1. System identification of a linear dynamical system. In system identification, we are given some time series values for a discrete-time input vector signal,

\[ u(1), u(2), \ldots, u(N) \in \mathbb{R}^m, \]

and also a discrete-time state vector signal,

\[ x(1), x(2), \ldots, x(N) \in \mathbb{R}^n, \]

and we are asked to find matrices \( A \in \mathbb{R}^{n \times n} \) and \( B \in \mathbb{R}^{n \times m} \) such that we have

\[ x(t + 1) \approx Ax(t) + Bu(t), \quad t = 1, \ldots, N - 1. \tag{1} \]

We use the symbol \( \approx \) since there may be small measurement errors in the given signal data, so we don’t expect to find matrices \( A \) and \( B \) for which the linear dynamical system equations hold exactly. Let’s give a quantitative measure of how well the linear dynamical system model (1) holds, for a particular choice of matrices \( A \) and \( B \). We define the RMS (root-mean-square) value of the residuals associated with our signal data and a candidate pair of matrices \( A, B \) as

\[
R = \left( \frac{1}{N-1} \sum_{t=1}^{N-1} \| x(t + 1) - Ax(t) - Bu(t) \|^2 \right)^{1/2}.
\]

We define the RMS value of \( x \), over the same period, as

\[
S = \left( \frac{1}{N-1} \sum_{t=1}^{N-1} \| x(t + 1) \|^2 \right)^{1/2}.
\]

We define the normalized residual, denoted \( \rho \), as \( \rho = R/S \). If we have \( \rho = 0.05 \), for example, it means that the state equation (1) holds, roughly speaking, to within 5%. Given the signal data, we will choose the matrices \( A \) and \( B \) to minimize the RMS residual \( R \) (or, equivalently, the normalized residual \( \rho \)).

a) Explain how to do this. Does the method always work? If some conditions have to hold, specify them.

b) Carry out this procedure on the data in lds_sysid.m on the course web site. Give the matrices \( A \) and \( B \), and give the associated value of the normalized residual. Of course you must show your matlab source code and the output it produces.
Solution.

a) Since $S$ is a constant, we can minimize $\rho$ by minimizing

$$R = \left( \frac{1}{N-1} \sum_{t=1}^{N-1} \|x(t+1) - Ax(t) - Bu(t)\|^2 \right)^{1/2}.$$

This is equivalent to minimizing

$$W = \sum_{t=1}^{N-1} \|x(t+1) - Ax(t) - Bu(t)\|^2.$$

Let’s let

$$A = \begin{bmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_n^T \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} b_1^T \\ b_2^T \\ \vdots \\ b_n^T \end{bmatrix}.$$

Then,

$$W = \sum_{t=1}^{N-1} \begin{bmatrix} x_1(t+1) - a_1^T x(t) - b_1^T u(t) \\ x_2(t+1) - a_2^T x(t) - b_2^T u(t) \\ \vdots \\ x_n(t+1) - a_n^T x(t) - b_n^T u(t) \end{bmatrix}^2$$

$$= \sum_{t=1}^{N-1} \sum_{i=1}^n \|x_i(t+1) - a_i^T x(t) - b_i^T u(t)\|^2$$

$$= \sum_{i=1}^n \sum_{t=1}^{N-1} \|x_i(t+1) - a_i^T x(t) - b_i^T u(t)\|^2$$

which shows that $a_i$ and $b_i$ can be chosen independently of $a_j$ and $b_j$ for $i \neq j$. This means that the problem can be separated into $n$ smaller problems each of which solves for a a single row of $A$ and single row of $B$. In other words, we can minimize $W$ by minimizing $\sum_{t=1}^{N-1} \|x_i(t+1) - a_i^T x(t) - b_i^T u(t)\|^2$ for each $i$, $1 \leq i \leq n$. Specifically, for each $i$, $1 \leq i \leq n$, we need find $a_i$ and $b_i$ to minimize

$$\begin{bmatrix} x_i(2) \\ x_i(3) \\ \vdots \\ x_i(N) \end{bmatrix} - \begin{bmatrix} x^T(1) & u^T(1) \\ x^T(2) & u^T(2) \\ \vdots & \vdots \\ x^T(N-1) & u^T(N-1) \end{bmatrix} \begin{bmatrix} a_i \\ b_i \end{bmatrix}.$$

Provided $C$ is skinny and full rank, we can find the desired $z_i$ using least-squares. For each $i$, the optimum $z_i$ is

$$\hat{z}_i = \begin{bmatrix} \hat{a}_i \\ \hat{b}_i \end{bmatrix} = (C^T C)^{-1} C^T y_i.$$
Then, the desired $A$ and $B$ are

$$A = \begin{bmatrix} a_1^T \\ a_2^T \\ \vdots \\ a_n^T \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} b_1^T \\ b_2^T \\ \vdots \\ b_n^T \end{bmatrix}.$$ 

b) The following matlab code implements the solution described above. For the given data,

$$A = \begin{bmatrix} 0.3000 & -0.0800 & -0.3600 \\ 0.9700 & 0.1900 & 0.90 \\ -0.6300 & -0.7600 & 0.73 \end{bmatrix}, \quad B = \begin{bmatrix} 0.9900 & 0.7999 \\ -0.8000 & 0.7700 \\ 0.9000 & 0.7900 \end{bmatrix},$$

$$S = 5.0174, \quad R = 9.4598 \times 10^{-4}, \quad \rho = 1.8854 \times 10^{-4}.$$ 

