



Lab Booklet

Linear Algebra (University of Melbourne)



THE UNIVERSITY OF

MELBOURNE

School of Mathematics and Statistics

Semester 2, 2018

MAST10007 Linear Algebra

Lab Sheets

Introduction

Objectives

The MATLAB classes introduce a powerful computer package which is widely used in Science, Engineering, Commerce and the Mathematical Sciences.

MATLAB is a commercial software package, designed primarily for performing matrix operations in an easy manner, without writing detailed code as in Python, C++, Java etc. It is widely used in universities and in industry.

The main aims for the computer laboratories are:

- To help students further understand some aspects of the Linear Algebra course.
- To further develop some concepts beyond the level covered in lectures.
- To give some idea of how a computer package can make life easier for the applied scientist.
- To give students a starting point for their future use of MATLAB by helping them develop a working knowledge of some basic commands and the ability to write simple programs.

Contents

1	Computer Lab 1: Introduction to MATLAB	1
2	Computer Lab 2: Row Reduction, Inverses and Determinants	9
3	Computer Lab 3: Determinants, Area and Volume	13
4	Computer Lab 4: Linear Combinations	21
5	Computer Lab 5: Span, Linear Independence and Bases	25
6	Computer Lab 6: The Vector Space \mathbb{F}_2^n	33
7	Computer Lab 7: Hamming Codes	41
8	Computer Lab 8: Inner Products	48
9	Computer Lab 9: Eigenvalues and Eigenvectors	53
10	Computer Lab 10: Linear Transformations	60
11	Computer Lab 11: Test	65

1 Computer Lab 1: Introduction to MATLAB

1.1 Introduction

Today you will be introduced to MATLAB: a powerful computational mathematics program used in science, engineering, commerce and the mathematical sciences all over the world.

While the main strength of MATLAB lies in numerical linear algebra, it can be used to plot functions of one variable, polar plots, surface plots and contour plots for functions of two variables. In addition, some symbolic manipulation is possible.

At several points in the lab, you will be asked to think, discuss, and guess what will happen before you type. Make sure you actually make a guess! It doesn't matter if you are wrong.

1.2 Getting Started with MATLAB

Log in to your machine with your University username and password. Then open MATLAB:

Start menu → All programs → MATLAB → R2015b → MATLAB R2015b

When MATLAB opens, you should see several panes:

Command Window: The largest area is the *Command Window*. You type commands next to the `>>` and the answer will be displayed below the command.

Workspace: The *Workspace* area shows all the variables (named arrays) defined during the MATLAB session. You can double-click on a variable to edit its value.

Command History: The *Command History* area displays commands you have previously used. You can copy and paste commands from this window to the *Command Window*.

Current Folder: This pane shows any files that have been loaded in to MATLAB. We will use this pane in later lab classes.

If some of these panes are not visible in your window, you can show them using the “Layout” button from the MATLAB toolbar.

To see an example of the sort of things MATLAB can do, type

`»vibes`

into the command window. Note that you should not type the `»` from the above line!

Remark: A full description of the features of MATLAB can be found in the online help. To access this, click the ‘?’ icon in the MATLAB toolbar and then click the ‘MATLAB’ link.

1.3 Exercises

Exercise 1: Entering Data

Let's start at the basics: we will enter a variable x , and give it the value 3. Type
`>>x=3`

You have entered your first variable. Notice that it appears in the Workspace window. Now whenever you type x , MATLAB will recognize it is the number 3. Let's enter the following matrix:

$$A = \begin{bmatrix} 1 & 3 \\ 8 & -7 \end{bmatrix}$$

To enter this matrix, we need to type in the information in a way MATLAB understands. So, we need to learn the *syntax* for entering a matrix.

In the command window, type
`>>A = [1 x; 8 -7]`

Several features should be noted:

- ▶ Square brackets start and end the matrix;
- ▶ Spaces separate entries;
- ▶ A semicolon starts a new row.

We could have easily typed 3 instead of x , but this illustrates how MATLAB now recognizes x as being 3. Notice that the variable A has now appeared in the Workspace window.

Try to enter the following matrix into MATLAB

$$B = \begin{bmatrix} 0 & -4 & 7 & 1.87 \\ 5 & 1 & -3 & -3 \\ 276 & 0.4136 & 2 & 1 \end{bmatrix}$$

When we need to use vectors in MATLAB they are entered as column matrices. To enter $v = (1, -6, 9.3, 4)$ enter the matrix

$$v = \begin{bmatrix} 1 \\ -6 \\ 9.3 \\ 4 \end{bmatrix}$$

Finally, MATLAB is designed to deal with *complex numbers* with ease. To enter complex numbers, the syntax is pretty easy.

Type
`>>z = 3+5i`
 to enter the complex number $3 + 5i$.

Exercise 2: Arithmetic Operations

Enter the following matrices into MATLAB:

$$A = \begin{bmatrix} 1 & 4 \\ -2 & 0 \\ 1 & 3 \end{bmatrix}$$

$$B = \begin{bmatrix} -8 & 1 \\ 1 & 9 \\ 0 & 3 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & -1 & 3 \\ 1 & 3 & -4 \\ -1 & -6 & 3 \end{bmatrix}$$

$$D = \begin{bmatrix} 8 & -1 & 2 \\ 0 & 1 & -5 \\ 1 & 8 & 1 \end{bmatrix}$$

MATLAB uses a wide range of operators. We will introduce a few of them here. First, the **addition operator** `+`. This operator will do normal scalar addition, or matrix addition depending what variables you are trying to add.

(a) **Adding A and B .**

To remind ourselves of what A and B are, first type `A`, then type `B`.

See that MATLAB shows you their values. Alternatively you can double click on the grid to the left of either A or B in the *Workspace* to get a new window showing their entries.

Two matrices that have the same size can be added together by simply adding the corresponding entries.

First do the calculation by hand:

$$A + B =$$

Now add them on MATLAB by typing

`>A+B`

(b) **The multiplication operator `*`.** Just like the addition operator, it will automatically do normal, scalar, or matrix multiplication depending on what variables you are trying to multiply. It doesn't matter if you haven't seen matrix multiplication before; one of the purposes of this lab is to introduce the idea that matrices can (sometimes) be multiplied together.

Use this operator to find (i) $3C$ (ii) CD (iii) DC .

(c)

Type `A'`. What did it do?

What would you guess this operator is called?

Try using it on some other matrices and vectors.

Exercise 3: A Thinking Point

(a)

Is it possible to multiply the matrices A and C (in that order)?

What do you think MATLAB will do if you try to multiply them?

Try it!!

(b)

Is it possible to add the matrix A to the scalar 2?

What do you think MATLAB will do if you try to add them?

Try it. Did it give an error? What did MATLAB actually do?

The lesson is to always know how an operator works, and not to assume it will do exactly the same thing that we would expect mathematically.

Exercise 4: Correcting Data

Input the matrix $B = \begin{bmatrix} -8 & 1 \\ 1 & 9 \\ 0 & 3 \end{bmatrix}$

Then enter

`>3*B`

Suppose that you meant to enter

$$B = \begin{bmatrix} -8 & 73 \\ 1 & 9 \\ 0 & 3 \end{bmatrix}$$

(a) You can correct the wrong entry by using the up arrow:

(i) Press \uparrow .

(ii) Highlight 1 in the *Command Window* and replace it with 73.

(iii) Press ENTER

(b) To correct the entry and find $3 * B$ in one go:

(i) Select the commands `B=[-8 1; 1 9; 0 3]` and `3*B` in the *Command History*.

(ii) Copy and paste both commands to the *Command Window*.

(iii) Change the entry in B and press ENTER to execute both commands at the same time.

(c) To change B back to the original matrix,

$$B = \begin{bmatrix} -8 & 1 \\ 1 & 9 \\ 0 & 3 \end{bmatrix}$$

```
type in
    »B(1,2)=1
The B(1,2) refers to the 1st row and 2nd column of B.
```

(d) Look in the *Workspace* pane to see the matrices and variables we have entered so far. Double-click on the entry for B to open it in a spreadsheet.

```
Highlight the Row 2, Col 1 entry. Type in 14 and hit return.
You have changed one entry in B.
```

Exercise 5: Extracting Entries from Matrices

We can **find the value of a specific entry** in the matrix using row-column notation.

```
Type
    »B(1,2)
```

Notice that it tells us the entry in the first row, and second column. This can be very useful when we are dealing with incredibly large matrices. Also notice that since we didn't assign a variable to the answer, MATLAB automatically assigns it to 'ans'. It will overwrite ans next time, so always assign to a variable if you want to use it later.

We can also **extract a whole column**.

```
Type v=B(:,2)
```

Important features are:

- ▶ the colon picks all rows
- ▶ the 2 picks the 2nd column

```
What would you need to type to extract the 3rd row? Try it out.
```

Exercise 6: Data Entry Shortcuts

(a) A useful operator is the **colon operator**.

```
Type in the following commands and figure out what it does.
```

```
(i) 2:8      (ii) 3:0.5:6
```

```
Investigate further by trying your own combinations. When you think you've figured it out, use the
colon operator to create the vector [1 1.2 1.4 1.6 1.8 2.0 2.2 2.4]
```


(b) Just like in mathematics, a **function** takes one or several **arguments**, and outputs a **result**. The arguments can be all sorts of things: scalars, matrices, even files. Similarly, the outputs can be things like graphs, files, scalars and more.

Functions in MATLAB all use a common syntax, very similar to the way we use function in mathematics. The name of the function goes first, followed immediately by round brackets. Inside the brackets go the arguments, separated by commas. The arguments must go in the right order.

Lets begin with some basic functions which create new matrices.

Type `eye(4)`. What does this function do?

Type `ones(6,3)`. What does this function do? What do the two arguments represent?

Type `zeros(4,1)`. What does this function do?

This time our function will take a vector as its argument. Type in a random vector and call it `v`. Now type `diag(v)`. What does this function do?

Type `[zeros(3,2) ones(3,5)]` to see what happens. What about `[zeros(2,3); ones(5,3)]`?

(c) *Challenge!!* Enter the following 12×12 matrix using the least number of possible keystrokes **without** using the *Workspace*. That is, use the functions from above to get this matrix, rather than typing in each entry individually.

$$Q = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 & 7 & 7 & 7 & 7 & 7 & 7 & 7 \\ 0 & 8 & 0 & 0 & 0 & 7 & 7 & 7 & 7 & 7 & 7 & 7 \\ 0 & 0 & 8 & 0 & 0 & 7 & 7 & 7 & 7 & 7 & 7 & 7 \\ 0 & 0 & 0 & 11 & 0 & 7 & 7 & 7 & 7 & 7 & 7 & 7 \\ 0 & 0 & 0 & 0 & 1 & 7 & 7 & 7 & 7 & 7 & 7 & 7 \\ 12 & 12 & 12 & 12 & 12 & 1 & 7 & 7 & 7 & 7 & 7 & 7 \\ 12 & 12 & 12 & 12 & 12 & 7 & 1 & 7 & 7 & 7 & 7 & 7 \\ 12 & 12 & 12 & 12 & 12 & 7 & 7 & 1 & 7 & 7 & 7 & 7 \\ 12 & 12 & 12 & 12 & 12 & 7 & 7 & 7 & 1 & 7 & 7 & 7 \\ 12 & 12 & 12 & 12 & 12 & 7 & 7 & 7 & 7 & 1 & 7 & 7 \\ 12 & 12 & 12 & 12 & 12 & 7 & 7 & 7 & 7 & 7 & 1 & 7 \\ 12 & 12 & 12 & 12 & 12 & 7 & 7 & 7 & 7 & 7 & 7 & 1 \end{bmatrix}$$

1.4 Why MATLAB?

In this subject we will only use a small portion of the features of MATLAB. This last part of the first lab is designed to give you some idea of the possibilities of the program. While some of the graphics in the demonstration files may seem simple, they are the product of quite sophisticated matrix manipulation.

