1. The smoothest input that takes the state to zero. We consider the discrete-time linear dynamical system \( x(t+1) = Ax(t) + Bu(t) \), with

\[
A = \begin{bmatrix} 1.0 & 0.5 & 0.25 \\ 0.25 & 0 & 1.0 \\ 1.0 & -0.5 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 1.0 \\ 0.1 \\ 0.5 \end{bmatrix}, \quad x(0) = \begin{bmatrix} 25 \\ 0 \\ -25 \end{bmatrix}.
\]

The goal is to choose an input sequence \( u(0), u(1), \ldots, u(19) \) that yields \( x(20) = 0 \). Among the input sequences that yield \( x(20) = 0 \), we want the one that is smoothest, i.e., that minimizes

\[
J_{\text{smooth}} = \left( \frac{1}{20} \sum_{t=0}^{19} (u(t) - u(t-1))^2 \right)^{1/2},
\]

where we take \( u(-1) = 0 \) in this formula. Explain how to solve this problem. Plot the smoothest input \( u_{\text{smooth}} \), and give the associated value of \( J_{\text{smooth}} \).

Solution. We first express \( x(20) \) in terms of \( u(0), \ldots, u(19) \) and \( x(0) \). Using the linear recursion \( x(t+1) = Ax(t) + Bu(t) \), we get

\[
x(2) = Ax(1) + Bu(1)
\]
\[
= A(Ax(0) + Bu(0)) + Bu(1)
\]
\[
= A^2 x(0) + ABu(0) + Bu(1)
\]

Continuing in this way we get

\[
x(20) = A^{20} x(0) + A^{19} Bu(0) + \cdots + ABu(18) + Bu(19).
\]

In matrix form, and substituting \( x(20) = 0 \), we have

\[
\begin{bmatrix} A^{19}B & \cdots & AB & B \end{bmatrix} \begin{bmatrix} u(0) \\ \vdots \\ u(18) \\ u(19) \end{bmatrix} = -A^{20} x(0). \tag{1}
\]

Thus, we have an underdetermined set of linear equations, with 3 equations and 20 variables. We know how to find the smallest (or least-norm) solution of this set of equations, but we’re
asked here to find the smoothest solution, so we have to do a little more work. We define the difference between two consecutive inputs as \( \delta(t) = u(t) - u(t-1) \), with \( \delta(0) = u(0) \) (consistent with \( u(-1) = 0 \)). These are sometimes called input increments or input differences. We can express the inputs in terms of these increments as

\[
u(t) = \delta(0) + \cdots + \delta(t),
\]

which can be expressed in matrix form as \( u = F\delta \), where \( u = (u(0), \ldots, u(19)) \), \( \delta = (\delta(0), \ldots, \delta(19)) \), and

\[
F = \begin{bmatrix}
1 & 0 & 0 & \cdots & 0 & 0 \\
1 & 1 & 0 & \cdots & 0 & 0 \\
\vdots & & & \ddots & & \\
1 & 1 & 1 & \cdots & 1 & 0 \\
1 & 1 & 1 & \cdots & 1 & 1
\end{bmatrix}.
\]

Now, we substitute \( u = F\delta \) into (?) to obtain

\[
\begin{bmatrix}
A^{19}B & \cdots & AB & B
\end{bmatrix} F\delta = -A^{20}x(0).
\]

Defining \( G = \begin{bmatrix}
A^{19}B & \cdots & AB & B
\end{bmatrix} F \), we can express this as

\[
G\delta = -A^{20}x(0).
\]

The original smoothness measure can be expressed in terms of \( \delta \) as

\[
J_{\text{smooth}} = \left( \frac{1}{20} \sum_{t=0}^{19} (u(t) - u(t-1))^2 \right)^{1/2} = \left( \frac{1}{20} \sum_{t=0}^{19} \delta(t)^2 \right)^{1/2} = \frac{1}{\sqrt{20}} \|\delta\|.
\]

Thus, we have reduced our problem to one we know how to solve: find the least norm vector \( \delta_{\text{ln}} \) that satisfies \( G\delta = -A^{20}x(0) \). The solution is given by

\[
\delta_{\text{ln}} = G^T(GG^T)^{-1}(-A^{20}x(0)).
\]

Then the smoothest input is given by

\[
u_{\text{smooth}} = F\delta_{\text{ln}} = -FG^T(GG^T)^{-1}A^{20}x(0).
\]

