch the input is zero, and so is unaffected by increasing μ. We were attempting to use the heuristic 'keeping u small keeps x small' but this fails, because when $u = 0$ the hovercraft just keeps going in a straight line.

e) We would like to minimize $J_3 + \gamma J_2$ subject to the constraints that the hovercraft moves through the waypoints. Denote the sequence of positions of the hovercraft by

$$y_{\text{seq}} = \begin{bmatrix} y(0) \\ \vdots \\ y(T) \end{bmatrix}$$

where $T = 70$. Then we have

$$y_{\text{seq}} = T u_{\text{seq}}$$

where T is the Toeplitz matrix

$$T = \begin{bmatrix} 0 & & & \\ CB & 0 & & \\ CAB & CB & 0 & \\ \vdots & & \ddots & \\ CA^{T-1}B & CA^{T-2}B & \dots & CB \end{bmatrix}$$

Now the cost function is

$$\begin{aligned} J_3 + \gamma J_2 &= \|T u_{\text{seq}}\|^2 + \gamma \|u_{\text{seq}}\|^2 \\ &= \|W u_{\text{seq}}\|^2 \end{aligned}$$

where

$$W = \begin{bmatrix} T \\ \sqrt{\gamma} I \end{bmatrix}$$

f) We'd like to solve

$$\begin{array}{ll} \text{minimize} & \|Wu\|^2 \\ \text{subject to} & Au = y_{\text{des}} \end{array}$$

One way to solve this is using Lagrange multipliers; if we augment the cost function by the Lagrange multipliers multiplied by the constraints, we have

$$L(u, \lambda) = u^T \Sigma u + \lambda^T (Au - y_{\text{des}})$$

and the optimality conditions are

$$\frac{\partial L}{\partial u} = 2u_{\text{opt}}^T \Sigma + \lambda^T A = 0$$

$$\frac{\partial L}{\partial \lambda} = u_{\text{opt}}^T A^T - y_{\text{des}}^T = 0$$

The first condition gives

$$u_{\text{opt}} = -\frac{1}{2} \Sigma^{-1} A^T \lambda$$

and substituting this into the second we have

$$-\frac{1}{2} A \Sigma^{-1} A^T \lambda = y_{\text{des}}$$

hence

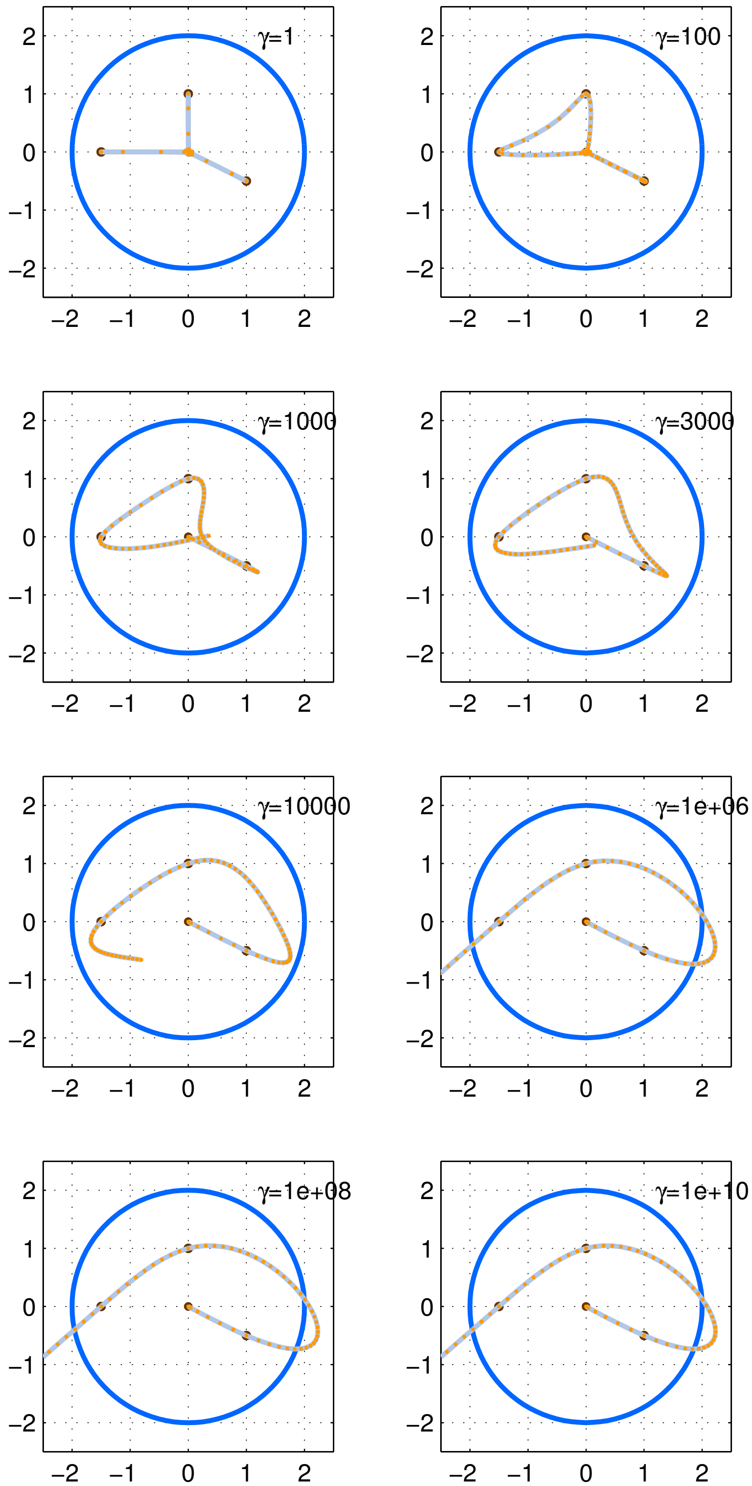
$$\lambda = -2(A \Sigma^{-1} A^T)^{-1} y_{\text{des}}$$