Instead of writing all your commands in the MATLAB window, it is possible to prepare them in a file. This is called an *m-file* or *m-script* (so called because each file has the extension *.m*). You will learn how to write these files later in the semester. The following demonstrations are all stored in the form of m-files.

We first consider the program *xpbombs*. Try entering
`>help xpbombs`

The result is a description of how to play the game.

Now enter
`>xpbombs`
Try clicking on one of the squares to find out what happens. The aim is to work out which squares contain bombs and put a flag on them, without detonating any of the bombs.

Some of the other demo programs that you may like to try are listed below.

First use
`> help programname`
to find out about the program, and then run it.

fifteen life travel xpklein xpsound graf2d2 peaks imagedemo

2 Computer Lab 2: Row Reduction, Inverses and Determinants

2.1 Introduction

One of the main strengths of MATLAB is that it provides an excellent tool for working with large matrices. This week's tutorial introduces some useful MATLAB commands for doing this. You will see how MATLAB is used for solving systems of linear equations and finding the inverse of matrices. In addition, you will be using some special programs that have been designed to help you investigate the use of row operations for finding matrix inverses. Finally, you will be asked to investigate one of these special programs, and use it as a basis for your first programming task.

2.2 Exercises

Exercise 1: Row Reduction

The most basic command for doing row operations is the reduced row echelon form function:

$$R = \text{rref}(A)$$

The result is the reduced row echelon form of the matrix A , which you have seen in lectures.

(a)

Use the `rref` command to calculate the reduced row-echelon form of the following matrices.

$$(i) \quad A = \begin{bmatrix} -1 & 2 & 1 & -3 & 1 \\ 1 & 0 & 1 & -1 & 3 \\ 3 & 1 & 0 & 3 & -1 \end{bmatrix} \quad \text{rref}(A) =$$

$$(ii) \quad B = \begin{bmatrix} 4 & 4 & -4 & -2 & 5 & -5 \\ -1 & -5 & 9 & -2 & -5 & 14 \\ 12 & 0 & 12 & 13 & 9 & -1 \\ -7 & 0 & -7 & 11 & 14 & -14 \end{bmatrix} \quad \text{rref}(B) =$$

Notice that MATLAB has absolutely no difficulty dealing with awkward decimals, unlike humans. In real world applications, you are more than likely to get ugly numbers, and also very large matrices, so software like MATLAB can be an immense time saver.

(b)

Solve, with the help of the `rref` command, the following system of equations:

$$\begin{aligned} 4x_1 + 4x_2 - 4x_3 - 2x_4 + 5x_5 &= -5 \\ -x_1 - 5x_2 + 9x_3 - 2x_4 - 5x_5 &= 14 \\ 12x_1 + 12x_3 + 13x_4 + 9x_5 &= -1 \\ -7x_1 - 7x_3 + 11x_4 + 14x_5 &= -14 \end{aligned}$$

$$\text{Answer: } \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} =$$

Hint: MATLAB has already done some of the work for us in part (a), but you must do the rest by hand.

Exercise 2: Inverses and Determinants

The command `inv(A)` finds the inverse of a matrix and `det(A)` finds the determinant of a matrix.

Can we find the inverse of the above matrix A ? Why or why not?

Try to get MATLAB to do it. What does it say?

Make up your own square matrix, B say, and find its inverse and determinant.

How could we check to make sure MATLAB found the inverse correctly?

What do you expect the $\det(B^{-1})$ to be?

Use MATLAB to confirm your answer.

Exercise 3: Ways to find Inverse Matrices

In this exercise we will be working with the following matrix:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 3 & 0 \\ 1 & 0 & 0 & 1 \\ 2 & 1 & 0 & -3 \end{bmatrix}$$

We will find A^{-1} in three ways:

(a)

Use the command `inv`.

The inverse is $A^{-1} =$

(b)

Create an augmented matrix M which has the matrix A on the left, and the identity on the right. You should try to use the function you learnt last week that creates identity matrices quickly. Now find the reduced row echelon form of this augmented matrix.

How will this tell us the inverse? Did it work?

(c) We now use MATLAB to mirror the process of finding inverse matrices by hand. Before doing so we need to locate the programs that will be used in this process.

Locating and Opening Script Files (aka m-files)

MATLAB can execute a sequence of commands stored in a computer file. Such files are stored in script files called **m-files** (so called because the file name is always *filename.m*). We will locate the program files for this week, copy them to the desktop and make them available to MATLAB.

You will need to remember the following instructions for your future classes.

- ▶ Make sure the *Current Folder* pane is showing in MATLAB.
(If it is not, use the 'Layout' menu on the MATLAB toolbar to add it.)
- ▶ Go to:
Start menu → Computer → Lab-Materials (L:) → MAST10007 Linear Algebra
- ▶ Drag the folder called *10007sheet2* directly into the MATLAB *Current Folder* pane
- ▶ Right-click on the *10007sheet2* folder in MATLAB's *Current Folder* pane, go to "Add to Path" and choose "Selected Folders and Subfolders".

The m-files in the folder *10007sheet2* are now available to use in MATLAB.

We will use the following functions defined in the m-files. Try them out using the matrix A entered already.

rowswap.m

Enter

```
» [B,E]=rowswap(A,i,j)
```

(choosing some numerical values for i and j , eg. $i = 2, j = 4$).

This swaps row i with row j in matrix A .

Output: B will be matrix A with its rows swapped, and E will be the corresponding elementary matrix, that is, E is the identity matrix with rows i and j swapped.

rowmul.m

Enter

```
» [B,E]=rowmul(A,c,i)
```

(choosing numerical values for c and i).

This multiplies row i of A by c .

Output: B will be matrix A with row i multiplied by c , and E will be the corresponding elementary matrix, that is, the identity matrix with rows i multiplied by c .

rowadd.m

Enter

```
» [B,E]=rowadd(A,c,i,j)
```

(again choosing numerical values for c, i and j).

This adds $c \times \text{row } i$ to row j of matrix A .

Output: B will be matrix A with $c \times \text{row } i$ added to row j , and E will be the corresponding elementary matrix, that is, the identity matrix with $c \times \text{row } i$ added to row j .

Use these functions to row reduce the matrix M defined above. Take great care with your operations. Call the new matrices M_1, M_2, \dots , and the elementary matrices E_1, E_2, \dots . For example, you could start by doing:

```
» [M1,E1]=rowswap(M,1,3)
```

and then perhaps:

```
» [M2,E2]=rowadd(M1,-2,1,4) Note the M1 here!
```

We will be using the matrices E_1, E_2, \dots in part (d).

Remember, if you make a mistake, you can use \uparrow or the *Command History* pane to retrieve and edit previous commands.

(d)

Check that the inverse equals the product of the elementary matrices. Remember that the order in which you multiply them is important.

Exercise 4: Simple Programming

For this exercise, you will have an opportunity to look at the inner workings of the `rowswap` function.

From the toolbar at the top of the MATLAB window, click Open, and navigate to *My Documents* → *MATLAB* → *10007sheet2*. Open the file *rowswap.m*. It will appear in a new pane called *Editor*.

(a) Analysing the program

Analyse each line of the *rowswap.m* and write down what you think it does.

Line 1	Declares the function and tells MATLAB the outputs and input of the function.
Line 2	Comment. [†]
Line 3	Comment.
Line 4	
Line 5	
Line 6	
Line 7	
Line 8	
Line 9	
Line 10	
Line 11	

[†] Comments are used to make notes about the program.

MATLAB ignores the content of lines to the right of the symbol %.

(b) Changing the program

First save the new file as `colswap.m`

Modify the `rowswap` function, and turn it into a `colswap` function that swaps two columns instead of two rows. To do this, you will need to figure out currently how the function works, and make appropriate changes.

Remember to try your new function out on a matrix. If it doesn't work as it should (and don't worry if it doesn't work first time) you will need to *debug* your function: this means going through it to figure out where the mistake is.

When you've got it to work, pat yourself on the back, and call a tutor over to bask in their praise.

3 Computer Lab 3: Determinants, Area and Volume

3.1 Introduction

In this class we will further investigate the determinant of a matrix and use some of the graphics capabilities of MATLAB.

A fundamental property of determinants is that they are unchanged by the elementary row operation of adding a multiple of one row to another row. We will make use of this property to investigate how determinants relate to areas and volumes. MATLAB allows us to explore this from graphical, numerical and symbolic perspectives.

Before starting the exercises, start MATLAB and load this week's m-files:

- ▶ Make sure the *Current Folder* pane is showing in MATLAB.
(If it is not, use the 'Layout' menu on the MATLAB toolbar to add it.)
- ▶ Go to:
Start menu → Computer → Lab-Materials (L:) → MAST10007 Linear Algebra
- ▶ Drag the folder called *10007sheet3* directly into the MATLAB *Current Folder* pane
- ▶ Right-click on the *10007sheet3* folder in MATLAB's *Current Folder* pane, go to "Add to Path" and choose "Selected Folders and Subfolders".

3.2 Basic Commands

The table below lists some commands that you will need today plus others that relate to the material in lectures.

To complete the table, you will need to enter the following matrices into MATLAB:

$$A = \begin{bmatrix} 2 & 1 \\ 3 & -1 \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} 2 \\ -2 \\ 1 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} 3 \\ -1 \\ 2 \end{bmatrix}$$

$$\mathbf{u} = \begin{bmatrix} 2 & 3 \end{bmatrix} \quad \mathbf{v} = \begin{bmatrix} 1 & -1 \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} -3 & 1 \end{bmatrix} \quad \mathbf{z} = \begin{bmatrix} -2 & 1 \end{bmatrix}$$

Then use the command $B=[\mathbf{u};\mathbf{v}]$ and $C=[\mathbf{w};\mathbf{z}]$ to obtain two 2×2 matrices.

Complete the following table:

Command	Meaning	Result of calculation
<code>det(A)</code>	The determinant of A	
<code>dot(x,y)</code>	The dot product: $\mathbf{x} \cdot \mathbf{y}$	
<code>cross(x,y)</code>	The cross product $\mathbf{x} \times \mathbf{y}$	
<code>pgram1(B)</code>	Draws the parallelogram with \mathbf{u} & \mathbf{v} on two sides where B has the vectors \mathbf{u} & \mathbf{v} in its rows.	
<code>pgram2(B,C)</code>	Draws two parallelograms, the first in blue bounded by \mathbf{u} & \mathbf{v} from the rows of B and the second in red bounded by \mathbf{w} & \mathbf{z} from the rows of the matrix C .	

As well as the commands above you may find the following useful. (Do not test them now.)