Its RMS value is \( J_{\text{smooth}} = 1/\sqrt{20}\|\delta_{\text{ln}}\| \). Below is the matlab code used to compute the smoothest input.

```matlab
A = [1 .5 .25; .25 0 1; 1 -0.5 0];
B = [1; .1; .5];
x0 = [25; 0; -25];
N = 20;
Ck = [];
for k = 0:N-1
```

2
Ck = [A^k*B Ck];
end
F = tril(ones(N,N)); % lower triagonal of ones
del_smooth = pinv(Ck*F)*(-A^N*x0);
u_smooth = F*del_smooth
plot([0:N-1],u_smooth); xlabel('time'); ylabel('usmooth');
Jsmooth = 1/sqrt(N)*norm(del_smooth)
xN = A^N*x0 + Ct*u_smooth
>> xN =
  1.0e-10 *
  -0.0546
  -0.2910
  0.0909

The plot of the smoothest input is given below. Its RMS smoothness measure is $J_{\text{smooth}} = 1.1246$.

It is interesting to note some of the alternative methods people used to solve this problem. One method worked as follows: we have three equality constraints, i.e., the constraint that the three components of $x(20)$ be zero. We can use these to eliminate three of our variables, say, $u(17)$, $u(18)$, and $u(19)$. Thus, we have an unconstrained problem, with 17 variables, $\tilde{u} = (u(0), \ldots, u(16))$. We can choose any values we like for these, and then $u(17)$, $u(18)$, and $u(19)$ are determined (to ensure $x(20) = 0$). Finally, we express the objective, the smoothness measure, as the norm of a linear function of $\tilde{u}$ minus a constant vector. Then we apply least-squares. (That’s the idea — the details and equations are not too pretty!) This method is correct, i.e., it gets the exact solution, so it got full credit. Some people formulated the the problem as a mixed quadratic problem with equality constraints, and derived the general solution of such problems. This also works. The most popular alternative approach involved a regularization method. In this method, the input $u$ is considered unconstrained. We identify
two objectives, $J_{\text{smooth}}$ and $J_{\text{miss}}$, where $J_{\text{miss}}$ gives the final state error:

$$J_{\text{miss}} = \|x(20)\|.$$ 

We then form a regularized problem: we minimize

$$J_{\text{smooth}}^2 + \lambda J_{\text{miss}}^2$$

where $\lambda > 0$ is a parameter. Now we let $\lambda$ get really big, which drives the final state error to be really small. This method works, at least in the limit. Even though the solution with a large value of $\lambda$ is numerically very close to the exact solution, we did deduct some points for this. The reason is simple: this method is only approximate, or involves a limit. The other methods described above, in contrast, are exact.

2. Portfolio selection with sector neutrality constraints. We consider the problem of selecting a portfolio composed of $n$ assets. We let $x_i \in \mathbb{R}$ denote the investment (say, in dollars) in asset $i$, with $x_i < 0$ meaning that we hold a short position in asset $i$. We normalize our total portfolio as $\mathbf{1}^T x = 1$, where $\mathbf{1}$ is the vector with all entries 1. (With normalization, the $x_i$ are sometimes called portfolio weights.)

The portfolio (mean) return is given by $r = \mu^T x$, where $\mu \in \mathbb{R}^n$ is a vector of asset (mean) returns. We want to choose $x$ so that $r$ is large, while avoiding risk exposure, which we explain next.

First we explain the idea of sector exposure. We have a list of $k$ economic sectors (such as manufacturing, energy, transportation, defense, ...). A matrix $F \in \mathbb{R}^{k \times n}$, called the factor loading matrix, relates the portfolio $x$ to the factor exposures, given as $R_{\text{fact}} = F x \in \mathbb{R}^k$. The number $R_{\text{fact}}^i$ is the portfolio risk exposure to the $i$th economic sector. If $R_{\text{fact}}^i$ is large (in magnitude) our portfolio is exposed to risk from changes in that sector; if it is small, we are less exposed to risk from that sector. If $R_{\text{fact}}^i = 0$, we say that the portfolio is neutral with respect to sector $i$.

Another type of risk exposure is due to fluctuations in the returns of the individual assets. The idiosyncratic risk is given by

$$R_{\text{id}} = \sum_{i=1}^{n} \sigma_i^2 x_i^2,$$

where $\sigma_i > 0$ are the standard deviations of the asset returns. (You can take the formula above as a definition; you do not need to understand the statistical interpretation.)