and

$$u_{\text{opt}} = \Sigma^{-1} A^T (A \Sigma^{-1} A^T)^{-1} y_{\text{des}}$$

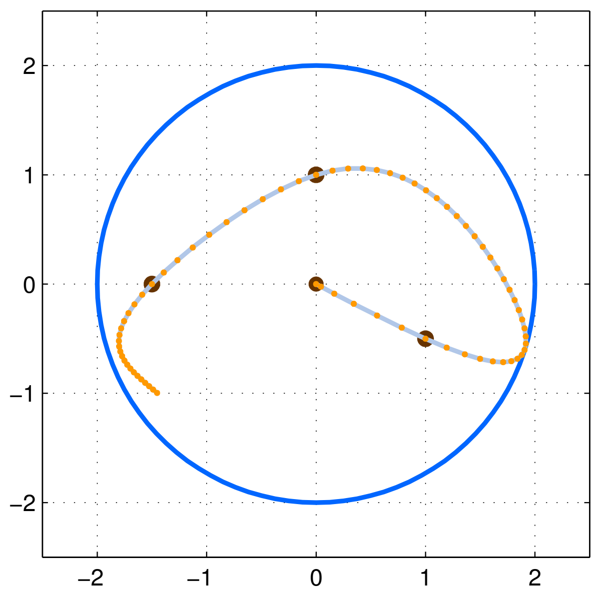
as desired.

g) The trajectory for a range of γ values is shown below. (Actually these are clearer on separate plots)



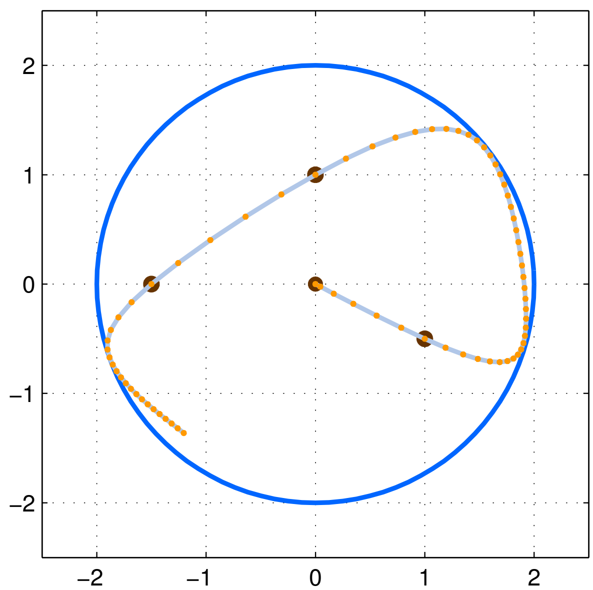
We can see the trade-off clearly; decreasing γ causes the hovercraft to try very hard to stay close to the origin. Also notice the asymmetry caused by the different times at which the hovercraft must be at the waypoints.

h) A good choice of gamma is about 1.7×10^4. Here the trajectory just remains within range, as shown below.

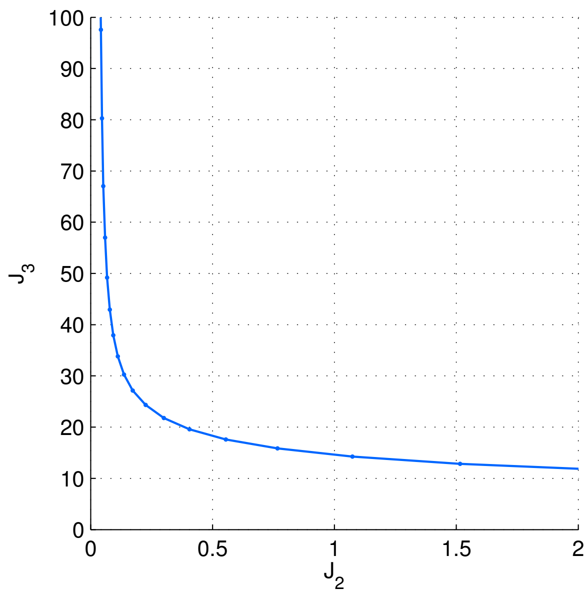


This is not the smallest-norm u that keeps the hovercraft within range and drives the hovercraft through the waypoints, because we are minimizing the *sum* of the squares of $\|q(t)\|$, rather than constraining each $\|q(t)\|$ independently. You can see this in the plot, since in the final stretch the hovercraft is expending extra effort to stay well within range, and this excessive input could be reduced.