Command	Meaning
<code>hold on</code>	Tells MATLAB to draw the next graph on the same set of axes as the previous graph.
<code>hold off</code>	Tells MATLAB to start a new graph deleting the previous graph.
<code>figure(2)</code>	Tells MATLAB to start a new graph called Figure 2. Does not delete Figure 1. Can have figure(3) etc.

Reminder You will be using commands more than once in this Lab. Remember that in MATLAB you can use the up arrow to select a previous command, changing some of it if necessary. Also you can copy and paste from the command history window to the command window.

3.3 Exercises

Exercise 1: Graphical investigation: the area of parallelograms

- (a) Let $\mathbf{u} = [4 \ 2]$, $\mathbf{v} = [2 \ 3]$ and $A = \begin{bmatrix} 4 & 2 \\ 2 & 3 \end{bmatrix}$ ($A=[\mathbf{u};\mathbf{v}]$). Use the command `det(A)` to find the determinant of A . Check your answer by hand.

$\det(A) =$

Use `pgram1(A)` to draw the parallelogram bounded by \mathbf{u} and \mathbf{v} .

- (b) Now find $\mathbf{w} = \mathbf{v} - \alpha\mathbf{u}$ with $\alpha = 1.5$.

Put \mathbf{u} and \mathbf{w} into the rows of a matrix B .

What row operation was used to obtain B from A ?

What is the determinant of B ? $\det(B) =$

How does it compare to the determinant of A ?

Use `pgram2(A,B)` to draw the parallelograms bounded by \mathbf{u} and \mathbf{v} and by \mathbf{u} and \mathbf{w} .

Thinking of \mathbf{u} as the base of the parallelograms what do you notice about the height and hence the areas of the two parallelograms?

- (c) Repeat part(b) with three other values of α in the range -2 to 2 .

α	B	$\det B$

What do you notice about the determinants of all the matrices found?

What do you notice about the areas of the various parallelograms?

- (d) Repeat the whole exercise, this time leaving \mathbf{v} as fixed and replacing \mathbf{u} in matrix A by $\mathbf{z} = \mathbf{u} - \beta\mathbf{v}$ choosing two or three different values of β in the range -2 to 2 .

β	B	$\det B$

What do you notice about the determinants of all the matrices found?

What do you notice about the areas of the various parallelograms?

- (e) Finally start with the matrix A and find the two matrices B and C as follows:

To obtain B replace row 2 of matrix A with row 2 $- \alpha$ (row 1) where α is chosen to give a zero as the entry $B(2, 1)$ in row 2 and column 1.

To obtain C replace row 1 of matrix B with row 1 $- \beta$ (row 2) where β is chosen to give a zero in $C(1, 2)$. C should be diagonal.

Find the determinants of A , B and C .

$\det(A) =$ $\det(B) =$ $\det(C) =$

What do you notice about these determinants?

In MATLAB enter `pgram2(A,B);figure(2);pgram2(B,C)` and study the resulting figures carefully.

What do you notice about the areas of the various parallelograms?

What is the area of the parallelogram spanned by the vectors \mathbf{u} and \mathbf{v} ?

How does the determinant of A relate to this area?

- (f) If you swap the rows of A how does this affect the area spanned by the vectors in the rows of A ? How does it affect the determinant?

- (g) Based on your experiment above, develop a working hypothesis as to the relationship between determinant of 2×2 matrices and the area of the parallelogram spanned by the vectors in the rows of the matrix.

Exercise 2: Symbolic investigation: verifications with the help of MATLAB

Choose *Clear Workspace* from the toolbar so that all previous vectors and matrices are deleted.

The symbolic toolbox provided as part of MATLAB enables you to do symbolic calculations as well as numeric calculations. In order to use this toolbox you first need to specify some variables as being symbolic. This is illustrated in the following.

Enter

```
»syms s r x y x1 x2 y1 y2 real
```

This declares these variables to be symbolic rather than numeric.

Try the following commands, noticing that the result of your entries are formulae.

Command	Result of calculation
<code>x= [x1 x2]</code>	
<code>y= [y1 y2]</code>	
<code>s^2</code>	
<code>s+r</code>	
<code>s*r</code>	

Command	Result of calculation
<code>dot(x,y)</code>	
<code>abs(-10)</code>	
<code>sqrt(2)</code>	
<code>simplify((s+r)*(s-r)-s^2+r^2)</code>	
simplifies a symbolic formula	

Area of a parallelogram in \mathbb{R}^2

In this section we show how one can use MATLAB to do algebraic calculations. We will prove the hypothesis from Exercise 1, that is,

$$\text{In } \mathbb{R}^2 : |\det(A)| = \text{area of parallelogram spanned by the vectors } \mathbf{x} \text{ and } \mathbf{y} \quad (1)$$

where $\mathbf{x} = (x_1, x_2)$, $\mathbf{y} = (y_1, y_2)$ and $A = \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \end{bmatrix}$.

1. First set symbolic variables in MATLAB by

```
»syms x1 x2 y1 y2 real
```

2. Then, in MATLAB define $\mathbf{x} = [x_1 \ x_2]$ and similarly for \mathbf{y} and $A = [\mathbf{x}; \mathbf{y}]$.

3. Write down the determinant of A without using MATLAB.

4. Try the command

```
» d=det(A)
```

5. Show that $\mathbf{w} = (x_2, -x_1)$ is perpendicular to $\mathbf{x} = (x_1, x_2)$ using MATLAB.

```
» dot(x,w)
```

6. Draw (by hand) a parallelogram P determined by vectors \mathbf{x} and \mathbf{y} , and add the vector \mathbf{w} perpendicular to \mathbf{x} (draw \mathbf{x} horizontal). Indicate the projection of \mathbf{y} onto \mathbf{w} .

7. Observe that the magnitude of the projection of \mathbf{y} onto \mathbf{w} is the height of the parallelogram P . The magnitude of the projection is given by:

$$\left| \frac{\mathbf{w} \cdot \mathbf{y}}{\|\mathbf{w}\|} \right|$$

8. Calculate this in MATLAB by typing

```
»height=abs(dot(w,y)/sqrt(dot(w,w)))
```

This gives an expression for the height of the parallelogram.

9. The base of the parallelogram is the length of \mathbf{x}

```
»base=sqrt(dot(x,x))
```

10. The area of the parallelogram is base times height. Calculate this in MATLAB and check that it is the same (up to sign) as the determinant of A .

$$\begin{array}{rcl} \det(A) & = & \\ \text{base} \times \text{height} & = & \end{array}$$

Thus equation (1) is proved.

Exercise 3: Volume of a parallelepiped in \mathbb{R}^3

Choose *Clear Workspace* from the toolbar so that all previous vectors and matrices are deleted.

Let $\mathbf{u} = (3, 0, 0)$, $\mathbf{v} = (0, -2, 0)$ and $\mathbf{w} = (0, 0, 2)$.

- (a) Enter \mathbf{u} , \mathbf{v} and \mathbf{w} into MATLAB.

$$\text{Let } A = \begin{bmatrix} 3 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 2 \end{bmatrix}.$$

To enter A into MATLAB use the command $\mathbf{A}=[\mathbf{u};\mathbf{v};\mathbf{w}]$.

To draw the parallelepiped spanned by the vectors that make up the rows of A try the command `pped1(A)`

What is the volume of this parallelepiped?

What is the determinant of A ?

- (b) Add any multiple of \mathbf{u} to \mathbf{w} to get a new vector \mathbf{x} . Put \mathbf{u} , \mathbf{v} and \mathbf{x} into the rows of a matrix B .

What is $\det(B)$?

$\det(B) =$

Try the command `ppped2(A,B)`. If you click on \odot at the top of the figure window, you can use the mouse to rotate the figure to see it more clearly.

Notice that the base and height of A and B are the same. What do you conclude about the volumes of the two parallelepipeds?

How do the volumes relate to the determinants?

Try adding a multiple of \mathbf{v} to \mathbf{w} to get \mathbf{x} and again draw the parallelepipeds and find the determinants.

Use a few other multiples of \mathbf{u} and \mathbf{v} and see what happens. What do you notice?

- (c) Try other elementary row operations on A where you either

(1) add a multiple of one row to another, or

(2) you swap rows

using `ppped2(A,B)`, `det(A)` and `det(B)` to estimate the effect on volume at each stage and to find the determinants.

What do you notice?

- (d) Use a row operation of type (1) on A , changing row 3 to get a new matrix $A1$. For example, the commands:

```
A1=A; A1(3,:)= A(3,:)-2* A(2,:);A1
```

replace row 3 by row 3 $-2 \times$ (row 2). Then `ppped2(A,A1)` will enable you to compare volumes.

Use row operation (1) on $A1$ changing row 2 to get $A2$. Draw the parallelepipeds to compare volumes.

Use row operation (1) on $A2$ changing row 1 to get $A3$. Draw the parallelepipeds to compare volumes.

Use one last row operation of type (1) on $A3$ to get $A4$.

What do you conclude about the volumes of A and $A4$?

How do these compare to the determinants of A and $A4$?

- (e) Based on your experiments, develop a working hypothesis as to the relationship between determinant of 3×3 matrices and the volume of the parallelepiped spanned by the vectors in the rows of the matrix.

Note that your experiment does not constitute a proof. You need to use methods such as those in Exercise 2 for a rigorous proof.

4 Computer Lab 4: Linear Combinations

4.1 Introduction

Today we will be investigating linear combinations, linear (in)dependence and span. We will be using some MATLAB graphics to help visualize these concepts in 3 dimensions, but all of the ideas apply in higher dimensions as well.

Before starting the exercises, start MATLAB and load this week's m-files:

- ▶ Make sure the *Current Folder* pane is showing in MATLAB.
(If it is not, use the 'Layout' menu on the MATLAB toolbar to add it.)
- ▶ Go to:
Start menu → Computer → Lab-Materials (L:) → MAST10007 Linear Algebra
- ▶ Drag the folder called *10007sheet4* directly into the MATLAB *Current Folder* pane
- ▶ Right-click on the *10007sheet4* folder in MATLAB's *Current Folder* pane, go to "Add to Path" and choose "Selected Folders and Subfolders".

4.2 Exercises

Exercise 1: Linear combinations of two vectors

(a) First, let's go over the definition of a *linear combination* of two vectors. You might need to ask your tutor if you're unsure.

Complete the following statement:

The vector \mathbf{w} is a linear combination of the vectors $\{\mathbf{u}, \mathbf{v}\}$ if

$$\mathbf{w} = \quad \quad \quad \text{for some } a, b \in \mathbb{R}.$$

(b) Enter `linspan` in MATLAB to obtain a GUI (Graphical User Interface) which will be used to help us visualize some linear combinations.

In the GUI enter the vectors $\mathbf{u} = (1, 2, 1)$ and $\mathbf{v} = (2, -1, 0)$

Then click the 'Plot u and v' button.

The vectors appear in the graph on the left. You can click and drag on the graph to rotate it in 3D.

(c) The span of two vectors is the collection of all linear combinations of the two vectors.

Complete the following statement:

The span of two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^3$ is the set of vectors:

$$\{\mathbf{w} \in \mathbb{R}^3 \mid \quad \quad \quad \}$$

(d)

Write down a geometric description of the span of two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^3$.