We will choose the portfolio weights $x$ so as to maximize $r - \lambda R_{\text{id}}$, which is called the risk-adjusted return, subject to neutrality with respect to all sectors, i.e., $R_{\text{fact}} = 0$. Of course we also have the normalization constraint $\mathbf{1}^T x = 1$. The parameter $\lambda$, which is positive, is called the risk aversion parameter. The (known) data in this problem are $\mu \in \mathbb{R}^n$, $F \in \mathbb{R}^{k \times n}$, $\sigma = (\sigma_1, \ldots, \sigma_n) \in \mathbb{R}^n$, and $\lambda \in \mathbb{R}$.

a) Explain how to find $x$, using methods from the course. You are welcome (even encouraged) to express your solution in terms of block matrices, formed from the given data.

b) Using the data given in sector_neutral_portfolio_data.m, find the optimal portfolio. Report the associated values of $r$ (the return), and $R_{\text{id}}$ (the idiosyncratic risk). Verify that $\mathbf{1}^T x = 1$ (or very close) and $R_{\text{fact}} = 0$ (or very small).
Solution.

a) We define $\Sigma \in \mathbb{R}^{n \times n}$ to be a diagonal matrix with $\Sigma_{ii} = \sigma_i^2$, so $R_{id} = x^T \Sigma x$. The problem we are trying to solve is

$$\begin{align*}
\text{maximize} & \quad \mu^T x - \lambda x^T \Sigma x \\
\text{subject to} & \quad 1^T x = 1, \quad Fx = 0
\end{align*}$$

with variable $x \in \mathbb{R}^n$. Maximizing an objective is equivalent to minimizing the negative of the objective, so we can rewrite this as

$$\begin{align*}
\text{minimize} & \quad -\mu^T x + \lambda x^T \Sigma x \\
\text{subject to} & \quad 1^T x = 1, \quad Fx = 0
\end{align*}$$

We introduce Lagrange multipliers $\kappa \in \mathbb{R}$ and $\nu \in \mathbb{R}^k$ for the two constraints, and write the Lagrangian of this problem as

$$L(x, \nu, \kappa) = -\mu^T x + \lambda x^T \Sigma x + \nu^T (Fx) + \kappa (1^T x - 1).$$

The optimality conditions are then given by

$$\nabla_x L = -\mu + 2\lambda \Sigma x + F^T \nu + \kappa 1 = 0, \quad \nabla_\nu L = Fx = 0, \quad \nabla_\kappa L = 1^T x - 1 = 0,$$

which we can write in block matrix form as

$$\begin{bmatrix} 2\lambda \Sigma & F^T & 1 \\ F & 0 & 0 \\ 1^T & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \nu \\ \kappa \end{bmatrix} = \begin{bmatrix} \mu \\ 0 \\ 1 \end{bmatrix}.$$ 

To find the optimal $x$ we solve this set of $n + k + 1$ linear equations in $n + k + 1$ variables.

b) Using the data provided we arrive at optimal values $r = 26.71$, and $R_{id} = 133.6$, with the optimal objective value of the original problem being 13.3. The following matlab code implements the method of part (a).

```matlab
% sector_neutral_portfolio_data

% construct the relevant matrices from the data
M=[2*lambda*diag(sigmas.^2) F' ones(n,1); 
   F zeros(k,k) zeros(k,1); 
   ones(1,n) zeros(1,k) 0];

v=[mu;zeros(k,1);1];

% this solves the set of linear equations M*all_vars = v
all_vars = M\v;

% extract the portfolio weights
x=all_vars(1:n)

% plug in to find the optimal objective value
objective = -(-mu'*x + lambda*x'*diag(sigmas.^2)*x)

r = mu'*x
R_id = x'*diag(sigmas.^2)*x
```

5
3. Analysis of a power control algorithm. In this problem we consider again the power control method described in homework problem 2.1. Please refer to this problem for the setup and background. In that problem, you expressed the power control method as a discrete-time linear dynamical system, and simulated it for a specific set of parameters, with several values of initial power levels, and two target SINRs. You found that for the target SINR value $\gamma = 3$, the powers converged to values for which each SINR exceeded $\gamma$, no matter what the initial power was, whereas for the larger target SINR value $\gamma = 5$, the powers appeared to diverge, and the SINRs did not appear to converge. You are going to analyze this, now that you know alot more about linear systems.

a) Explain the simulations. Explain your simulation results from the problem 1(b) for the given values of $G$, $\alpha$, $\sigma$, and the two SINR threshold levels $\gamma = 3$ and $\gamma = 5$.

b) Critical SINR threshold level. Let us consider fixed values of $G$, $\alpha$, and $\sigma$. It turns out that the power control algorithm works provided the SINR threshold $\gamma$ is less than some critical value $\gamma_{\text{crit}}$ (which might depend on $G$, $\alpha$, $\sigma$), and doesn’t work for $\gamma > \gamma_{\text{crit}}$. (‘Works’ means that no matter what the initial powers are, they converge to values for which each SINR exceeds $\gamma$.) Find an expression for $\gamma_{\text{crit}}$ in terms of $G \in \mathbb{R}^{n \times n}$, $\alpha$, and $\sigma$. Give the simplest expression you can. Of course you must explain how you came up with your expression.