In fact, one can compute the exact optimal, but this is not required and not covered in this course; (an approximation of) it is below.



i) The trade-off is below.



Notice that the vertical asymptote occurs when $J_2 \approx 0.03$; this is the minimum-norm of u which drives the hovercraft through the desired trajectory, as seen in part (b).

code that solves this problem
helper functions

```
function y=vec(x)
% VEC produces a vector of length m*n from an m by n matrix
```

```
%
%  function y=vec(x)
%
% Given an m by n matrix x, y=vec(x) constructs a vector y
% consisting of the columns of x stacked on top of each other
%

[m,n] = size(x);

y = reshape(x,m*n,1);


function T=sys_toeplitz(A,B,C,D,out_times,in_times);
% SYS_TOEPLITZ computes toeplitz matrices for a discrete-time LDS
%
%   T=sys_toeplitz(A,B,C,D,out_times,in_times);
%
% A,B,C,D specify a discrete-time state-space realization
%
% out_times and in_times are row vectors
%
% for example, if out_times is [1,2,4] and in_times is [0:10]
% then T is the matrix which maps
%
%   [u(0); u(1); ...  u(10)]  to  [y(1); y(2); y(4)]
%
% Notice that T is Toeplitz if out_times and in_times are
% both of the form a:b

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% setup parameters

% number of states
n=size(A,1);

% num inputs and outputs
ny=size(C,1);
nu=size(B,2);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% construct the Toeplitz matrix

% this is neither efficient nor numerically reliable for big matrices
% but is simple and works well for small cases
```

```
T=[];
for r=out_times

  % now create a row for this output time
  T_row=[];

  for s=in_times
    % three cases; either CA^tB, D, or 0

    if s<r
      % below the diagonal
      this_block= C*A^(r-s-1)*B;

    elseif s==r
      % on the diagonal
      this_block=D;

    else
      % above the diagonal
      this_block=zeros(ny,nu);

    end

    T_row=[T_row, this_block];
  end

  T=[T; T_row];

end
```

main code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% compute min-norm input that drives a hovercraft through a
% given set of waypoints


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% parameters

% desired radius
r_max=2;

% desired time steps and positions
```

```
way_times=[ 6 40 50 ];

way_points=[    1,   0,   -1.5  ;
     -0.5,   1,    0    ];

% final time step
t_max=70;

% sampling time
h=1;

% discrete-time system
A=[1 h 0 0 ;
   0 1 0 0 ;
   0 0 1 h ;
   0 0 0 1 ];

B=[h^2/2 0;
   h 0;
   0 h^2/2 ;
   0 h ];

C=[ 1 0 0 0 ;
    0 0 1 0 ];

D=[ 0 0 ;
    0 0 ];

% number of states
n=size(A,1);

% num inputs and outputs
ny=size(C,1);
nu=size(B,2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% real work is here

% toeplitz matrix mapping inputs to position at way_times
A_hov=sys_toeplitz(A,B,C,D, way_times, 0:t_max-1);

% find minimum norm input
u_tmp=pinv(A_hov)*vec(way_points);

% for convenience of simulation, reshape the inputs
```

```
% so that u_opt(:,k+1) is the input vector at time k
u_opt=reshape(u_tmp,2,t_max);

% simulate
x=zeros(n,t_max+1);
for k=0:t_max-1
  x(:,k+2)=A*x(:,k+1) + B*u_opt(:,k+1);
end
y=C*x;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% everything from here on is just plotting

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% plot trajectory in the plane

figure(1);
clf;
hold on;
axis equal;
axis([-2.5,2.5,-2.5,2.5]);
grid;
box on;


% plot radio range
[xs,ys]=ellipse(r_max^2*eye(2),[0;0]);
plot(xs,ys,'r');

% plot way points
for k=1:size(way_times,2)
  plot(way_points(1,k),way_points(2,k),'ko');
end

% plot trajectory
plot(y(1,:),y(2,:),'.-');



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% plot thruster input versus time

figure(2);
clf;
hold on;
```

```
grid;

h=plot((0:t_max-1),u_opt(1,1:t_max),'b.-');
h=plot((0:t_max-1),u_opt(2,1:t_max),'r.-');




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% for a hovercraft
% compute the trade-off curve of input norm to distance norm
% subject to the constraint that
% the trajectory passes through the waypoints

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% parameters

% desired radius
r_max=2;

% desired time steps and positions
way_times=[ 6 40 50 ];

way_points=[    1,   0,   -1.5  ;
     -0.5,   1,    0    ];

n_way_points=size(way_points,2);

% final time step
t_max=70;

% sampling time
h=1;

% discrete-time system

A=[1 h 0 0 ;
   0 1 0 0 ;
   0 0 1 h ;
   0 0 0 1 ];

B=[h^2/2 0;
   h 0;
   0 h^2/2 ;
   0 h ];
```

```
C=[ 1 0 0 0 ;
    0 0 1 0 ];

D=[ 0 0 ;
    0 0 ];

% number of states
n=size(A,1);

% num inputs and outputs
ny=size(C,1);
nu=size(B,2);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% compute trade-off for weighted min-norm problem


% toeplitz matrix mapping inputs to position at way_times
A_hov=sys_toeplitz(A,B,C,D, way_times, 0:t_max-1);