(e)

Press the ‘Plot Linear Combinations’ button. The computer calculates 20 random linear combinations of \mathbf{u} and \mathbf{v} , and graphs them. Click and drag on the graph to rotate it in 3D. Do you get what you described in (d)?

(f)

Click on the ‘Plot Span’ button. This should confirm your answers from above.

Try some other vectors to see what happens.

(g)

Is it possible to find two vectors whose span is a plane that does not pass through the origin? If not, why not. If so, find two vectors that achieve this.

Exercise 2: Linear dependence and independence

Two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^3$ are *linearly dependent* if there exist $a, b \in \mathbb{R}$, not both zero, such that $a\mathbf{u} + b\mathbf{v} = \mathbf{0}$. That is, they are linearly dependent if there is a non-trivial linear combination that gives the zero vector. They are *linearly independent* if they are not linearly dependent.

(a)

In geometric terms, what does it mean for a set of two vectors to be linearly dependent?

How about linearly independent?

(b)

Write down a pair of vectors in \mathbb{R}^3 that are linearly independent and one pair that are linearly dependent. Use the GUI to check your answers by giving a geometric description of the span in each case.

		span
independent pair:		
dependent pair:		

Exercise 3: Linear combinations of three vectors in \mathbb{R}^3

(a)

Complete the following statements:

- The span of a set of vectors $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{R}^3$ is given by

$$\{\mathbf{x} \in \mathbb{R}^3 \mid \quad \quad \quad\}$$

- Three vectors $\mathbf{u}, \mathbf{v}, \mathbf{w}$ are linearly dependent if

(b)

In the GUI enter the vectors: $\mathbf{u} = (1, 2, 1)$, $\mathbf{v} = (2, -1, 0)$, $\mathbf{w} = (1, 3, 3)$.

Is $\mathbf{w} \in \text{span}\{\mathbf{u}, \mathbf{v}\}$?

Is the set $\{\mathbf{u}, \mathbf{v}, \mathbf{w}\}$ linearly dependent or independent?

How would you describe $\text{span}\{\mathbf{u}, \mathbf{v}, \mathbf{w}\}$?

(c)

Starting again, use the same \mathbf{u} and \mathbf{v} , but with $\mathbf{w} = (-4, 7, 2)$.

Is this new set $\{\mathbf{u}, \mathbf{v}, \mathbf{w}\}$ linearly dependent or independent?

How would you describe $\text{span}\{\mathbf{u}, \mathbf{v}, \mathbf{w}\}$ in this case?

How does it compare to $\text{span}\{\mathbf{u}, \mathbf{v}\}$?

(d)

Can you think of three vectors whose span is a straight line?

Try graphing them in the GUI.

Exercise 4: Using matrices to determine dependence

Whether or not three vectors in \mathbb{R}^3 are linearly independent can be determined by looking at the determinant of a corresponding matrix.

(a)

In MATLAB enter a matrix A whose columns are given by the following three vectors:

$$\mathbf{u} = (-1, -2, 3), \mathbf{v} = (5, 6, -1), \mathbf{w} = (3, 2, 1).$$

(b) Remember from last week that the determinant of A is equal to the absolute value of the volume of the parallelepiped defined by \mathbf{u} , \mathbf{v} and \mathbf{w} . If the three vectors lie in a plane, then this volume will be zero. If the volume is non-zero, then the vectors do not lie in a plane.

What is the volume of the parallelepiped defined by \mathbf{u} , \mathbf{v} and \mathbf{w} ?

In light of your answer, is $\text{span}\{\mathbf{u}, \mathbf{v}, \mathbf{w}\}$ a line, a plane, or something else?

Use the GUI to confirm this.

(c)

We saw above that the vectors $\{(1, 2, 1), (2, -1, 0), (-4, 7, 2)\}$ are linearly dependent. What does this say about the determinant of the following matrix ?

$$B = \begin{bmatrix} 1 & 2 & -4 \\ 2 & -1 & 7 \\ 1 & 0 & 2 \end{bmatrix} \quad \det(B) =$$

(You should not need to do any calculation to answer this!)

(d)

Try to complete the following statement:

The vectors $\{\mathbf{v}_1, \dots, \mathbf{v}_n\} \in \mathbb{R}^n$ are linearly dependent if and only if $\det(A) =$
where A is the matrix

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

5 Computer Lab 5: Span, Linear Independence and Bases

5.1 Introduction

Today we continue to explore linear combinations, linear (in)dependence and span. As part of this, we will write a simple program that uses a for-loop and then save it as an *m-file*. In the final exercise we use matlab as a tool to investigate the row-space, column-space and solution-space of a matrix.

Before starting the exercises, start MATLAB and load this week's m-files:

- ▶ Make sure the *Current Folder* pane is showing in MATLAB.
(If it is not, use the 'Layout' menu on the MATLAB toolbar to add it.)
- ▶ Go to:
Start menu → Computer → Lab-Materials (L:) → MAST10007 Linear Algebra
- ▶ Drag the folder called *10007sheet5* directly into the MATLAB *Current Folder* pane
- ▶ Right-click on the *10007sheet5* folder in MATLAB's *Current Folder* pane, go to "Add to Path" and choose "Selected Folders and Subfolders".

5.2 Exercises

Exercise 1: Drawing vectors

We are going to use a program called **drawvec1** to continue exploring spans and linear combinations graphically. This program draws vectors on a graph. It requires 3 input variables:

- (1) a row or column matrix representing a vector in \mathbb{R}^2
- (2) a colour, and
- (3) a number to determine the size of the axes on which the vector is drawn.

Enter the following commands in MATLAB:

```
>x=[3 -4]
>abs(x)
>m=2+max(abs(x))
```

What do these last two commands do?

Notice that **max** returns the maximum value of all the entries in a matrix and **abs(x)** gives the absolute value of each entry in x .

Now enter

```
>drawvec1(x,'b',m)
```

You should see a new window called Figure 1 appear. If necessary you can bring this to the front by clicking on it. You should see a blue vector with its base at the origin and its end at $(3, -4)$. The second input variable for *drawvec1* was 'b' which determines the colour of the vector. The options are

'c', 'm', 'y', 'r', 'g', 'b', 'w' and 'k'

which correspond to cyan, magenta, yellow, red, green, blue, white and black. The third input, m , gives the limits of the axes, $-m \leq x \leq m$ and $-m \leq y \leq m$.

Try drawing the vector $(-4, 5)$ in green and the vector $(3, -7)$ in cyan.

We will now draw more than one vector on the same graph.

Enter

```
»x=[2 3];y=[-3 5];z=[4 -5]
```

Notice that we have entered three commands in one line. The ; is used to separate such commands. The output only gives the result of the final command.

Enter

```
»u=[x y z]
»m=2+max(abs(u))
»drawvec1(x,'r',m)
»hold on
»drawvec1(y,'b',m);hold on;drawvec1(z,'g',m)
»hold off
```

You should have a graph showing all three vectors.

Look at the output from $u = [x \ y \ z]$. This is the matrix with x in the first two columns, y in the 3rd and 4th columns and z in the final two columns. As you have seen in previous labs, this is a very handy way to produce a matrix from two smaller matrices. In contrast, the command $[x; y; z]$ produces a matrix with x in the top row, y in the second row and z in the third row.

Exercise 2: Exploration — The span of a vector

Now try entering the following commands which will generate 10 random multiples of the vector $(1, -1)$. Notice that some of the commands do not have » at the start. This is because when one enters **for** $i=1:10$, MATLAB expects a further sequence of commands ending in an **end** statement. The semi-colons at the end of the command lines are important. These suppress output from that command. Otherwise we would have ten lots of output in the command window.

```
»x=[1 -1];
»m=10;
»for i=1:10
    y=(rand-0.5)*20*x;
    drawvec1(y,'g',m+2);
    hold on
end
»hold off
```

Look at the output in the Figure 1 window.

So what did we do?

- After entering x and $m=10$ for the axes range, we introduced, in the command **for** $i=1:10$, a counter, i , which takes the values $1, 2, \dots, 10$. Initially i is set at 1. We then work through the commands:

```
y=(rand-0.5)*10*x;
drawvec1(y,'g',m+2);
hold on
```

When MATLAB first reaches the **end** command, i is set to the value 2 and the commands are then repeated. Each time MATLAB finishes the commands it increases the value of i until $i = 10$, when it moves on past **end** to **hold off**.

- The command **rand** generates random numbers between 0 and 1. We subtracted 0.5 from these to give numbers between -0.5 and 0.5 and then multiplied by 20, thus effectively generating random numbers

between -10 and 10 .

- We multiplied \mathbf{x} by the random number to produce \mathbf{y} and graphed the result.

So we have drawn 10 random vectors in the span of $\mathbf{x} = (1, -1)$.

The notation $\langle \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m \rangle$ denotes the *span* of the vectors $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m\}$. It is equal to the set of all vectors of the form

$$\alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2 + \dots + \alpha_m \mathbf{v}_m \quad \text{for } \alpha_1, \alpha_2, \dots, \alpha_m \in \mathbb{R}$$

Equivalently, the span is the set of all linear combinations of the vectors $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m\}$.

Looking at the figure you just produced, what do you think $\langle \mathbf{x} \rangle$ is?

Does \mathbf{x} span \mathbb{R}^2 ?

Exercise 3: Creating m-files

Let's save the commands you have just used as an m-file.

Enter

`>edit`

A new edit window will appear. Go to the *Command History* pane (if it is hidden, use the 'Layout' menu from the toolbar to show it) and highlight the commands from `x= ...` to `hold off` by clicking on the top line and shift-clicking on the bottom line. Copy these lines and paste them into the new window.

It should read

```
x=[1 -1];
m=10;
for i=1:10
    y=(rand-0.5)*20*x;
    drawvec1(y,'g',m+2);
    hold on
end
hold off
```

If it does not look like this, edit the commands appropriately.

Save your commands to an m-file by choosing 'Save' from the toolbar or pressing Ctrl+S. Save it under *My Documents* → *MATLAB* with filename `span.m`.

Now run your new program by typing `>span` in the command window.

(If you have any problem, check that the path is correctly set.)

In the edit window, try changing

```
x=[1 -1];
```

in your program to

```
x=(rand([1,2])-0.5)*2;
```

Save your program using Ctrl+S. Now run `span` again.

`rand([1,2])` generates a 1×2 matrix with entries taking values between 0 and 1, so \mathbf{x} is a randomly generated row vector with entries between -1 and 1 .

Run your program a few times.

Can a single vector ever span \mathbb{R}^2 ?

Exercise 4: Exploration — The Span of Two Vectors

Enter

```
>>help span2
```

to find out what the program `span2` does, and then

```
>>span2
```

to run it.

Look at the results in the figure window. What does it look like?

We randomly generated scalars between -10 and 10 to find our linear combinations of \mathbf{u} and \mathbf{v} .

Would we get a similar result for scalars between -100 and 100 ?

What kind of object do you think the span of \mathbf{u} and \mathbf{v} is?

Do \mathbf{u} and \mathbf{v} span \mathbb{R}^3 ?

Enter

```
>>edit span2
```

in the Command Window to see the code used in this program.

First you will notice a number of lines starting with `%`. These are not read as commands, but are comments which are printed in response to `help span2`.

The command `A=[0 0 0;u]` produces a matrix with the top row being the zero vector and the second row being the vector $\mathbf{u} = (1, -1, 2)$.