The small value for $\rho$ indicates that $A$ and $B$ model the given data well.

close all;
clear all;
lds Sysid;
N = size(X,2); % number of vectors given
z = [];
C = [X(:,1:N-1);U(:,1:N-1)];
C = C';
for i = 1:n
    y = X(i,2:N);
y = y';
z = [z C\y];
end
A = z(1:3, :);
A = A';
B = z(4:5, :);
B = B';
X_hat = A*X(:,1:N-1)+B*U(:,1:N-1);
error = X(:,2:N)-X_hat;
R = 0;
S = 0;
for t = 1:N-1
    R = R +norm(error(:,t))^2;
    S = S +norm(X(:,t+1))^2;
end
R = sqrt(1/(N-1)*R);
S = sqrt(1/(N-1)*S);
rho = R/S;
2. A greedy control scheme. Our goal is to choose an input $u : \mathbb{R}_+ \rightarrow \mathbb{R}^m$, that is not too big, and drives the state $x : \mathbb{R}_+ \rightarrow \mathbb{R}^n$ of the system $\dot{x} = Ax + Bu$ to zero quickly. To do this, we will choose $u(t)$, for each $t$, to minimize the quantity

$$\frac{d}{dt} \|x(t)\|^2 + \rho \|u(t)\|^2,$$

where $\rho > 0$ is a given parameter. The first term gives the rate of decrease (if it is negative) of the norm-squared of the state vector; the second term is a penalty for using a large input.

This scheme is greedy because at each instant $t$, $u(t)$ is chosen to minimize the composite objective above, without regard for the effects such an input might have in the future.

a) Show that $u(t)$ can be expressed as $u(t) = Kx(t)$, where $K \in \mathbb{R}^{m \times n}$. Give an explicit formula for $K$. (In other words, the control scheme has the form of a constant linear state feedback.)

b) What are the conditions on $A$, $B$, and $\rho$ under which we have $(d/dt)\|x(t)\|^2 < 0$ whenever $x(t) \neq 0$, using the scheme described above? (In other words, when does this control scheme result in the norm squared of the state always decreasing?)

c) Find an example of a system (i.e., $A$ and $B$), for which the open-loop system $\dot{x} = Ax$ is stable, but the closed-loop system $\dot{x} = Ax + Bu$ (with $u$ as above) is unstable, when $\rho = 1$. Try to find the simplest example you can, and be sure to show us verification that the open-loop system is stable and that the closed-loop system is not. (We will not check this for you. You must explain how to check this, and attach code and associated output.)

Solution.

a) We have

$$\frac{d}{dt} \|x(t)\|^2 + \rho \|u(t)\|^2 = \frac{d}{dt} (x(t)^T x(t)) + \rho \|u(t)\|^2$$

$$= \dot{x}(t)^T x(t) + x(t)^T \dot{x}(t) + \rho \|u(t)\|^2$$

$$= (Ax(t) + Bu(t))^T x(t) + x(t)^T (Ax(t) + Bu(t)) + \rho \|u(t)\|^2$$

$$= x(t)^T (A^T + A)x(t) + 2x(t)^T Bu(t) + \rho \|u(t)\|^2.$$

To minimize this with respect to $u(t)$, we set the gradient equal to zero:

$$2B^T x(t) + 2\rho u(t) = 0.$$

This yields $u(t) = Kx(t)$, with $K = -(1/\rho)B^T$.

For use in part (c), we note that the closed-loop dynamics are given by

$$\dot{x} = Ax + Bu = (A - (1/\rho)BB^T)x.$$

b) With this value of $u(t)$, we have

$$\frac{d}{dt} \|x(t)\|^2 = x(t)^T \left(A^T + A - (2/\rho)BB^T\right) x(t).$$
This is negative for all nonzero \( x(t) \) if and only if the matrix in the quadratic form is negative definite, \textit{i.e.,}

\[
A^T + A - (2/\rho)BB^T < 0.
\]

There are many other ways to write this condition.

c) We can’t find an example with \( n = 1 \), but for \( n = 2 \) we can use

\[
A = \begin{bmatrix}
-0.1 & 1 \\
0 & -0.1
\end{bmatrix}, \quad \rho = 1.
\]

Clearly \( A \) has negative eigenvalues and therefore the open-loop is stable but as the following matlab code shows, the closed-loop system is unstable. The matlab code is

```matlab
A=[-0.1 1; 0 -0.1];
B=[1 -1]’;
rho=1;

eig(A)
eig(A-(1/rho)*B*B’)
```

The output of the above code is

```
ans =
   -0.1000
   -0.1000

ans =
  0.3142
 -2.5142
```