% toeplitz matrix mapping inputs to sequence of positions
A_pos=sys_toeplitz(A,B,C,D, 0:t_max, 0:t_max-1);

% desired gamma values
gammas=logspace(0,5,40);


% space to save costs
J2=[];
J3=[];

for i=1:size(gammas,2)
  this_gamma=gammas(i);

  % weight parameter - sigma inverse
  sig_inv=inv(A_pos'*A_pos + this_gamma*eye(t_max*nu));

  % stack up the way_points
  y_des=reshape(way_points,2*n_way_points,1);

  % compute input that minimizes weighted cost
  lambda=(A_hov*sig_inv*A_hov')\y_des;
  u=sig_inv*A_hov'*lambda;
```

```
  % compute corresponding trajectory in the plane
  y=A_pos*u;

  % keep track of achieved costs
  J3(i) = norm(y)^2;
  J2(i) = norm(u)^2;

  % store a nicely shaped y for plotting
  y_keep{i}=reshape(y,2,t_max+1);

end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% everything from here is just plotting

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% first plot trade off curve

% figure setup
figure(1);
clf;
hold on;
grid;
axis([0,3,0,200]);

% plot
plot(J2,J3,'.-');
xlabel('J_2 position cost');
ylabel('J_3 input cost');
title('trade off curve')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% now plot all the different inputs

% loop over each gamma value
for i=1:size(gammas,2)

  figure(2);
  clf;
  hold on;
  axis equal;
  axis([-2.5,2.5,-2.5,2.5]);
  grid;
```

```
  box on;
  title(sprintf('gamma=%g',gammas(i)));

  % plot radio range
  [xs,ys]=ellipse(r_max^2*eye(2),[0;0]);
  plot(xs,ys,'r');

  % plot way points
  for k=1:size(way_times,2)
    plot(way_points(1,k),way_points(2,k),'ko');
  end

  % plot trajectory
  y=y_keep{i};
  plot(y(1,:),y(2,:),'.-');

  % wait for 100 milliseconds between plots
  pause(0.1);