Enter

```
>>A(:,1)
```

in the Command Window. This prints the first column of A , which corresponds to the x coordinates of the two vectors $(0, 0, 0)$ and \mathbf{u} .

Try the commands `B(:,2)` and `A(:,3)`. What do you obtain?

$$A(:,1) = \begin{bmatrix} \\ \end{bmatrix} \quad B(:,2) = \begin{bmatrix} \\ \end{bmatrix} \quad A(:,3) = \begin{bmatrix} \\ \end{bmatrix}$$

The command `plot3(A(:,1),A(:,2),A(:,3),'k-')` draws a line from $(0, 0, 0)$ to $(1, -1, 2)$ giving a graphical representation of \mathbf{u} . There are 4 input variables for `plot3`. The first three give the x , y and z coordinates respectively of $(0, 0, 0)$ and $(1, -1, 2)$. The final input determines the colour, the type of line and the type of marker at the end of the line. In our case we specified the colour (black) and a solid line (-). The command for drawing a line in 2 dimensions is `plot`. If you want to find out more about `plot` and `plot3`, type in `help plot` and `help plot3` in the Command Window.

Change `span2` to randomly generate \mathbf{u} and \mathbf{v} . Run the program a few times to test it.

Save your new program using *Save As* from the top bar, calling it `myspan2.m`

Exercise 5: Theory — Bases

(a)

A basis for the space V is a set of vectors B that satisfy two important conditions. What are they?

(1)

(2)

An important use of a basis is that **every** vector in V can be written **uniquely** as a linear combination of the basis vectors. Note also that there are an infinite number of possible bases for any (non-trivial) subspace of \mathbb{R}^n . For particular applications, some bases might be easier to work with than others.

(b)

(i) Give a set of vectors in \mathbb{R}^3 which satisfies condition 1 but not condition 2.

(ii) Why do we need condition 2 in the definition of a basis?

(c)

(i) Now, try to find a set of vectors which satisfies condition 2 but not condition 1.

(ii) Why do we need condition 1 in the definition of a basis?

Hint: the phrases “too many vectors” and “not enough vectors” might be useful.

Exercise 6: Exploration — Column, Row and Solution Space

Let A be a $m \times n$ matrix.

- ▶ The *row space* is equal to the span of the rows of A . It is a subspace of \mathbb{R}^n .
- ▶ The *column space* is equal to the span of the columns of A . It is a subspace of \mathbb{R}^m .
- ▶ It is always the case that:

$$\dim(\text{column space}) = \dim(\text{row space}) = \text{rank}(A)$$

- ▶ The set of solutions of the equation $A\mathbf{x} = \mathbf{0}$ is called the *solution space*. It is a subspace of \mathbb{R}^n .
- ▶ The *nullity* of A is defined as the dimension of the solution space.
- ▶ The nullity and the rank are related by the equation:

$$\text{rank}(A) + \text{nullity}(A) = n$$

Using the things you have learned in lectures, and the MATLAB commands you have learned in the MATLAB classes, find a basis for the column space, row space and solution space of the following matrix. To save you typing-in the values you can run the program *tut2a*.

$$A = \begin{bmatrix} 25 & 30 & 24 & -17 & 61 & -12 \\ 41 & -10 & 64 & 76 & 44 & -1 \\ 57 & -50 & 104 & 170 & 29 & 13 \\ -48 & 120 & -120 & -280 & 49 & -36 \\ 162 & -220 & 328 & 619 & 7 & 60 \end{bmatrix}$$

Answers:

Row-reduced form of A (use the command `rats` to find rational approximations):

$$\left[\begin{array}{cccccc} & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \\ & & & & & \end{array} \right]$$

Basis for column space of A :

$\dim(\text{column space}) =$

Basis for row space of A :

$\dim(\text{row space}) =$

Basis for solution space of A :

nullity =

Note: You can check your answer for the last part using the MATLAB command `null(A, 'r')`, which gives a basis for the solution space of A .

Remark: Remember that a full description of the features of MATLAB can be found in the online help. To access this, click the **?** icon in the MATLAB toolbar and then click the 'MATLAB' link.

For example, you could use the search to look up the commands `rand`, `max`, `abs` etc.

6 Computer Lab 6: The Vector Space \mathbb{F}_2^n

6.1 Introduction

In this class you will learn about a new vector space, \mathbb{F}_2^n . As part of this lab you will need to do *modulo 2 arithmetic*. Next week, you will use this to develop a method of coding words for transmission or for storage in a database.

As usual, start up MATLAB and load this week's m-files. The files for this week are in the folder
 Computer → Lab-Materials (L:) → MAST10007 Linear Algebra → 10007sheet6 .
 Copy this folder into the MATLAB Current Folder pane and add it to the path.

6.2 The field \mathbb{F}_2

Let

$$\mathbb{F}_2 = \{0, 1\}.$$

This set containing just two elements, equipped with the operations of addition and multiplication given below, is called *the integers modulo 2*. Addition and multiplication are defined as follows:

Addition

$$\begin{aligned} 0 + 0 &= 0 \\ 1 + 0 &= 1 \\ 0 + 1 &= 1 \\ 1 + 1 &= 0 \end{aligned}$$

Multiplication

$$\begin{aligned} 0 \times 0 &= 0 \\ 1 \times 0 &= 0 \\ 0 \times 1 &= 0 \\ 1 \times 1 &= 1 \end{aligned}$$

The main difference between \mathbb{F}_2 and \mathbb{Z} is that $1 + 1 = 0$. We call the arithmetic in \mathbb{F}_2 , *modular* or *mod 2 arithmetic*. Try entering `mod(1+1,2)` in the MATLAB.

With the above definition of addition and multiplication, \mathbb{F}_2 is a *field*, that is, a system of numbers that shares many of the properties as \mathbb{Q} (the rational numbers), \mathbb{R} (the real numbers) and \mathbb{C} (the complex numbers).

Exercise 1: Exploring \mathbb{F}_2

1. Use *hand* calculations to find the following:
 - (a) $(1 + 0) \times 1 =$
 - (b) $1 + 0 \times 1 =$
 - (c) $1 + 1 + 1 =$
2. Use *hand* calculations to answer the following:
 - (a) If $a + b = b + a = 0$ then b is called the *additive inverse* of a . What are the additive inverses of the elements 0 and 1 in \mathbb{F}_2 ?
 - (b) If $a \times b = b \times a = 1$ then b is called the *multiplicative inverse* of a . What is the multiplicative inverse of the element 1 in \mathbb{F}_2 ?
 Does $0 \in \mathbb{F}_2$ have a multiplicative inverse?

Exercise 2: Row reducing and multiplying matrices using modulo 2 arithmetic

In the same way that we have matrices with real entries, we can have matrices with entries in \mathbb{F}_2 . The usual operations apply, except that addition and multiplication is done using mod 2 arithmetic.

In MATLAB enter the matrices

$$A = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix} \text{ and } \mathbf{x} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

- (a) Using *hand* calculations, find $A\mathbf{x}$ using mod 2 arithmetic.

$$A\mathbf{x} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} =$$

- (b) In the box below, use *hand* calculations to find the reduced row echelon form for the matrix A using mod 2 arithmetic. Remember to use $1 + 1 = 0$ to obtain zeros where needed.

$$A = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix} \sim$$

- (c) Use the commands `mod(A*x,2)` and `rrefmod2(A)` to check your answers.

$$\text{mod}(A*\mathbf{x},2)= \quad \quad \quad \text{rrefmod2}(A)=$$

Compare the results with the answers in (a) and (b).

The second command finds the row reduced form of the matrix A using modulo 2 arithmetic. It is a special function created for these classes. ***The usual command rref will not give the right answer.***

6.3 The vector space \mathbb{F}_2^n

We define addition and scalar multiplication for the set

$$\mathbb{F}_2^n = \{(a_1, a_2, \dots, a_n) \mid a_i \in \mathbb{F}_2, i = 1, \dots, n\}$$

as follows. Let $(a_1, a_2, \dots, a_n), (b_1, b_2, \dots, b_n) \in \mathbb{F}_2^n$ and $\lambda \in \mathbb{F}_2$. Then

$$(a_1, a_2, \dots, a_n) + (b_1, b_2, \dots, b_n) = (a_1 + b_1, a_2 + b_2, \dots, a_n + b_n)$$

and

$$\lambda(a_1, a_2, \dots, a_n) = (\lambda \times a_1, \lambda \times a_2, \dots, \lambda \times a_n)$$

where "+" and " \times " is mod 2 arithmetic as described in Section 6.2.

Theorem The set \mathbb{F}_2^n with addition and scalar multiplication as described above is a vector space using the integers modulo 2 as scalars.

6.4 Vector Space Basics

In this section you will revise some of the important definitions and use them in the context of \mathbb{F}_2^n . **It is important to realise that your lecture notes contain clear statements of the definitions required below. So do get out your notes and use them.**

Before answering the questions below run the program

`zed2input`

This contains the vectors used in this section and the matrix and vectors used in Section 6.5.

Remember to use the command `rrefmod2` throughout the following exercises.

Exercise 3 In this exercise you will be able to use the vectors \mathbf{u}_1 , \mathbf{u}_2 , \mathbf{u}_3 , \mathbf{u}_4 , \mathbf{u}_5 and \mathbf{u}_6 that you have just read into MATLAB. Let

$$\mathbf{u}_1 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \mathbf{u}_2 = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \mathbf{u}_3 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \mathbf{u}_4 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \mathbf{u}_5 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \mathbf{u}_6 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \in \mathbb{F}_2^5.$$

- (a) What is the rank of a matrix?

- (b) Use MATLAB to put the vectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_4$ in the columns of a matrix called B .

Find the rank of B .

- (c) Define a spanning set of a vector space V .

- (d) Use *hand* calculations to list all the vectors in $\text{span}\{\mathbf{u}_1, \mathbf{u}_2\}$. (Remember that the only scalars are 0 and 1.)

$$\mathbf{u}_1 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}, \mathbf{u}_2 = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

- (e) How can you test whether a set of vectors $\mathbf{v}_1, \dots, \mathbf{v}_k$ spans a given vector space \mathbb{F}_2^n ?

Before continuing you might find it useful to put the vectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_5, \mathbf{u}_6$ in the columns of a matrix called C and $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_4, \mathbf{u}_5, \mathbf{u}_6$ in the columns of a matrix called D .

- (c) Decide if either of the following sets of vectors spans \mathbb{F}_2^5 . Give a brief explanation. You may assume that $\dim(\mathbb{F}_2^5) = 5$. Do keep using `rrefmod2`.

(i) $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_5, \mathbf{u}_6\}$

(ii) $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_4, \mathbf{u}_5, \mathbf{u}_6\}$

- (d) Define linear dependence and linear independence of a set of vectors $\mathbf{v}_1, \dots, \mathbf{v}_k$.

- (d) How can you test for linear dependence and independence of the vectors $\mathbf{v}_1, \dots, \mathbf{v}_k \in \mathbb{F}_2^n$?

- (e) Are the following sets of vectors linearly dependent or linearly independent? Why?

(i) $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_5, \mathbf{u}_6\}$

(ii) $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_4, \mathbf{u}_5, \mathbf{u}_6\}$

(f) Write down the definition of basis and dimension of a vector space.