Solution.

a) In the homework we found that the powers propagate according to a linear system. The power update rule for a single transmitter can be found by manipulating the definitions given in the problem.

$$
p_i(t + 1) = \frac{\alpha \gamma p_i(t)}{S_i(t)} = \frac{\alpha \gamma p_i(t) q_i(t)}{s_i(t)} = \frac{\alpha \gamma p_i(t) \left[ \sigma + \sum_{j \neq i} G_{ij} p_j(t) \right]}{G_{ii} p_i(t)} = \frac{\alpha \gamma \left[ \sigma + \sum_{j \neq i} G_{ij} p_j(t) \right]}{G_{ii}}.
$$

In matrix form the equations represent a linear dynamical system with constant input, $p(t + 1) = Ap(t) + b$.

$$
\begin{bmatrix}
p_1(t + 1) \\
p_2(t + 1) \\
p_3(t + 1) \\
\vdots \\
p_n(t + 1)
\end{bmatrix}
= \alpha \gamma 
\begin{bmatrix}
0 & G_{12} & G_{13} & \cdots & G_{1n} \\
G_{21} & 0 & G_{23} & \cdots & G_{2n} \\
G_{31} & G_{32} & 0 & \cdots & G_{3n} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
G_{n1} & G_{n2} & G_{n3} & \cdots & 0
\end{bmatrix}
\begin{bmatrix}
p_1(t) \\
p_2(t) \\
p_3(t) \\
\vdots \\
p_n(t)
\end{bmatrix}
+ \alpha \gamma \sigma 
\begin{bmatrix}
0 \\
0 \\
0 \\
\vdots \\
0
\end{bmatrix}
\begin{bmatrix}
p_1(t) \\
p_2(t) \\
p_3(t) \\
\vdots \\
p_n(t)
\end{bmatrix}
\begin{bmatrix}
G_{11} \\
G_{22} \\
G_{33} \\
\vdots \\
G_{nn}
\end{bmatrix}
+ \alpha \gamma \sigma 
\begin{bmatrix}
0 \\
0 \\
0 \\
\vdots \\
0
\end{bmatrix}
\begin{bmatrix}
p_1(t) \\
p_2(t) \\
p_3(t) \\
\vdots \\
p_n(t)
\end{bmatrix}
+ b.
$$

where $A = \alpha \gamma P$. This is a discrete LDS, and is stable if and only if $|\lambda_i| < 1$ for all $i = 1, \ldots, n$, where $\lambda_i$ are the eigenvalues of $A$. When $\gamma = 3$ the eigenvalues of $A$ are 0.6085, $-0.3600$, and $-0.2485$, so the system is stable; for all initial conditions, the powers converge to their equilibrium values.
Also, the SINR at each receiver $i$, given by $S_i$, converges to the same constant value $\alpha \gamma$, which is enough for a successful signal reception. This can be shown by observing that at equilibrium $p_i(t + 1) = p_i(t) = \overline{p}_i$, and the power update equation gives

$$\overline{p}_i = \overline{p}_i(\alpha \gamma / S_i(t)).$$

After cancellation, we obtain the constant value for each SINR, $S_i = \alpha \gamma$.

When $\gamma = 5$, the eigenvalues of $A$ are $1.0141$, $-0.6000$, and $-0.4141$. This system is unstable because of the first eigenvalue, so this means there are initial conditions from which the powers diverge.

\begin{verbatim}
>> inv(v)*b
-0.0670
-0.0000
-0.0182
\end{verbatim}

b) The critical SINR threshold level is a function of dominant system eigenvalue. We will assume that matrix $P$ is diagonalizable and that its eigenvalues are ordered by their magnitude when forming $\Lambda$ matrix. Using the property that scaling of any matrix scales its eigenvalues by the same constant, we can derive:

$$A = \alpha \gamma P = \alpha \gamma T \Lambda T^{-1}$$
$$= T \text{diag}(\alpha \gamma \lambda_1, \ldots, \alpha \gamma \lambda_n) T^{-1}$$