end
```

13.2030. A method for rapidly driving the state to zero. We consider the discrete-time linear dynamical system

$$x(t+1) = Ax(t) + Bu(t),$$

where $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times k}$, $k < n$, is full rank. The goal is to choose an input u that causes $x(t)$ to converge to zero as $t \to \infty$. An engineer proposes the following simple method: at time t, choose $u(t)$ that minimizes $\|x(t+1)\|$. The engineer argues that this scheme will work well, since the norm of the state is made as small as possible at every step. In this problem you will analyze this scheme.

a) Find an explicit expression for the proposed input $u(t)$ in terms of $x(t)$, A, and B.

b) Now consider the linear dynamical system $x(t+1) = Ax(t) + Bu(t)$ with $u(t)$ given by the proposed scheme (*i.e.*, as found in (a)). Show that x satisfies an autonomous linear dynamical system equation $x(t+1) = Fx(t)$. Express the matrix F explicitly in terms of A and B.

c) Now consider a specific case:

$$A = \begin{bmatrix} 0 & 3 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

Compare the behavior of $x(t+1) = Ax(t)$ (*i.e.*, the orginal system with $u(t) = 0$) and $x(t+1) = Fx(t)$ (*i.e.*, the original system with $u(t)$ chosen by the scheme described above) for a few initial conditions. Determine whether each of these systems is stable.

Solution.

a) We should choose $u(t)$ such that $\|x(t+1)\| = \|Ax(t)+Bu(t)\|$ is minimized. This is simply a least-squares problem in the form $\min_x \|\bar{y} - \bar{A}\bar{x}\|$ where $\bar{y} := Ax(t)$, $\bar{A} := -B$ and $\bar{x} := u(t)$. Therefore the minimizing $u(t)$ is

$$u(t) = \bar{x}_{\text{ls}} = (\bar{A}^\mathsf{T}\bar{A})^{-1}\bar{A}^\mathsf{T}\bar{y} = -(B^\mathsf{T}B)^{-1}B^\mathsf{T}Ax(t).$$

b) We have

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t) \\ &= Ax(t) - B(B^\mathsf{T}B)^{-1}B^\mathsf{T}Ax(t) \\ &= (I - B(B^\mathsf{T}B)^{-1}B^\mathsf{T})Ax(t), \end{aligned}$$

and therefore

$$F = (I - B(B^\mathsf{T}B)^{-1}B^\mathsf{T})A.$$

c) With A and B as given

$$F = \begin{bmatrix} 0 & 1.5 \\ 0 & -1.5 \end{bmatrix}.$$

The eigenvalues of A are 0, 0 and as a result $x(t+1) = Ax$ is stable. However, the eigenvalues of F are 0, -1.5 and therefore F is unstable. Thus, this method, though reasonable sounding, not only does not rapidly drive the state to zero — it can actually destabilize a stable system! The state trajectory of the original system $x(t+1) = Ax(t)$ and for the system $x(t+1) = Fx(t)$ are shown in Figures 1 and 2 respectively for two initial conditions. Clearly, for the system $x(t+1) = Ax(t)$, $x(t)$ goes to zero as t increases (actually just after two steps), while for the system $x(t+1) = Fx(t)$, $\|x(t)\|$ increases as t increases.

18.1250. Chasing a sea monster. A sea monster is loose in the Pacific Ocean! Your monster-chasing colleague has been measuring the sea monster's movements and has predicted it will surface at m positions $p_i \in \mathbb{R}^2$ at times s_i. Here p_i is the ith column of the matrix P given by