(g) Are either of the following sets of vectors a basis for \mathbb{F}_2^5 ? Give a brief explanation?

(i) $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{u}_5, \mathbf{u}_6\}$

(ii) $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_4, \mathbf{u}_5, \mathbf{u}_6\}$

(h) Why does $\dim(\mathbb{F}_2^5) = 5$?

6.5 Column and solution spaces

During the following exercises we assume that all entries in matrices and vectors are elements of \mathbb{F}_2 . You should use mod 2 arithmetic.

Exercise 4 This exercise will refer to the matrix M and the vectors \mathbf{v}_1 , \mathbf{v}_2 , \mathbf{v}_3 and \mathbf{v}_4 that were read in when you ran `zed2input`.

(a) Write down the definition of the solution space for a matrix M . (If necessary look at your lecture notes!)

(b) Let M be a $m \times n$ matrix and \mathbf{x} be a $n \times 1$ coordinate vector. How can you check whether or not \mathbf{x} is in the solution space of M ?

(c) Let

$$M = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

Using MATLAB decide whether or not the following are in the solution space of M .

(i) $\mathbf{v}_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$	(ii) $\mathbf{v}_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$	(iii) $\mathbf{v}_3 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$
--	---	--

(d) Look closely at your answer to (c)(iii). You should have found that $M\mathbf{v}_3 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$.

Notice that this is exactly the same as the fourth column of M . Also notice the following:

(i) $\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$. In fact, in \mathbb{F}_2^n , any vector when added to itself will give the zero vector.

(ii) Let $\mathbf{v}_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$. Then $M\mathbf{v}_4 = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$. Think about why this happens.

(iii) What vector could you multiply by $M = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{bmatrix}$ to get $\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$?

Can you see why the answer is $\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$?

(iv) What will you obtain if you find $M(\mathbf{v}_3 + \mathbf{v}_4)$?

(v) Is $\mathbf{v}_3 + \mathbf{v}_4$ in the solution space of M ?

- (vi) If we change the fourth entry of \mathbf{v}_3 from 1 to 0 we obtain $\mathbf{v}_3 + \mathbf{v}_4$, a vector in the solution space. So, by noticing that $M\mathbf{v}_3$ gives the *fourth column* in M , we know that by changing just the *fourth entry* in \mathbf{v}_3 we will get a vector in the solution space! In next week's lab we will utilize this fact to correct errors in data transmission.

Exercise 5: This exercise will refer to the matrix M and the vectors \mathbf{w}_1 and \mathbf{w}_2 that were read in when you ran `zed2input`.

- (a) Write down the definition of the column space for a matrix M . (If necessary look at your lecture notes!)

- (b) Let M be a $m \times n$ matrix and \mathbf{w} be a $m \times 1$ coordinate vector. How can you check whether or not \mathbf{w} is in the column space of M ?

- (c) Let M be as in Exercise 4.

Using MATLAB decide whether or not the following are in the column space of M .

(i) $\mathbf{w}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

(ii) $\mathbf{w}_2 = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}$

7 Computer Lab 7: Hamming Codes

7.1 Introduction

In this class you will learn how a Hamming Code is used to reduce errors occurring in data storage and message transmission.

As usual, start up MATLAB and load this week's m-files. The files for this week are in the folder
 Computer → Lab-Materials (L:) → MAST10007 Linear Algebra → 10007sheet7 .
 Copy this folder into the MATLAB Current Folder pane and add it to the path.

7.2 Coding

When information, such as messages or pictures, is stored or transmitted, there is always the possibility that the signal is corrupted along the way, due to imperfection in the wires, atmospheric conditions, or other causes. A single wrongly received letter could completely ruin a message, so we want to find a way to include **redundancy** in messages: extra information to reduce the effect of single errors. For instance, if I wanted to send the message "help" to someone on Mars, it is quite likely that one of the letters would get corrupted along the way, so they might receive "kelp".

In order to reduce the effect of an error, we could send the message this way:

hhhhheeeellllppppp

This time, if a single letter is corrupted, we would still be able to figure out what the message was originally:

e.g. hhkhheeeellllppppp.

We could take this idea further, repeating the letter even more, but eventually it becomes unfeasible as the message would require too much **bandwidth**, severely reducing the amount of information we can send at any time.

So, it becomes a question of balancing error-correction ability, with low bandwidth usage.

In this lab you will explore one way of coding data, a *Hamming code*, so that errors can not only be detected but also corrected efficiently. For a basic Hamming code the following occurs

1. Data is in the form of a string of 1s and 0s. These 1s and 0s are called *binary numbers*.
2. The strings are subdivided into strings of four 1's and 0's.
3. A 3×7 matrix, H , is chosen. The entries in H are elements of \mathbb{F}_2 , that is, they are also binary numbers.
4. Three 1's and 0's called the *check digits* are appended to each string in (2). These are chosen so that the resulting string is in the *solution space* of H .

The following exercise will walk you through the process of choosing check digits for a given H .

Exercise 1: A Basic Hamming Code

(a) Let

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

You may notice that the columns of this matrix are the binary forms of the numbers from 1 to 7 written bottom to top. For example, the binary number for 3, 011 is written from bottom to top in column 3 of H . This is convenient for identifying columns. The matrix H will be called the *check matrix* for our code.

Enter H in MATLAB.

We can find a basis for the solution space of H by solving

$$H \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \end{bmatrix} = \mathbf{0}$$

(b) What is the dimension of the solution space?

To find the solutions of the above equation, we let $\alpha_1 = s, \alpha_2 = t, \alpha_3 = u, \alpha_4 = v$ and find that

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \end{bmatrix} = \begin{bmatrix} s \\ t \\ u \\ v \\ t+u+v \\ s+u+v \\ s+t+v \end{bmatrix} = s\mathbf{v}_1 + t\mathbf{v}_2 + u\mathbf{v}_3 + v\mathbf{v}_4 = s \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} + t \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} + u \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} + v \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Note: While we usually choose the variables corresponding to the non-leading columns for our parameters, in this case we use the first four variables to ensure that the codeword includes the original word as the first four digits.

(c) Enter $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ and \mathbf{v}_4 in MATLAB as column vectors. Give them the names $\mathbf{v1}, \mathbf{v2}, \mathbf{v3}, \mathbf{v4}$. Note that it is easier to enter these vectors as the transpose of row matrices (e.g. $\mathbf{v1}=[1 \ 0 \ 0 \ 0 \ 0 \ 1 \ 1]'$).

Check that they the vectors are in the solution space of H , i.e. that $H\mathbf{v}_i = \mathbf{0}$. (Remember to use modulo 2 arithmetic, for example `mod(H*v1,2)`.)

(d) Are the vectors linearly independent?

(e) Are they a basis for the solution space of H ?

The linear combinations of \mathbf{v}_1 , \mathbf{v}_2 , \mathbf{v}_3 and \mathbf{v}_4 are the code words. Notice that

$$s\mathbf{v}_1 + t\mathbf{v}_2 + u\mathbf{v}_3 + v\mathbf{v}_4 = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 & \mathbf{v}_4 \end{bmatrix} \begin{bmatrix} s \\ t \\ u \\ v \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} s \\ t \\ u \\ v \end{bmatrix}$$

- (f) Enter the matrix $G = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3 & \mathbf{v}_4 \end{bmatrix}$ into MATLAB by using the command
`G=[v1 v2 v3 v4]`

G is called the *generating matrix* of the code. If the original data is $stuv$, the codeword is the string of bits obtained from

$$G \begin{bmatrix} s \\ t \\ u \\ v \end{bmatrix} \quad \text{using mod 2 arithmetic.}$$

For example if we have data 1000 and 1101 we obtain codewords as follows:

$$1000 : \quad G \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \text{The codeword is 1000011.}$$

$$1101 : \quad G \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad \text{The codeword is 1101001.}$$

Notice that the original data is the first four bits of the codeword and multiplication by G adds three additional bits. These extra three digits are called the *check bits*.

- (g) Use MATLAB to find the codewords for the following data. (Do not forget to use mod 2 arithmetic.) Record your answers.

1001:

0111:

7.3 Coding Letters and Messages

To code text, we need a way to convert letters, punctuation and common symbols into binary strings. Then we can code them using the methods of the previous exercise.

ASCII (American Standard Code for Information Interchange) is a common encoding system. There is a partial list converting ASCII codes to binary form at the end of this lab. For example, you will notice that R has the binary code 0101 0010.

Look at matrix G on your screen. It has only 4 columns. This means that it can only be used to create code words from 4 bits. But the binary code has 8 bits. Thus we need to divide the binary code into two sections, 0101 and 0010, and encrypt these as two codewords.

Exercise 2: Coding Messages

(a) Let

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

and enter this into MATLAB. Now find $G * R$ using mod 2 arithmetic.

You should obtain

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

So the codewords for R are 0101010 0010110. To reinterpret these as R we just take the first four digits of each codeword which is just the binary representation of R.

(b) Try encoding *Cat?* by

- (i) putting the ASCII code (see the end of the lab) in the 8 columns of a matrix called `cat`, then
- (ii) finding the codewords by multiplying `cat` by the generating matrix G

Word	ASCII Code	Coded Data
C		
a		
t		
?		

Check your encoding of *Cat?*, by multiplying H by your coded message. What should you get?

Exercise 3: Decoding messages

- (a) We will now try our hand at decoding a message. Retrieve the message to be decoded by running the program

`secrets`

You should see two matrices **M1** and **M2** containing the codewords in their columns. The first step is to detect any errors in the data.

Enter `cm1=mod(H*M1,2)` into MATLAB.

Does **cm1** contain all zeros? If it doesn't, what does this mean about the coded data you are trying to decode?

- (b) (i) The second column contains:

$$\begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}.$$

This means that the **second column** of **M1** has been corrupted. To correct the error

- Look at **H**. You will see that

$$\begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

is the **third** column of **H**. This means that the error has occurred in the **third bit** of the code word.

- Open **M1** in the workspace. Change the entry in the third row, second column from 0 to 1.
- Try `cm1=mod(H*M1,2)` again. This time the second column of **cm1** should be all zeros.

- (ii) Correct any other errors in **M1**.

- (iii) Use the corrected matrix to decode $M1$ and write the ASCII codes in the box below.
- (iv) Check, correct and decode $M2$ in the same way.
- (v) What does the whole message say?

ASCII Code First Half	ASCII Code Second Half	Translation

7.4 ASCII codes and their binary form

ASCII	Binary
space	0010 0000
!	0010 0001
\$	0010 0100
,	0010 1100
.	0010 1110
0	0011 0000
1	0011 0001
2	0011 0010
3	0011 0011
4	0011 0100
5	0011 0101
6	0011 0110
7	0011 0111
8	0011 1000
9	0011 1001
:	0011 1010
;	0011 1011
?	0011 1111

ASCII	Binary
A	0100 0001
B	0100 0010
C	0100 0011
D	0100 0100
E	0100 0101
F	0100 0110
G	0100 0111
H	0100 1000
I	0100 1001
J	0100 1010
K	0100 1011
L	0100 1100
M	0100 1101
N	0100 1110
O	0100 1111
P	0101 0000
Q	0101 0001
R	0101 0010
S	0101 0011
T	0101 0100
U	0101 0101
V	0101 0110
W	0101 0111
X	0101 1000
Y	0101 1001
Z	0101 1010

ASCII	Binary
a	0110 0001
b	0110 0010
c	0110 0011
d	0110 0100
e	0110 0101
f	0110 0110
g	0110 0111
h	0110 1000
i	0110 1001
j	0110 1010
k	0110 1011
l	0110 1100
m	0110 1101
n	0110 1110
o	0110 1111
p	0111 0000
q	0111 0001
r	0111 0010
s	0111 0011
t	0111 0100
u	0111 0101
v	0111 0110
w	0111 0111
x	0111 1000
y	0111 1001
z	0111 1010

8 Computer Lab 8: Inner Products

8.1 Introduction

In this class you will look at ways in which MATLAB can be utilized for applications of inner product spaces:

- Gram-Schmidt process for finding an orthonormal basis for the span of a set of vectors.
- Fourier series — analysis of music.

