Lab 2 – ECE410F Linear Control Systems
Numerical Linear Algebra and Controllability

Main concepts of this lab

- How to find the basis for the image of a matrix.
- How to find a basis for the sum and intersection of two subspaces.
- How to find coordinates of $v \in \mathbb{R}^n$ in a given basis $\{v^1, \ldots, v^n\}$.
- How to find the matrix representation of an LT $\mathbf{L} : \mathbb{R}^n \to \mathbb{R}^m$ in given bases $\{v^1, \ldots, v^n\}$ and $\{w^1, \ldots, w^m\}$.
- How to check rank, injectivity, surjectivity of LTs.
- How to check whether an equation $Ax = b$ has a solution, whether such solution is unique, and how to find a particular solution.
- How to check A-invariance of a subspace and find the normal form of the representation theorem.
- How to check controllability of a pair (A, B) and perform the Kalman decomposition.

1 Introduction

At the end of the previous lab, you arrived at various representations of a linearized model of a control system, including transfer function form, zero-pole form, and state space form. In this course, our primary focus is on the analysis of linear systems in state space form, which strongly relies on the machinery of linear algebra. In this lab you will review the basic techniques of numerical linear algebra, and then apply those techniques to the analysis of a simple linear system.

Throughout the lab, you will be guided through a number of steps which will require you to write Matlab code. You will write your code in a Matlab script called `labx.m`, where `x` is the lab number. You will submit this code as a group. Your code should provide certain outputs (figures, numbers, and the like). A request for output is highlighted with a shaded area of text, such as the following.

Output 1. Print the eigenvalues of the matrix.

Parts of the text containing directions for the writing of your Matlab code will be highlighted in a different colour, such as this:

Create a function `pendulum.m`. The first line of the function will be

```
function xdot = pendulum(t,x,parameters)
```

2 Submission guidelines and mark breakdown

Marks are assigned to groups. Members of each group get identical marks, unless special circumstances occur. The group lab mark will be made up of three components.

Matlab code	6 pts
Lab report	4 pts
Total	**10 pts**

Matlab code. Your code should be clean and readable, and it should contain abundant commentary, so that the instructors may follow your development and check its correctness. This component of the mark will be based on the correctness of the code and its readability, and it will be assigned as follows:

0 out of 6	the code is absent
1 out of 6	the code is largely incomplete
2 out of 6	the code is largely complete, but there are parts missing and the outputs are incorrect
3 out of 6	the code is complete, but it does not produce correct outputs
4 out of 6	the code is complete, it produces correct outputs, but there is no commentary in the code, and/or the code is poorly organized
5 out of 6	the code is complete, correct, and contains some but insufficient commentary, and/or its organization is below par
6 out of 6	the code is complete, correct, and the commentary and organzation are adequate so that it is easy to read

Lab report. Write a concise report containing the requested outputs, but don't just print out a bunch of matrices and figures. Add some flesh to the outpus that are requested within the lab document so that one can follow the logical development of the lab. Aim for a style like this:

> **Output 1.** The following are the bases of the subspaces $\mathcal{V}$ and $\mathcal{W}$ that were obtained:
> (...)
> We observe that the columns of V were already linearly independent, while those of $\mathcal{W}$ weren't. We further note that (...).

Output 2. Below the solution of the differential equation. There are three figures. The first two figures depict $x_1(t)$ and $x_2(t)$, while the third figure depicts the orbit.

Do not screenshot Matlab figures. Rather, save them as jpeg files (or other formats) and include then in the report in the appropriate order.

When the lab document asks you to comment on something, strive to provide meaningful, insightful commentary, that demonstrates you have understood your own work and how it connects to the course concepts. This portion of the mark will be assigned as follows:

0 out of 4	the report is either absent, or a simple printout of the Matlab outputs without commentary
1 out of 4	the report is incomplete *and* the commentary is inadequate
2 out of 4	the report is incomplete *xor* the commentary is inadequate
3 out of 4	the report is complete and the commentary is adequate, but lacks insight/clear understanding
4 out of 4	the report is complete and the commentary is adequate and demonstrates clear understanding

3 Numerical linear algebra

In this lab, it is assumed that you have read the linear algebra chapter in the textbook, and have reviewed the lecture notes on this subject. If you haven't, this is a good time to do it.