$$P = \begin{bmatrix} 1 & 1.75 & 2.4 & 2 & 0.5 & 0 \\ 0.75 & 0.6 & 1.2 & 2.3 & 0.75 & 0 \end{bmatrix}$$

and the times $s = (2, 5, 8, 11, 17, 20)$. You plan to observe the monster with a drone. Unfortunately the sea monster ate the last two drones you sent and you are almost out of research funding so your drone's sensors are not very good, and the drone must be exactly in the right position to observe the monster.

a) The dynamics of the drone are

$$\ddot{q} = u$$

where $q \in \mathbb{R}^2$ is the position of the drone, and $u \in \mathbb{R}^2$ is an input force. Write this as a linear dynamical system of the form

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned}$$

where $y \in \mathbb{R}^2$ is the position of the drone.

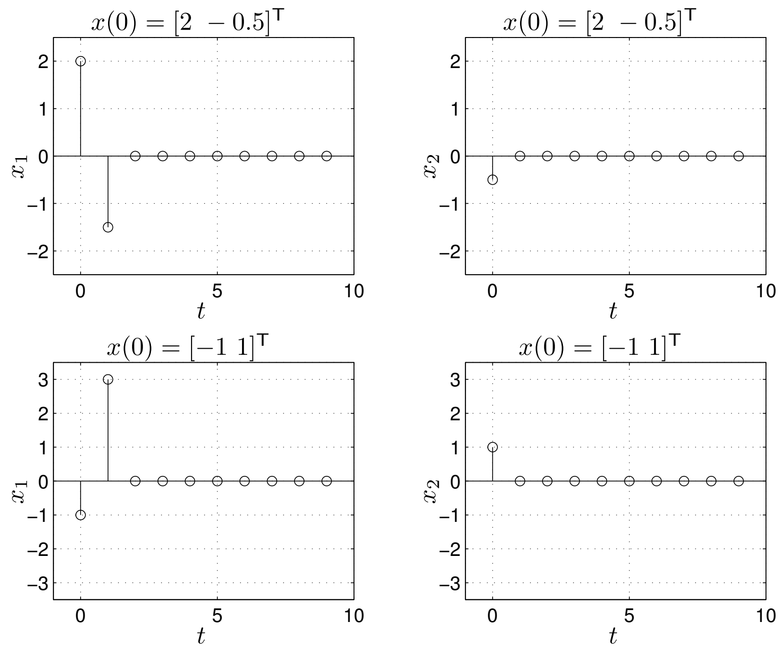


Figure 1: state trajectory for system $x(t+1) = Ax(t)$ for $x(0) = [2\ -0.5]^{\mathsf{T}}$ and $x(0) = [-1\ 1]^{\mathsf{T}}$

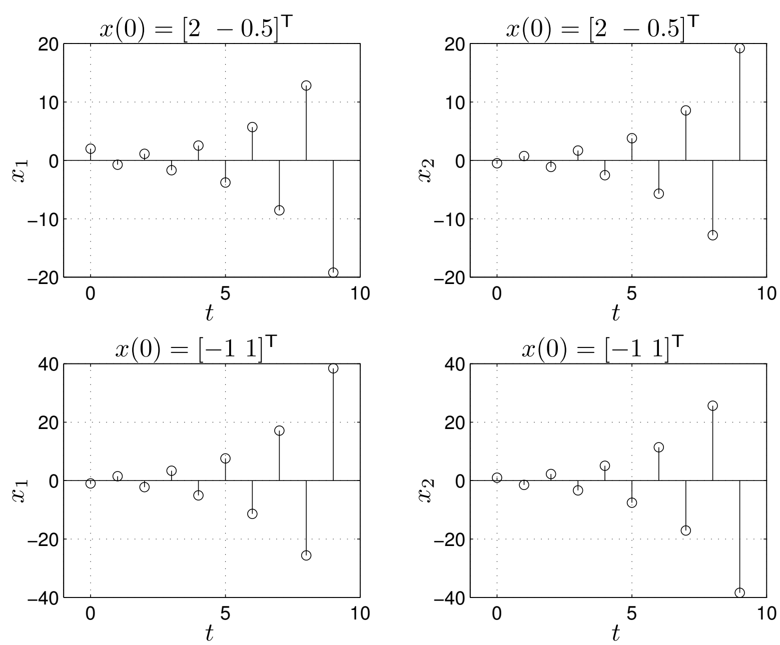


Figure 2: state trajectory for system $x(t+1) = Fx(t)$ for $x(0) = [2\ -0.5]^{\mathsf{T}}$ and $x(0) = [-1\ 1]^{\mathsf{T}}$

b) We will use sample period h. Assume that the force input is piecewise constant on sample intervals, and construct the exact discretization

$$\begin{aligned} x_d(k+1) &= A_d x_d(k) + B_d u_d(k) \\ y_d(k) &= C_d x_d(k) \end{aligned}$$

where $x_d(k) = x(kh)$, and similarly for y_d and u_d.

c) The drone starts at the origin with zero velocity, and we would like to move the drone so that $y(s_i) = p_i$ for $i = 1, \dots, m$. We will operate the drone on the time interval $[0, T]$ where $T = s_m$. For convenience, let $N = T/h$. Since drone batteries are limited, we would like to minimize

$$J = \sum_{k=0}^{N-1} \|u_d(k)\|^2$$

Explain in detail how you would solve this problem.

d) Use your method to compute the optimal input u, and plot u versus time. Use $h = 0.1$.

e) Report the optimal value of J that you obtained.

f) Plot the trajectory of the drone. Use axes q_1 and q_2, so that the plot shows the path followed by the drone. Mark on your plot the points p_i where the monster surfaces.

g) Draw a sea monster for 1 point of extra credit.

Solution.

a) The dynamics are

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned}$$

where

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

b) We have

$$A_d = \begin{bmatrix} 1 & h & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & h \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad B_d = \begin{bmatrix} h^2/2 & 0 \\ h & 0 \\ 0 & h^2/2 \\ 0 & h \end{bmatrix} \quad C_d = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

c) We want to constrain the drone's position $y_d(s_i)$ for $i = 1, \ldots, m$. This is several constraints and we will have to concatenate them into one large equality constraint. We have

$$\begin{aligned} y_d(t) &= \begin{bmatrix} C_d A_d^{t-1} B_d & \ldots & C_d A_d B_d & C_d B_d & 0 \ldots 0 \end{bmatrix} \begin{bmatrix} u_d(0) \\ \vdots \\ u_d(N-1) \end{bmatrix} \\ &= H_t u \end{aligned}$$

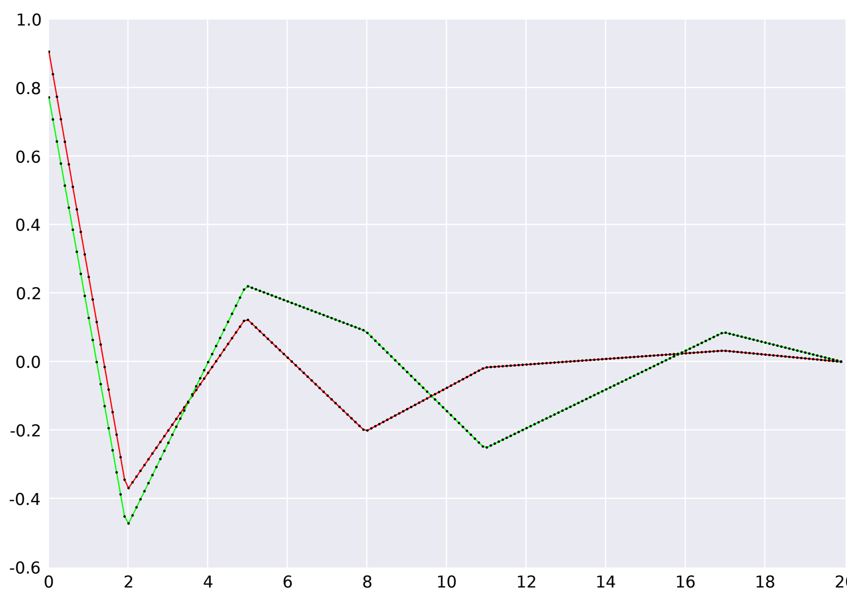
Define $k_i = s_i/h$ and let the matrices H and z be

$$H = \begin{bmatrix} H_{k_1} \\ \vdots \\ H_{k_m} \end{bmatrix} \qquad z = \begin{bmatrix} p_1 \\ \vdots \\ p_m \end{bmatrix}$$