As usual, start up MATLAB and load this week's m-files. The files for this week are in the folder
 Computer → Lab-Materials (L:) → MAST10007 Linear Algebra → 10007sheet8 .
 Copy this folder into the MATLAB Current Folder pane and add it to the path.

8.2 The Gram-Schmidt Process

Recall from lectures that

The Gram-Schmidt procedure converts any basis $\{\mathbf{v}_1, \dots, \mathbf{v}_k\}$ for a vector space into an orthonormal basis. If there are three vectors in the basis the algorithm reads
 Step 1. Normalize \mathbf{v}_1 : $\mathbf{u}_1 = \mathbf{v}_1 / \|\mathbf{v}_1\|$.
 Step 2. Form $\mathbf{w}_2 = \mathbf{v}_2 - \langle \mathbf{v}_2, \mathbf{u}_1 \rangle \mathbf{u}_1$ and normalize \mathbf{w}_2 to get $\mathbf{u}_2 = \mathbf{w}_2 / \|\mathbf{w}_2\|$.
 Step 3. Form $\mathbf{w}_3 = \mathbf{v}_3 - \langle \mathbf{v}_3, \mathbf{u}_2 \rangle \mathbf{u}_2 - \langle \mathbf{v}_3, \mathbf{u}_1 \rangle \mathbf{u}_1$ and normalize \mathbf{w}_3 to get $\mathbf{u}_3 = \mathbf{w}_3 / \|\mathbf{w}_3\|$.
 The vectors $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\}$ are then an orthonormal basis.

In MATLAB, the *gschmidt* command does the above procedure very easily where the inner product is the Euclidean inner product. Let's have a go. First read in 3 vectors in $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 \in \mathbb{R}^4$:

```
>> tut5a
```

Put these as the columns of a matrix, A :

```
>> A=[v1 v2 v3 ]
```

Find the orthonormal basis of the set spanned by $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$, $S = \langle \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 \rangle$ by entering:

```
>> B=gschmidt(A)
```

Result: $B = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_3 \end{bmatrix}$

The orthonormal basis is the columns of this matrix B .

Enter

```
>> u1=B(:,1)
```

```
>> u2=B(:,2)
```

```
>> u3=B(:,3)
```

to obtain these as vectors.

Consider the vector $\mathbf{w} = (4, 0, 2, 1)$. We want to find the closest point in S , the span of $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$, to \mathbf{w} . This is $\text{Proj}_S(\mathbf{w}) = a_1\mathbf{u}_1 + a_2\mathbf{u}_2 + a_3\mathbf{u}_3$. To find a_i we use $a_i = \langle \mathbf{w}, \mathbf{u}_i \rangle = \mathbf{w} \cdot \mathbf{u}_i$.

Thus to find a_1, a_2, a_3 enter the following commands:

```
>> w=[4;0;2;1]
```

```
>> a1=dot(w,u1)
```

```
>> a2=dot(w,u2)
```

```
>> a3=dot(w,u3)
```

Answer: $a_1 =$ _____, $a_2 =$ _____, $a_3 =$ _____

Finally find $\text{Proj}_S(\mathbf{w})$, the closest point in S to \mathbf{w} by entering

```
>> proj= a1*u1+a2*u2+a3*u3
```

Answer: $\text{Proj}_S(\mathbf{w}) =$ $\begin{bmatrix} \\ \\ \\ \end{bmatrix}$
--

8.3 Fourier Series – Music Analysis

This exploratory exercise is to give you some idea how important the inner product space of functions is in real life. Here you will explore one of the most important tools used in science and technology – least squares estimation using Fourier series. Please complete as much as possible of the exercise in the Lab.

Given a sound, how do we break it down into its component sine and cosine waves? Recall from lectures that the set of vectors

$$\mathcal{B} = \left\{ \frac{1}{\sqrt{2\pi}} \right\} \cup \left\{ \frac{\sin x}{\sqrt{\pi}}, \frac{\sin 2x}{\sqrt{\pi}}, \frac{\sin 3x}{\sqrt{\pi}}, \dots \right\} \cup \left\{ \frac{\cos x}{\sqrt{\pi}}, \frac{\cos 2x}{\sqrt{\pi}}, \frac{\cos 3x}{\sqrt{\pi}}, \dots \right\}$$

forms an orthonormal set in the space of all continuous functions on the interval from $-\pi$ to π with the inner product defined by

$$\langle f, g \rangle = \int_{-\pi}^{\pi} fg \, dx.$$

Of course \mathcal{B} consists of an infinite number of vectors. In the following exercise we will use a finite subset of these vectors to approximate a sound.

We are going to consider one short cycle of sound. Typically this could be part of a piece of music or a sound produced by machinery or part of a sound received from outer space. Enter

```
>> tut5d
```

This uses the symbolic toolbox in MATLAB to define a function

$$f = (0.7(1 + (x - \pi)(x + \pi)(x - \pi/32)(x + \pi/32)x))/100$$

and plot it for you. You can see that f is a polynomial of degree 5. You may play this rather simple sound by entering

```
>> playnotef
```

To analyse this we find the orthogonal projection of f onto the space spanned by

$$\mathcal{B} = \left\{ \frac{1}{\sqrt{2\pi}} \right\} \cup \left\{ \frac{\sin x}{\sqrt{\pi}}, \frac{\sin 2x}{\sqrt{\pi}}, \frac{\sin 3x}{\sqrt{\pi}}, \dots, \frac{\sin 20x}{\sqrt{\pi}} \right\} \cup \left\{ \frac{\cos x}{\sqrt{\pi}}, \frac{\cos 2x}{\sqrt{\pi}}, \frac{\cos 3x}{\sqrt{\pi}}, \dots, \frac{\cos 20x}{\sqrt{\pi}} \right\}$$

Recall that to find the orthogonal projection of f onto a space spanned by a set of orthonormal vectors g_1, g_2, \dots, g_n we find, for $i = 1, 2, \dots, n$, $a_i = \langle f, g_i \rangle$ to obtain the vector

$$g = a_1 g_1 + a_2 g_2 + \dots + a_n g_n.$$

We call a_1, a_2, \dots, a_n the coordinates of g with respect to g_1, g_2, \dots, g_n . The function g is also called the *least squares approximation* of f in the space, W , spanned by the vectors in \mathcal{B} . In fact, $g \in W$ is the function in W which minimises $\int_0^{2\pi} (f(x) - g(x))^2 dx$.

We start by calculating the inner products. Enter

```
>> a0=eval(int(f*1/sqrt(2*pi),x,-pi,pi));
```

to find our first coordinate, a_0 . To find the sin coordinates enter (noting carefully the semi-colons)

```
>> A=zeros(1,20);
```

```
>> for i=1:20
```

```
A(i)=eval(int(f*sin(i*x)/sqrt(pi),x,-pi,pi));
```

```
end
```

Note that for a row matrix (i.e. dimension $1 \times n$), $A(i) = A(1, i)$.

The above commands create a matrix with the coordinates for $\sin x, \sin 2x, \dots, \sin 20x$ as its components. Open A in the workspace to see what the coordinates are. Now enter

```
>> tut5e
```

This does the same as the above code, but finding the coordinates for $\cos x, \dots, \cos 20x$ and putting them in a matrix B . Open B and you will see that all the coordinates are zero (f is an odd function). Thus we will take

$$g = a_0 \frac{1}{\sqrt{2\pi}} + A(1) \frac{\sin x}{\sqrt{\pi}} + A(2) \frac{\sin 2x}{\sqrt{\pi}} + \dots + A(20) \frac{\sin 20x}{\sqrt{\pi}}.$$

Enter

```
>> edit tut5f
```

to see the code for this and then enter

```
>> tut5f
```

to obtain $g = \text{Proj}_B f$ and to plot it and $f(x)$ on separate graphs. Open g in the Workspace to see the actual function obtained. Compare the two graphs and see if they look the same.

Try

```
>> playnotef
```

and

```
>> playnoteg
```

to compare the sounds. Does g sound the same as f ?

Further exercise 1: Look at the first 6 component parts of your sound,

$$A(1)\frac{\sin x}{\sqrt{\pi}}, A(2)\frac{\sin 2x}{\sqrt{\pi}}, \dots, A(6)\frac{\sin x}{\sqrt{\pi}}$$

in a graph by entering

```
>> tut5h
```

Look at your graph by bringing the graph window to the front. Then play the 6 component sounds by entering

```
>> playnoteh
```

Hit the space bar after each note to speed up the process. Only run playnoteh once or twice please.

Further exercise 2: Set $a0$, A and B to be zeros. (E.g. $A=\text{zeros}(1,20)$). Open $a0$, A and B in the Workspace and put your own numbers in them ensuring that the total of their absolute values is more than 0.5 and less than 1, i.e.

$$0.5 < |a0| + |A(1,1)| + \dots + |A(1,20)| + |B(1,1)| + \dots + |B(1,20)| < 1$$

Also make sure at least one of the $A(i,j) \neq 0$ or one of the $B(k,l) \neq 0$ as you need a sin or cos wave to make sound.

Run *tut5f* and *playnoteg* to see what you get.

9 Computer Lab 9: Eigenvalues and Eigenvectors

9.1 Introduction

This MATLAB class is designed to introduce the ideas of eigenvalues and eigenvectors and how to find them using MATLAB. We also consider problems with rounding errors and look at an important application of eigenvalues and eigenvectors.

As usual, start up MATLAB and load this week's m-files. The files for this week are in the folder
 Computer → Lab-Materials (L:) → MAST10007 Linear Algebra → 10007sheet9 .
 Copy this folder into the MATLAB Current Folder pane and add it to the path.

Note: Throughout this lab write your answers correct to 1 or 2 decimal places only.

9.2 Finding eigenvalues and eigenvectors with MATLAB

Let A be a square matrix. If $\mathbf{v} \neq \mathbf{0}$ satisfies $A\mathbf{v} = \lambda\mathbf{v}$ then \mathbf{v} is called an *eigenvector* of A and λ is the corresponding *eigenvalue*.

An $n \times n$ matrix A with n linearly independent eigenvectors $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n\}$ and corresponding eigenvalues $\lambda_1, \dots, \lambda_n$ can be *diagonalized* according to the formula

$$D = P^{-1}AP$$

where

$$P = [\mathbf{v}_1 \ \mathbf{v}_2 \ \cdots \ \mathbf{v}_n] \qquad D = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix}$$

Diagonalization is possible if and only if A has n linearly independent eigenvectors.