For a marginally stable system we need to have $|\alpha \gamma \lambda_1| \leq 1$. Manipulating equation $\alpha \gamma \text{crit}\lambda_1 = 1$, we obtain the critical SINR threshold level,

$$\gamma_{\text{crit}} = \frac{1}{\alpha |\lambda_1|}.$$  

4. Linear dynamical system with constant input. We consider the system $\dot{x} = Ax + b$, with $x(t) \in \mathbb{R}^n$. A vector $x_e$ is an equilibrium point if $0 = Ax_e + b$. (This means that the constant trajectory $x(t) = x_e$ is a solution of $\dot{x} = Ax + b$.)

a) When is there an equilibrium point?

b) When are there multiple equilibrium points?

c) When is there a unique equilibrium point?

d) Now suppose that $x_e$ is an equilibrium point. Define $z(t) = x(t) - x_e$. Show that $\dot{z} = Az$. From this, give a general formula for $x(t)$ (involving $x_e$, $\exp(tA)$, $x(0)$).

e) Show that if all eigenvalues of $A$ have negative real part, then there is exactly one equilibrium point $x_e$, and for any trajectory $x(t)$, we have $x(t) \to x_e$ as $t \to \infty$. 


Solution.

a) An equilibrium point $x_e$ exists if and only if $0 = Ax_e + b$, i.e., $-b = Ax_e$. This happens exactly when $-b \in \text{range}(A)$.

b) If $x_e$ is any equilibrium point, and $z \in \text{null}(A)$, then $x_e + z$ is also an equilibrium point. It follows that in order to have multiple equilibrium points, we need $\text{null}(A) \neq \{0\}$, as well as $b \in \text{range}(A)$.

c) For uniqueness, we need that $\text{null}(A) = \{0\}$, in addition to $b \in \text{range}(A)$. The nullspace condition implies that $A$ is nonsingular. But this means that $\text{range}(A) = \mathbb{R}^n$, so the condition $b \in \text{range}(A)$ holds automatically. In this case, the unique equilibrium point is $x_e = -A^{-1}b$. In summary: there is a unique equilibrium point if and only if $A$ is nonsingular; in this case, we have $x_e = -A^{-1}b$.

d) $z(t) = \exp(tA)z(0)$, so $x(t) = x_e + \exp(tA)(x(0) - x_e)$.

e) Assume that all eigenvalues of $A$ have negative real part. In particular, no eigenvalue can be zero, which means $A$ is nonsingular. Therefore the unique equilibrium point is $x_e = -A^{-1}b$. Since all eigenvalues of $A$ have negative real part, the matrix $\exp(tA)$ goes to zero as $t \to \infty$. From the formula for $x(t)$ above, we see that $x(t)$ converges to $x_e$.

5. Two-point boundary value problem. Consider the system described by $\dot{x} = Ax$, where $A = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$.

a) Find $e^A$.

b) Suppose $x_1(0) = 1$ and $x_2(1) = 2$. Find $x(2)$. (This is called a two-point boundary value problem, since we are given conditions on the state at two time points instead of the usual single initial point.)

Solution.

a) Many methods can be used to find $e^A$. In this case, power series expansion may be the easiest, since $A^k = A^2 = 0$ for all $k \geq 2$: $e^A = I + A = \begin{bmatrix} 0 & 1 \\ -1 & 2 \end{bmatrix}$. 

b) Expanding the equation $x(2) = e^A x(1) = e^{2A} x(0)$ yields

$$
\begin{bmatrix}
  x_1(2) \\
  x_2(2)
\end{bmatrix}
= 
\begin{bmatrix}
  0 & 1 \\
  -1 & 2
\end{bmatrix}
\begin{bmatrix}
  x_1(1) \\
  2
\end{bmatrix}
= 
\begin{bmatrix}
  0 & 1 \\
  -1 & 2
\end{bmatrix}^2
\begin{bmatrix}
  1 \\
  x_2(0)
\end{bmatrix}
= 
\begin{bmatrix}
  -1 & 2 \\
  -2 & 3
\end{bmatrix}
\begin{bmatrix}
  1 \\
  x_2(0)
\end{bmatrix}
= 
\begin{bmatrix}
  2x_2(0) - 1 \\
  3x_2(0) - 2
\end{bmatrix}.
$$

Examining the first line yields $x_1(2) = 2$ and so $x_2(0) = 1.5$; substituting into the last yields $x(2) = [2 \ 2.5]^T$.