Then this is a minimum norm problem, and so the optimal u is given by

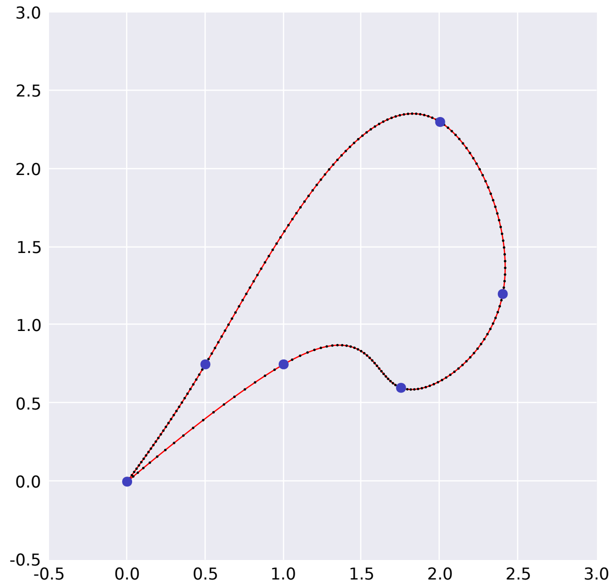
$$h = H^\dagger z$$

d) The plot is below.



e) The optimal J is $J = 12.90$.

f) The trajectory is shown below.



g) Sea monster.

18.2890. Linear dynamical systems for portfolio management. We consider a portfolio of n financial assets (like stocks) and cash, which we manage over T time steps of unit length (e.g. one month). We call $x_t \in \mathbb{R}^{n+1}$ for $t = 1, \ldots, T$ our state vector. The first n elements are our positions in each of the assets, in dollars, and the last element is the dollar amount of cash we hold. Every element of x can be either positive (for long positions) and negative (for short or borrowing). For $t = 1, \ldots, T-1$, the transition from x_t to x_{t+1} is composed of two steps.

- First, the portfolio positions change value because of market returns. Let $\mu \in \mathbb{R}^n_{++}$ be the vector of returns, where $\mathbb{R}^n_{++}$ is the set of all vectors of length n with strictly positive entries. We define the post-return portfolio $\tilde{x}_t$ to be

$$(\tilde{x}_t)_i = \begin{cases} \mu_i (x_t)_i & i = 1, \dots, n, \\ (x_t)_i & i = n+1. \end{cases}$$

(Intuitively, cash is unchanged, and the asset positions are multiplied by the corresponding element of the vector of returns.) For simplicity we assume that the vector of returns does not change in time.

- Then we trade. We can exchange any amount of cash for the corresponding amount of any of the assets. Note that the *only* valid trades are cash for asset. If you wish to trade some amount of an asset with the same amount of another asset, you have to perform *two trades*: trade the first asset with cash, and then trade cash with the second asset. (Think carefully about this definition of trade when you formulate the transaction costs.) For example, if we buy $c > 0$ dollars of the first asset and sell $d > 0$ dollars of the second asset the state evolves as

$$x_{t+1} = \tilde{x}_t + \begin{bmatrix} c \\ -d \\ 0 \\ \vdots \\ 0 \\ -(c-d) \end{bmatrix}.$$

Finally, we define the portfolio value $v_t \in \mathbb{R}$ for $t = 1, \dots, T$ to be

$$v_t = \mathbf{1}^T x_t.$$

a) Formulate the problem as a linear dynamical system of the form

$$x_{t+1} = Ax_t + Bu_t, \quad t = 1, \dots, T-1.$$

The control vector u_t should have dimension n.

b) Assume that our trades incur quadratic transaction costs with parameter $\rho > 0$. For example, if at time t we buy $c > 0$ dollars of the first asset, and we sell $d > 0$ dollars of the second asset (the example above), then the transaction costs for the transition x_t to x_{t+1} are

$$\rho(c^2 + d^2).$$

(Be careful, they are **not** $\rho(c+d)^2$.) Explain how to solve the problem of maximizing the final value of the portfolio v_T minus the total transaction costs. (The sequence of controls $u_1, \dots, u_{T-1}$ that achieves the maximum should be a function of A, B, and ρ). Use methods from EE263.

c) Apply your method to the following data.