3.1 Basic operations on subspaces

In this section, you will use the following Matlab commands.

Matlab commands

`orth(A)`	Find an orthogonal basis for the column space of A
`null(A)`	Find an orthogonal basis for the null space of A
`inv(A)`	Find the inverse of A

Let $\mathcal{X}$ be an n-dimensional vector space, and let $\mathcal{V}, \mathcal{W}$ be two subspaces of $\mathcal{X}$. Two key operations involving these subspaces are the subspace sum $\mathcal{V} + \mathcal{W}$ and subspace intersection $\mathcal{V} \cap \mathcal{W}$. In this section we will explore how to represent these subspaces numerically, and algorithms for performing these two operations.

Let $A \in \mathbb{R}^{n \times m}$ be a matrix. Recall that the image of A is simply the span of the columns of A

$$\mathrm{Im}(A) = \mathrm{span}\{a^1, \ldots, a^n\}$$

where a^i is the ith column of A. This fact motivates us to use matrices to represent subspaces. In order to computationally convenient and remove redundancy, we will need a procedure to find a basis for the image of a matrix. Matlab's `orth` function does exactly this.

Consider $\mathbb{R}^4$ and the subspaces

$$\mathcal{V} = \text{span}\left\{\begin{bmatrix}1\\-1\\0\\1\end{bmatrix}, \begin{bmatrix}1\\1\\0\\0\end{bmatrix}, \begin{bmatrix}3\\1\\0\\1\end{bmatrix}\right\} \qquad \text{and} \qquad \mathcal{W} = \text{span}\left\{\begin{bmatrix}1\\0\\2\\1\end{bmatrix}, \begin{bmatrix}1\\0\\-2\\0\end{bmatrix}\right\}$$

Using the Matlab `orth` function, begin your `lab2.m` script by finding bases for these subspaces.

Output 1. Print the matrices containing the bases of these subspaces. Comment on any interesting details. Are the given vectors linearly independent? What has happened to the basis for $\mathcal{W}$?

We now want to develop procedures for finding the subspace sum and intersection. First we address the subspace sum. Recall the definition:

$$\mathcal{V} + \mathcal{W} = \{x \in \mathcal{X} : x = v + w,\, v \in \mathcal{V},\, w \in \mathcal{W}\}\,.$$

If we have $\mathcal{V} = \text{Im}(V)$ and $\mathcal{W} = \text{Im}(W)$ for suitable matrices V and W, then

$$\mathcal{V} + \mathcal{W} = \text{Im}\left(\begin{bmatrix}V & W\end{bmatrix}\right).$$

Using this fact, do the following.

Create a function `sub_sum.m`. The first line of the function will be

```
function C = sub_sum(A, B)
```

The input matrices A and B contain bases for the subspaces we wish to sum. The matrix C that is to be returned should contain a basis for the sum of the subspaces.

The procedure for finding the intersection is slightly more involved. Suppose that $\mathcal{V}$ and $\mathcal{W}$ are l- and m-dimensional subspaces respectively, so that $V \in \mathbb{R}^{n\times l}$ and $W \in \mathbb{R}^{n\times m}$. For a vector x to be in $\mathcal{V} \cap \mathcal{W}$, it must be the case that there exist vectors $\lambda \in \mathbb{R}^l, \mu \in \mathbb{R}^m$ such that

$$x = V\lambda = W\mu\,.$$

Rearranging this, we get

$$0 = \begin{bmatrix}V & -W\end{bmatrix}\begin{bmatrix}\lambda\\\mu\end{bmatrix}.$$

For any vector $\begin{bmatrix}\lambda\\\mu\end{bmatrix}$ in the null space of $\begin{bmatrix}V & -W\end{bmatrix}$, the vector $x = V\lambda = W\mu$ will be in $\text{Im}(V) \cap \text{Im}(W)$.