6. Linear system with one-bit quantized output. We consider the system

$$
\dot{x} = Ax, \quad y(t) = \text{sign}(cx(t))
$$

where

$$
A = \begin{bmatrix}
  -0.1 & 1 \\
  -1 & 0.1
\end{bmatrix}, \quad c = [1 \ -1],
$$

and the sign function is defined as

$$
\text{sign}(a) = \begin{cases}
  +1 & \text{if } a > 0 \\
  -1 & \text{if } a < 0 \\
  0 & \text{if } a = 0
\end{cases}
$$

Roughly speaking, the output of this autonomous linear system is quantized to one-bit precision. The following outputs are observed:

$$
y(0.4) = +1, \quad y(1.2) = -1, \quad y(2.3) = -1, \quad y(3.8) = +1
$$

What can you say (if anything) about the following:

$$
y(0.7), \quad y(1.8), \quad \text{and } y(3.7)
$$

Your response might be, for example: “$y(0.7)$ is definitely +1, and $y(1.8)$ is definitely −1, but $y(3.7)$ can be anything (i.e., −1, 0, or 1)”. Of course you must fully explain how you arrive at your conclusions. (What we mean by “$y(0.7)$ is definitely +1” is: for any trajectory of the system for which $y(0.4) = +1$, $y(1.2) = −1$, $y(2.3) = −1$, and $y(3.8) = +1$, we also have $y(0.7) = +1$.)

Solution. We know that $x(t) = e^{At} x(0)$, so we have

$$
y(t) = \text{sign}(cx(t)) = \text{sign}(ce^{At} x(0)).
$$
What’s unknown here is the exact value of $x(0)$. The given output data is

$$ce^{A_{0.4}}x(0) > 0, \quad ce^{A_{1.2}}x(0) < 0, \quad ce^{A_{2.3}}x(0) < 0, \quad ce^{A_{3.8}}x(0) > 0.$$ 

These data give us information about $x(0)$. From that information about $x(0)$, we can (maybe) determine information about $y(0.4) = \text{sign} (ce^{A_{0.7}}x(0))$, $y(1.8) = \text{sign} (ce^{A_{1.8}}x(0))$, $y(3.7) = \text{sign} (ce^{A_{3.7}}x(0))$.

Geometrically, a constraint of the form $fx(0) > 0$, where $f \in \mathbb{R}^{1 \times 2}$ is a row vector, defines a halfspace in $\mathbb{R}^2$, with inward normal vector $f^T$. Therefore each of our four measurements of $y$ gives us a halfspace in $\mathbb{R}^2$ that $x(0)$ must lie in. Taking the intersection, we see that $x(0)$ must lie in a cone or sector in $\mathbb{R}^2$. This cone is the set of all possible values of $x(0)$ that are consistent with the measured data. This is shown below.
To determine the sign of $y(t)$ for $t = 0.7, 1.8, \text{ and } 3.7$, we have to check the sign of $ce^{At}x(0)$, for $t = 0.7, 1.8, 3.7$. Geometrically, this means we are checking whether the cone of possible values of $x(0)$ lies on one side, or both sides, of the halfspaces determine by $ce^{At}$, for $t = 0.7, 1.8, 3.7$. This results in $y(0.7) = -1$, and $y(1.8) = -1$, whereas $y(3.7)$ cannot be determined, since the
boundary of the halfplanes lies within the uncertainty region.