```
T = 12;
x_1 = [1000, 1000, 0, 1000, 0, 0];
mu = [1.001, 1.003, 1.004, 1.006, 1.007];
rho = 0.0001;
```

What is the final value v_T? What are the total transaction costs? Plot the trajectories of the portfolio positions x_t. (On the same plot you should draw $n+1$ lines, one for each of the assets and cash, with time on the x-axis.)

d) Now assume that we aim to *liquidate* an initial portfolio, which means that at time T we want to have zero positions in any of the n assets and only hold cash. We thus impose the constraint $(x_T)_i = 0$, for $i = 1, \dots, n$. Explain how to solve the problem of maximizing the final portfolio value (in this case, all cash) minus the transaction costs with this additional constraint. Use methods from EE263.

e) Apply your method to the data given above. What is the final value v_T? What are the total transaction costs? Plot the trajectories of the portfolio positions x_t. (On the same plot you should draw $n+1$ lines, one for each of the assets and cash, with time on the x-axis.)

Solution.

a) We define the vector

$$\tilde{\mu} = \begin{bmatrix} \mu \\ 1 \end{bmatrix},$$

then

$$\tilde{x}_t = \mathrm{diag}(\tilde{\mu}) x_t,$$

and we call

$$A = \mathrm{diag}(\tilde{\mu}).$$

We represent the trades by a vector $u_t \in \mathbb{R}^n$ for $t = 1, \dots, T-1$. Each element is the dollar amount of each of the assets that we exchange for cash. Then the matrix B is

$$B = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & & & 1 \\ -1 & -1 & \cdots & -1 \end{bmatrix}.$$

We thus have that

$$x_{t+1} = Ax_t + Bu_t, \quad t = 1, \dots, T-1.$$

b) The final state is given by

$$x_T = A^{T-1} x_1 + \sum_{t=1}^{T-1} A^{T-t-1} B u_t$$

or equivalently

$$x_T = A^{T-1} x_1 + \mathcal{B}^T \mathcal{U}$$

with

$$\mathcal{B}^T = \left[A^{T-2}B \ A^{T-3}B \ \cdots \ B\right], \quad \mathcal{U} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{T-1} \end{bmatrix}.$$

Thus, the final value of the portfolio is given by

$$v_T = \mathbf{1}^T A^{T-1} x_1 + \mathbf{1}^T \mathcal{B}^T \mathcal{U}.$$

The quadratic transaction costs are

$$\rho \|\mathcal{U}\|^2.$$

The maximization problem is

$$\text{maximize} \quad v_T - \rho\|\mathcal{U}\|^2$$

since the first term of v_T is a constant, this is equivalent to

$$\text{maximize} \quad \mathbf{1}^T \mathcal{B}^T \mathcal{U} - \rho\|\mathcal{U}\|^2$$

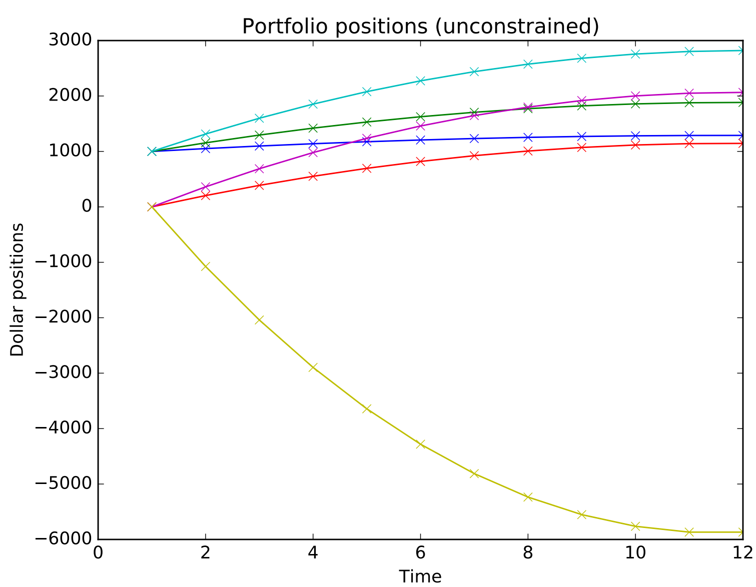
the solution is (by the first order condition, since the function is strictly concave)

$$\mathcal{U} = \frac{1}{2\rho}\mathcal{B}\mathbf{1}.$$

c) The final value and total transaction costs are

$$v_T = 3335.06, \quad \rho\|\mathcal{U}\|^2 = 111.24, \quad v_T - \rho\|\mathcal{U}\|^2 = 3223.81.$$

The yellow line is cash, the others are the asset positions.



d) The optimization problem is

$$\begin{array}{cc} \text{maximize} & v_T - \rho\|\mathcal{U}\|^2 \\ \text{s.t.} & (x_T)_i = 0 \quad i = 1,\ldots,n. \end{array}$$

Let $\bar{x}_t \in \mathbb{R}^n$ be the first n elements of each vector x_t. Then we have

$$\bar{x}_{t+1} = \mathrm{diag}(\mu)\bar{x}_t + Iu_t,$$

$$\bar{x}_T = \mathrm{diag}(\mu)^{T-1}\bar{x}_1 + \sum_{t=1}^{T-1} \mathrm{diag}(\mu)^{T-t-1}u_t$$

or

$$\bar{x}_T = \mathrm{diag}(\mu)^{T-1}\bar{x}_1 + \mathcal{C}^T\mathcal{U}$$

with

$$\mathcal{C} = \begin{bmatrix} \mathrm{diag}(\mu)^{T-2} \\ \mathrm{diag}(\mu)^{T-3} \\ \vdots \\ I \end{bmatrix}.$$

Thus the optimization problem is

$$\begin{array}{cc} \text{maximize} & v_T - \rho\|\mathcal{U}\|^2 \\ \text{s.t.} & \mathcal{C}^T\mathcal{U} = -\,\mathrm{diag}(\mu)^{T-1}\bar{x}_1. \end{array}$$

We could solve this problem by introducing a Lagrange multiplier, but instead we transform the problem by subtracting the optimal solution to (b)

$$\mathcal{U}' = \mathcal{U} - \frac{1}{2\rho}\mathcal{B}\mathbf{1}$$

The objective function is

$$\mathbf{1}^T\mathcal{B}^T(\mathcal{U}' + \frac{1}{2\rho}\mathcal{B}\mathbf{1}) - \rho\|\mathcal{U}' + \frac{1}{2\rho}\mathcal{B}\mathbf{1}\|^2.$$

Expanding the squared norm and ignoring constant terms we obtain that the problem is equivalent to

$$\begin{array}{cc} \text{maximize} & -\rho\|\mathcal{U}'\|^2 \\ \text{s.t.} & \mathcal{C}^T\mathcal{U}' = y \end{array}$$

where

$$y = -\,\mathrm{diag}(\mu)^{T-1}\bar{x}_1 - \frac{1}{2\rho}\mathcal{C}^T\mathcal{B}\mathbf{1}.$$

Note that $\mathcal{C}$ is full rank. The solution (least-norm underdetermined system) is

$$\mathcal{U}' = \mathcal{C}(\mathcal{C}^T\mathcal{C})^{-1}y$$

and

$$\mathcal{U} = \mathcal{C}(\mathcal{C}^T\mathcal{C})^{-1}y + \frac{1}{2\rho}\mathcal{B}\mathbf{1}.$$

e) The final value and total transaction costs are

$$v_T = 3120.65, \quad \rho\|\mathcal{U}\|^2 = 58.72, \quad v_T - \rho\|\mathcal{U}\|^2 = 3061.94.$$

The yellow line is cash, the others are the asset positions.