Create a function `sub_intersect.m`. The first line of the function will be

```
function C = sub_intersect(A, B)
```

The input matrices A and B contain bases for the subspaces we wish to intersect. The matrix C that is to be returned should contain a basis for the intersection of the subspaces. You will need to use the Matlab command `null`, which returns a basis for the null space (or kernel) of a given matrix.

Output 2. Return to your `lab2.m` script. Using the functions you developed above, find matrices spanning the sum and intersection of the vector spaces from earlier. Print those matrices.

3.2 Linear transformations and linear equations

Matlab commands	
`rank(A)`	Find the rank of A
`A\b`	Find a particular solution of $Ax = b$

We now turn our attention to the basic operations required for linear transformations and systems of linear equations. The first basic operation we consider is how to perform a change of basis. Let $P = \begin{bmatrix} x^1 & \cdots & x^n \end{bmatrix} \in \mathbb{R}^{n \times n}$ be a matrix whose columns constitute a basis for $\mathbb{R}^n$. We want to express a vector $x \in \mathbb{R}^n$ in terms of this basis. That is, we want coefficients $z_1, \ldots, z_n$ such that $x = z_1 x^1 + \cdots + z_n x^n$, or

$$x = Pz.$$

The vector $z \in \mathbb{R}^n$ represents the coordinates of x in the basis $\{x^1, \ldots, x^n\}$, and we see from the above that z is given by $z = P^{-1}x$.

Output 3. In your `lab2.m` script, consider the basis $\{x^1, x^2\}$ of $\mathbb{R}^2$ given by

$$x^1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \; x^2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}.$$

Find the coordinates z of the vector $x = \begin{bmatrix} 2 & 1 \end{bmatrix}^\top$ in this basis and print it. Then, numerically verify that the identity $x = z_1 x^1 + z_2 x^2$ holds.

Next we turn to matrix representations of linear transformations. For this, review the Procedure to find the matrix representation in Section 2.4 of the textbook. Consider a matrix $A \in \mathbb{R}^{m \times n}$ and

the associated linear transformation $\mathbf{A} : \mathbb{R}^n \to \mathbb{R}^m$ given by $\mathbf{A}(x) = Ax$. Suppose we have matrices $P = \begin{bmatrix} x^1 & \cdots & x^n \end{bmatrix} \in \mathbb{R}^{n\times n}$ and $Q = \begin{bmatrix} y^1 & \cdots & y^m \end{bmatrix} \in \mathbb{R}^{m\times m}$ whose columns constitute bases for $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. To find the matrix representation $\hat{A}$ of the transformation $\mathbf{A}$ in the given bases, we perform the following steps

1. For each $j \in \{1, \ldots, n\}$, we compute $\mathbf{A}(x^j)$, where x^j is the j-th column of P.
2. We express $\mathbf{A}(x^j)$ in the coordinates of the basis $\{y^1, \ldots, y^m\}$, where y^i is the i-th column of Q.
3. The m-vector of coordinates found in point 2 is the j-th column of $\hat{A}$.

Output 4.

- Using the conceptual procedure above, find the expression of the matrix $\hat{A}$ in terms of the matrices A, P, Q. Note that the textbook solves this problem in the special case when $P = Q$ and $m = n$.
- Using your expression for $\hat{A}$, in your `lab2.m` script find the matrix representation of the transformation $\mathbb{R}^4 \to \mathbb{R}^5$ defined as

$$\mathbf{A}(x) = \begin{bmatrix} 1 & 2 & 0 & -1 \\ 0 & 1 & -1 & -2 \\ 1 & 0 & 3 & 4 \\ 0 & -1 & 2 & 3 \\ 0 & 0 & 2 & 2 \end{bmatrix} x \tag{1}$$

in the bases

$$\mathbb{R}^4 = \operatorname{span}\left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \right\}, \quad \mathbb{R}^5 = \operatorname{span}\left\{ \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \right\}.$$