The matlab code below solves the problem using the approach just described.

```matlab
A = [-0.1 1; -1 0.1]; c = [1 -1];
M_1 = c*expm(A*0.4); M_2 = c*expm(A*1.2); M_3 = c*expm(A*2.3); M_4 = c*expm(A*3.8);
x = linspace(-10,10); coeff1 = -M_1(1)/M_1(2); y1 = coeff1*x;
coeff2 = -M_2(1)/M_2(2); y2 = coeff2*x; coeff3 = -M_3(1)/M_3(2);
y3 = coeff3*x; coeff4 = -M_4(1)/M_4(2); y4 = coeff4*x;
M_5 = c*expm(A*0.7); coeff5 = -M_5(1)/M_5(2); y5 = coeff5*x;
M_6 = c*expm(A*1.8); coeff6 = -M_6(1)/M_6(2); y6 = coeff6*x;
M_7 = c*expm(A*3.7); coeff7 = -M_7(1)/M_7(2); y7 = coeff7*x;
[a,b] = meshgrid(-10:.5:10,-10:.5:10); figure(1); clf;
subplot(2,2,1); plot(x,y1); grid on; hold on;
for i=1:(size(a,1))^2; if (b(i)<coeff1*a(i)); plot(a(i),b(i),'.'); end; end;
title('Halfplanes for y(0.4)'); subplot(2,2,2); plot(x,y1); grid on; hold on;
for i=1:(size(a,1))^2; if (b(i)<coeff1*a(i)); plot(a(i),b(i),'.'); end; end;
```
```matlab
plot(x,y2,'--');
for i=1:(size(a,1))^2;
    if (b(i)<coeff2*a(i)); plot(a(i),b(i),'+'); end;
end;
title('Adding y(1.2)');
subplot(2,2,3); plot(x(1:50),y1(1:50)); grid on; hold on;
axis([-10 10 -10 10]);
for i=1:(size(a,1))^2;
    if ((b(i)<coeff1*a(i)) & (b(i)<coeff2*a(i)));
        plot(a(i),b(i),'.'); end;
    if (b(i)<coeff3*a(i)); plot(a(i),b(i),'+'); end;
end;
plot(x(51:100),y2(51:100));
plot(x,y3,'--');
title('Adding y(2.3)');
subplot(2,2,4);
plot(x(1:50),y1(1:50)); grid on; hold on;
axis([-10 10 -10 10]);
for i=1:(size(a,1))^2;
    if ((b(i)<coeff1*a(i)) & (b(i)<coeff2*a(i)));
        plot(a(i),b(i),'.'); end;
    if (b(i)>coeff4*a(i)); plot(a(i),b(i),'+'); end;
end;
plot(x(1:50),y4(1:50));
title('Adding y(3.8)');
figure(2); clf
plot(x(1:50),y1(1:50)); grid on; hold on;
axis([-10 10 -10 10]);
for i=1:(size(a,1))^2;
    if ((b(i)<coeff1*a(i)) & (b(i)>coeff4*a(i)));
        plot(a(i),b(i),'.'); end;
end;
plot(x(1:50),y4(1:50)); plot(x,y5,'--');
```
Some of you used another method, which was less geometric but perfectly correct. First we observe that the eigenvalues are \( \pm 0.995j \), so all solutions are periodic, and we have

\[
 ce^{At}x(0) = \alpha \cos(0.995t + \theta),
\]

where \( \alpha \geq 0 \) and \( \theta \) give another parametrization of all possible trajectories. In this problem all that matters in the sign of this signal, which simply changes every \( \pi/0.995 = 3.16 \) seconds. What we need to do is figure out bounds on when the signal zero crossings can be. Evidently the positive to negative crossing takes place between \( t = 0.4 \) and \( t = 1.2 \), and the negative to positive transition occurs between 2.3 and 3.6. Putting this together, using the fact that the two crossings are 3.16 seconds apart, we find the original crossing occurs between 0.4 and 3.8 – 3.16 = 0.64. Hence the negative to positive crossing occurs between 0.4 + 3.16 = 3.56 and 3.8. From these fact we see that \( y(0.7) = -1 \), \( y(1.8) = -1 \), whereas \( y(3.7) \) cannot be determined.

7. **Some basic properties of eigenvalues.** Show the following:

   a) The eigenvalues of \( A \) and \( AT \) are the same.

   b) \( A \) is invertible if and only if \( A \) does not have a zero eigenvalue.

   c) If the eigenvalues of \( A \) are \( \lambda_1, \ldots, \lambda_n \) and \( A \) is invertible, then the eigenvalues of \( A^{-1} \) are \( 1/\lambda_1, \ldots, 1/\lambda_n \).

   d) The eigenvalues of \( A \) and \( T^{-1}AT \) are the same.
**Hint:** you’ll need to use the facts that det $A = \det(A^T)$, det $(AB) = \det A \det B$, and, if $A$ is invertible, det $A^{-1} = 1/\det A$.

**Solution.**

a) The eigenvalues of a matrix $A$ are given by the roots of the polynomial det$(sI - A)$. From determinant properties we know that det$(sI - A) = det(sI - A)^T = det(sI - A^T)$. We conclude that the eigenvalues of $A$ and $A^T$ are the same.

b) First we recall that $A$ is invertible if and only if det$(A) \neq 0$. But det$(A) \neq 0 \iff$ det$(-A) \neq 0$.

i. If 0 is an eigenvalue of $A$, then det$(sI - A) = 0$ when $s = 0$. It follows that det$(-A) = 0$ and thus det$(A) = 0$, and $A$ is not invertible. From this fact we conclude that if $A$ is invertible, then 0 is not an eigenvalue of $A$.

ii. If $A$ is not invertible, then det$(A) = det(-A) = 0$. This means that, for $s = 0$, det$(sI - A) = 0$, and we conclude that in this case 0 must be an eigenvalue of $A$.

From this fact it follows that if 0 is not an eigenvalue of $A$, then $A$ is invertible.

c) From the results of the last item we see that 0 is not an eigenvalue of $A$. Now consider the eigenvalue/eigenvector pair $(\lambda_i, x_i)$ of $A$. This pair satisfies $Ax_i = \lambda_i x_i$. Now, since $A$ is invertible, $\lambda_i$ is invertible. Multiplying both sides by $A^{-1}$ and $\lambda_i^{-1}$ we have $\lambda_i^{-1}x_i = A^{-1}x_i$, and from this we conclude that the eigenvalues of the inverse are the inverse of the eigenvalues.

d) First we note that det$(sI - A) = det(I(sI - A)) = det(T^{-1}T(sI - A))$. Now, from determinant properties, we have det$(T^{-1}T(sI - A)) = det(T^{-1}(sI - A)T)$. But this is also equal to det$(sI - T^{-1}AT)$, and the conclusion is that the eigenvalues of $A$ and $T^{-1}AT$ are the same.

8. **Some matlab exercises.** Consider the continuous-time system $\dot{x} = Ax$ where $A$ can be found in `sys_dynamics_matA.m` and is equal to

$$
A = \begin{bmatrix}
-0.1005 & 1.0939 & 2.0428 & 4.4599 \\
-1.0880 & -0.1444 & 5.9859 & -3.0481 \\
-2.0510 & -5.9709 & -0.1387 & 1.9229 \\
-4.4575 & 3.0753 & -1.8847 & -0.1164
\end{bmatrix}.
$$

a) What are the eigenvalues of $A$? Is the system stable? You can use the command `eig` in matlab.

b) Plot a few trajectories of $x(t)$, i.e., $x_1(t)$, $x_2(t)$, $x_3(t)$ and $x_4(t)$, for a few initial conditions. To do this you can use the matrix exponential command in matlab `expm` (not `exp` which gives the element-by-element exponential of a matrix). Verify that the qualitative behavior of the system is consistent with the eigenvalues you found in part (ii).

c) Find the matrix $Z$ such that $Zx(t)$ gives $x(t + 15)$. Thus, $Z$ is the ‘15 seconds forward predictor matrix’.

15
d) Find the matrix $Y$ such that $Yx(t)$ gives $x(t-20)$. Thus $Y$ reconstructs what the state was 20 seconds ago.

e) Briefly comment on the size of the elements of the matrices $Y$ and $Z$.

f) Find $x(0)$ such that $x(10) = [1 \ 1 \ 1]^T$.

**Solution.**

a) The eigenvalues are $\{-0.1 \pm j5, -0.15 \pm j7\}$. Since all the eigenvalues have negative real part, we conclude that the system is stable.

b) The plots are given in figure ???. We can see that they agree with the eigenvalues found in item (a). These eigenvalues indicate an oscillatory response that is lightly damped, since the imaginary parts are almost two orders of magnitude greater than the real part.

c) This matrix is given by

$$Z = e^{15A} = \begin{bmatrix}
0.2032 & -0.0068 & -0.0552 & -0.0708 \\
0.0340 & 0.0005 & -0.0535 & 0.1069 \\
0.0173 & 0.1227 & 0.0270 & 0.0616 \\
0.0815 & 0.0186 & 0.1151 & 0.1298
\end{bmatrix}$$


d) This matrix is given by

$$Y = e^{-20A} = \begin{bmatrix}
6.2557 & 3.3818 & 1.7034 & 2.2064 \\
-3.3972 & 17.3931 & -1.6257 & -2.8004 \\
-1.7269 & -6.5353 & 10.7081 & 2.9736
\end{bmatrix}$$
e) Since the system is stable, we know that all the components of $e^{At}$ will go to zero as $t \to \infty$. We can notice this already in $Z$: the entries are small, all less than one. This reflects the fact that 15 seconds in the future, the coefficients of the state are, roughly speaking, smaller than the current coefficients. In contrast the entries of $Y$ are all larger than one, which shows that 20 seconds ago the state was ‘larger’ than it is now.

f) It suffices to compute

$$x(0) = e^{-10A} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 3.9961 \\ 1.0650 \\ 3.8114 \\ 1.7021 \end{bmatrix}$$