For this part, you might want to review Sections 1.7-1.9 of the textbook. In order to determine the key properties of injectivity and surjectivity of a transformation, we first recall the *rank-nullity theorem* from linear algebra. Recall that this theorem states that for a linear transformation $\mathbf{A} : \mathbb{R}^n \to \mathbb{R}^m$,

$$\operatorname{rank}(\mathbf{A}) + \operatorname{nullity}(\mathbf{A}) = n\,,$$

where the rank of $\mathbf{A}$ is the dimension of the image of $\mathbf{A}$, and the nullity is the dimension of the kernel of $\mathbf{A}$. Since the rank is easy to check computationally, we can use it for simple tests of the injectivity and surjectivity of a transformation.

Output 5. In your `lab2.m` script, consider the matrix A given in (1).

- Print the rank of this matrix.
- Using the rank-nullity theorem, print the dimension of $\mathrm{Ker}(A)$.
- Display simple messages saying whether the matrix is injective, surjective, both, or neither.

  N.B.: please do this *programmatically*. That is, we should be able to replace the above matrix in your code with any other matrix, and the script would still work appropriately.

Finally, consider a system of linear equations $Ax = b, A \in \mathbb{R}^{m\times n}, x \in \mathbb{R}^n, b \in \mathbb{R}^m$. We want to determine if this system has any solutions, and if so if it has just one or many. To do this, recall that the rank of a matrix is by definition the dimension of the column space. In order for the equation $Ax = b$ to have a solution, it is necessary and sufficient that the vector b lie in the column space of A. So by constructing the augmented matrix

$$\begin{bmatrix} A & b \end{bmatrix}$$

and comparing its rank to the rank of A, we can determine if $Ax = b$ has any solutions[1].

The question of whether $Ax = b$ has only one or infinitely many solutions, provided it has at least one, comes down to determining the kernel of A, since if x is a particular solution of $Ax = b$ and y is in the kernel of A, then $A(x + y) = Ax + Ay = b$. So the general solution of $Ax = b$ is $x + y$, where x is a particular solution and y is an arbitrary vector in $\mathrm{Ker}(A)$[2]. We have already seen the Matlab command to find a basis for the kernel of a matrix. If a particular solution of $Ax = b$ exists, in Matlab you can find it using the backslash command, `A\b`.

Output 6. Return to your `lab2.m` script, and consider again the matrix A in (1).

- For each of the following vectors b, determine if $Ax = b$ has no solutions, one solution, or infinitely many solutions.

$$b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad \text{and} \qquad b = \begin{bmatrix} 1 \\ 1 \\ -2 \\ -2 \\ -2 \end{bmatrix}$$

- Print each of the preceding vectors, and indicate the number of solutions. If a solution exists, print it. If infinite solutions exist, provide two different solutions to the equation.

[1] See Theorem 1.34 in the textbook.
[2] See Theorem 1.36 in the textbook

4 A-invariance and Representation Theorem

In this course we make heavy use of the notion of A-invariant subspace. We use it, for instance, to compute Kalman decompositions. In this section you will learn (a) how to check whether a given subspace is A-invariant, and (b) how to numerically compute the coordinate transformation in the Representation Theorem for A-invariant subspace[3], as well as the block upper triangular matrix representation of A in new coordinates.

Testing for A-invariance. Suppose we have a subspace $\mathcal{V} \subset \mathbb{R}^n$ represented by a matrix $V \in \mathbb{R}^{n\times k}$ whose columns are a basis for $\mathcal{V}$, and that we are given a matrix $A \in \mathbb{R}^{n\times n}$. To check if $\mathcal{V}$ is A-invariant, we need to check whether $Av^i \in \mathcal{V}$ for each of the columns v^i of V, or equivalently whether $\mathrm{Im}(AV) \subset \mathrm{Im}(V)$. Numerically, this condition can be easily checked by testing whether $\mathrm{rank}\begin{bmatrix} AV & V \end{bmatrix} = \mathrm{rank}\, V$ (see Section 2.9 of the textbook).

Finding the matrix P of the Representation Theorem. We already have that the columns of V form a basis for the subspace $\mathcal{V}$. In order to find the coordinate transformation $\mathbf{T}(x) = P^{-1}x$ of the Representation Theorem, it only remains to find a basis for a subspace $\mathcal{W}$ that is an independent complement of $\mathcal{V}$, i.e., such that $\mathcal{V} \oplus \mathcal{W} = \mathbb{R}^n$. Equivalently, we need to find a matrix $W \in \mathbb{R}^{n\times(n-k)}$ such that the matrix $P := \begin{bmatrix} V & W \end{bmatrix}$ is invertible. Once this is done, the linear transformation $\mathbf{T}(x) = P^{-1}x$ will guarantee that the matrix representation of A in new coordinates, $\hat{A} = P^{-1}AP$, is block upper triangular.

To find the matrix $W \in \mathbb{R}^{n\times(n-k)}$, we rely on the following observation. Let the columns of W be a basis for $\mathrm{Ker}(V^\top)$, so that $V^\top W = 0$. Then, the columns of V and W are mutually orthogonal vectors in $\mathbb{R}^n$, and one can show (we will not do it) that $\mathrm{Im}(W)$ is an independent complement of $\mathrm{Im}(V)$ (in fact, the *orthogonal complement*). In conclusion, we numerically find W by finding a basis for $\mathrm{Ker}(V^\top)$, which we already know how to do in Matlab.

Output 7. Return to your script `lab2.m`, and consider the matrix

$$A = \begin{bmatrix} 1 & 2 & 2 \\ 1 & -3 & -5 \\ -1 & 2 & 4 \end{bmatrix},$$

and the subspace

$$\mathcal{V} = \mathrm{span}\left\{ \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} \right\}.$$

- Numerically show that $\mathcal{V}$ is A-invariant.
- Find the matrix P of the Representation Theorem, print it out, and verify that $P^{-1}AP$ is block upper triangular.

[3] Theorem 2.46 in the textbook.

5 Controllability and Kalman decomposition

Matlab commands

`ctrb(A,B)` Construct the controllability matrix of the pair (A, B)

For a control system (A, B), the Kalman decomposition for controllability is a straightforward application of the Representation Theorem in the special case when the A-invariant subspace of interest is $\mathrm{Im}(Q_c)$, the image of the *controllability matrix*

$$Q_c := \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix}.$$

The tools we have developed throughout this lab will now allow you to determine if the following system is controllable, and if not, what its Kalman decomposition for controllability is.

Return to your `lab2.m` script, and consider the control system

$$\dot{x} = \begin{bmatrix} 5 & -11 & 5 \\ 0 & -6 & 0 \\ -5 & 5 & -1 \end{bmatrix} x + \begin{bmatrix} 1 & -2 \\ 0 & 0 \\ 1 & 2 \end{bmatrix} u\,.$$

- Use the Matlab command `ctrb` to construct the controllability matrix for this system and check if (A, B) is controllable. You will find that the system isn't controllable. We have seen in class[4] that the subspace $\mathcal{V} = \mathrm{Im}(Q_c)$ is A-invariant, and that $\mathrm{Im}(B) \subset \mathrm{Im}(Q_c)$. The application of the Representation Theorem gives the Kalman decomposition for controllability, and for this you will use the concepts presented in Section 4.
- Find a full-rank matrix V whose columns form a basis for $\mathrm{Im}(Q_c)$.
- Using the concepts of Section 4, find the matrix P of the Representation Theorem.
- Using the coordinate transformation $z = \mathbf{T}(x) = P^{-1}x$, find the Kalman decomposition $(\hat{A}, \hat{B}) = (P^{-1}AP, P^{-1}B)$, and verify that it has the form predicted in Section 5.7 of the textbook.

Output 8.

- Print $(\hat{A}, \hat{B})$, the Kalman decomposition for controllability of the above system.
- Print the controllable and uncontrollable subsystems.