Below is the list of some of the matrices you looked at last week with the answers you should have obtained.

Please fill in the final column.

Matrix	How many linearly independent eigenvectors are there?	If there are any linearly independent eigenvectors, what are they?	Write down the corresponding eigenvalues.	Is the matrix diagonalizable with a real matrix P ?
$A1 = \frac{1}{4} \begin{bmatrix} 1 & 3 \\ 4 & 2 \end{bmatrix}$	2	$\begin{bmatrix} .7071 \\ -.7071 \end{bmatrix}, \begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix}$	-0.5, 1.25	
$A2 = \frac{1}{4} \begin{bmatrix} 2 & 4 \\ 2 & 4 \end{bmatrix}$	2	$\begin{bmatrix} -.8944 \\ .4472 \end{bmatrix}, \begin{bmatrix} .7071 \\ .7071 \end{bmatrix}$	0, 1.5	
$A3 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$	none	none	none	
$A4 = \frac{1}{4} \begin{bmatrix} 3 & 1 \\ -2 & 4 \end{bmatrix}$	none	none	none	
$A5 = \frac{1}{4} \begin{bmatrix} 6 & 4 \\ -1 & 2 \end{bmatrix}$	1	$\begin{bmatrix} .8944 \\ -.4472 \end{bmatrix}$	1	
$A6 = \frac{1}{4} \begin{bmatrix} 2 & 4 \\ -1 & -2 \end{bmatrix}$	1	$\begin{bmatrix} .8944 \\ -.4472 \end{bmatrix}$	0	

Enter the matrix

$$A = \begin{bmatrix} 4 & 2 \\ 4 & 3 \end{bmatrix}$$

into MATLAB. Now try the command

» `[P,D]=eig(A)`

You should obtain:

$$P = \begin{bmatrix} 0.6446 & -0.5101 \\ 0.7645 & 0.8601 \end{bmatrix} \text{ and } D = \begin{bmatrix} 6.3723 & 0 \\ 0 & 0.6277 \end{bmatrix}$$

The matrix P has two eigenvectors as its columns:

$$\mathbf{v}_1 = \begin{bmatrix} 0.6446 \\ 0.7645 \end{bmatrix} \text{ and } \mathbf{v}_2 = \begin{bmatrix} -0.5101 \\ 0.8601 \end{bmatrix}$$

The matrix D is a diagonal matrix containing the corresponding eigenvalues $\lambda_1 = 6.3723$ and $\lambda_2 = 0.6277$.

(Note that the order in which the eigenvalues appear in D is the same as the order in which the corresponding eigenvectors appear in P .)

Use the commands above to look at the six examples above again and fill in the table below. You can read-in the matrices by typing:

```
»lab10
```

You can use the up-arrow or copy/paste to save yourself retyping `[P,D]=eig(A)` each time.

Note that very small matrices will be preceded by $1.0e-a *$ where $a \in \mathbb{Z}$ meaning that the matrix should be multiplied by 10^{-a} .

(Write your answers correct to 2 decimal places.)

Matrix	Eigenvectors	Corresponding eigenvalues	Is the matrix diagonalizable?
$A1 = \frac{1}{4} \begin{bmatrix} 1 & 3 \\ 4 & 2 \end{bmatrix}$			
$A2 = \frac{1}{4} \begin{bmatrix} 2 & 4 \\ 2 & 4 \end{bmatrix}$			
$A3 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$			
$A4 = \frac{1}{4} \begin{bmatrix} 3 & 1 \\ -2 & 4 \end{bmatrix}$			
$A5 = \frac{1}{4} \begin{bmatrix} 6 & 4 \\ -1 & 2 \end{bmatrix}$			
$A6 = \frac{1}{4} \begin{bmatrix} 2 & 4 \\ -1 & -2 \end{bmatrix}$			

You will notice that not all the eigenvalues and eigenvectors are the same as those you identified by looking at \mathbf{x} and $A\mathbf{x}$ in \mathbb{R}^2 last week.

What is different about A_3 and A_4 and why could you not “see” the whole picture in \mathbb{R}^2 ?



While we do not deal with complex eigenvalues and eigenvectors in detail in this subject, they are very important in applications, for example, the solution of systems of differential equations describing simple harmonic and damped motion.

Look at the results for A_6 . While last week we identified just one eigenvalue of 0 and one corresponding eigenvector of $(.8944, -.4472)$, using the command `[P,D]=eig(A6)` we obtained two distinct and very small eigenvalues (of the order of 10^{-16}) and corresponding distinct eigenvectors differing by a very small amount. Why is this? The problem is that numerical errors arise because computations of eigenvalues and associated eigenvectors are sensitive to round-off error especially where eigenvalues are repeated. Users of MATLAB should be aware of such possible errors.

Exercise for later: Look at matrix A_6 by hand. Confirm that the characteristic equation is $x^2 = 0$ and find the number of eigenvectors to confirm whether or not A_6 is diagonalizable.

9.3 A Markov chain

A **stochastic process** is any sequence of events in which the outcome at any stage depends on chance. A **Markov process** is a stochastic process with the properties that the set of possible outcomes is finite, the probability of the next outcome depends only on the previous outcome and the probabilities are constant over time. The following is a typical Markov process.

Abominable snowmen are found in three regions of the Alps: Arcadia, Boronia and Catalia (say A,B and C for short). It is known that the abominable snowmen stay in one place for most of the time, but migrate to a new region at the start of January with the following probabilities:

		Location in year n		
		Arcadia	Boronia	Catalia
Location in Year $n + 1$	Arcadia	0.8	0.2	0.1
	Boronia	0.1	0.7	0.3
	Catalia	0.1	0.1	0.6

We can write these probabilities in a matrix

$$A = \begin{bmatrix} 0.8 & 0.2 & 0.1 \\ 0.1 & 0.7 & 0.3 \\ 0.1 & 0.1 & 0.6 \end{bmatrix}$$

and use it to calculate the number of abominable snowmen in each region. For example, assume that in year 0 there are 200 snowmen in Arcadia, 600 in Boronia and 200 in Catalia. Let a_n, b_n, c_n denote the number of abominable snowmen in Arcadia, Boronia and Catalia in year n respectively. Then in year 1 we would expect to have:

$$\begin{bmatrix} a_1 \\ b_1 \\ c_1 \end{bmatrix} = A \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix} = \begin{bmatrix} 0.8 & 0.2 & 0.1 \\ 0.1 & 0.7 & 0.3 \\ 0.1 & 0.1 & 0.6 \end{bmatrix} \begin{bmatrix} 200 \\ 600 \\ 200 \end{bmatrix} = \begin{bmatrix} 300 \\ 500 \\ 200 \end{bmatrix}$$

In year 2 we would expect to have:

$$\begin{bmatrix} a_2 \\ b_2 \\ c_2 \end{bmatrix} = A \begin{bmatrix} a_1 \\ b_1 \\ c_1 \end{bmatrix} = A^2 \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix} = \begin{bmatrix} 0.8 & 0.2 & 0.1 \\ 0.1 & 0.7 & 0.3 \\ 0.1 & 0.1 & 0.6 \end{bmatrix}^2 \begin{bmatrix} 200 \\ 600 \\ 200 \end{bmatrix} = \begin{bmatrix} 360 \\ 440 \\ 200 \end{bmatrix}$$

After n years we expect:

$$\begin{bmatrix} a_n \\ b_n \\ c_n \end{bmatrix} = A^n \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix}$$

Into MATLAB enter the matrix A and the initial numbers of snowmen $v = \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix}$.

Use MATLAB to find how many abominable snowmen there are in each location.

$$\begin{array}{ll} \text{After 5 years: } \begin{bmatrix} a_5 \\ b_5 \\ c_5 \end{bmatrix} = \begin{bmatrix} \\ \\ \end{bmatrix} & \text{After 10 years: } \begin{bmatrix} a_{10} \\ b_{10} \\ c_{10} \end{bmatrix} = \begin{bmatrix} \\ \\ \end{bmatrix} \\ \\ \text{After 100 years: } \begin{bmatrix} a_{100} \\ b_{100} \\ c_{100} \end{bmatrix} = \begin{bmatrix} \\ \\ \end{bmatrix} \end{array}$$

Now we ask the question: assuming that we have no loss or addition to our 1000 abominable snowmen through birth or death, will we ever have a stable number of snowmen in each region?

This amounts to asking the question: does

$$A^n \begin{bmatrix} a_0 \\ b_0 \\ c_0 \end{bmatrix} = \begin{bmatrix} 0.8000 & 0.2000 & 0.1000 \\ 0.1000 & 0.7000 & 0.3000 \\ 0.1000 & 0.1000 & 0.6000 \end{bmatrix}^n \begin{bmatrix} 200 \\ 600 \\ 200 \end{bmatrix}$$

stop changing for large enough n ?

Use the command `eig` (as above) to find the matrices P and D such that $D = P^{-1}AP$ or, equivalently $A = PDP^{-1}$.

Find $C = \lim_{n \rightarrow \infty} D^n$ as follows:

$$\begin{aligned} \lim_{n \rightarrow \infty} D^n &= \lim_{n \rightarrow \infty} \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}^n \\ &= \begin{bmatrix} \lim_{n \rightarrow \infty} \boxed{} & 0 & 0 \\ 0 & \lim_{n \rightarrow \infty} \boxed{} & 0 \\ 0 & 0 & \lim_{n \rightarrow \infty} \boxed{} \end{bmatrix} \\ &= \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix} = C \end{aligned}$$

Enter C into MATLAB and then

`»B=P*C*inv(P)`

to find $B = \lim_{n \rightarrow \infty} A^n$.

$$B = \begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}$$

What are the long term numbers of abominable snowmen in each location?

(Use the command $u=B*v$ and the result, u , will contain the required numbers.)

Arcadia:

Boronia:

Catalia:

Check that $Au = u$.

We finish today's lab with another way of looking at the long term locations of the abominable snowmen. We ask the question: "Does $A\mathbf{u} = \mathbf{u}$ have a solution?", or equivalently does A have an eigenvalue of 1? Look at D .

Do we have an eigenvalue of 1?

YES NO

If yes, what is the eigenvector from P associated with the eigenvalue 1?

$$\mathbf{u} = \begin{bmatrix} \\ \\ \end{bmatrix}$$

The eigenvector is a unit length vector. However because the number of abominable snowmen is constant at 1000, we need an eigenvector with entries adding to 1000. So the numbers of abominable snowmen are found from the eigenvector by multiplying it by $1000/(\text{sum of entries in eigenvector})$.

To find the long-term numbers of snowmen enter:

```
» w=P(:,1) » u=1000/sum(w)*w
```

The command `sum(w)` sums the entries in the unit length eigenvector w .

Write down your results.

$$\begin{bmatrix} \text{Number in } A \\ \text{Number in } B \\ \text{Number in } C \end{bmatrix} = \frac{1000}{(\text{sum of entries in } w)} \times w = \begin{bmatrix} \\ \\ \end{bmatrix}$$

Did you get the same long-term numbers as above?

YES NO

10 Computer Lab 10: Linear Transformations

10.1 Introduction

The main aim of this class is to help develop your understanding of linear transformations. We will be illustrating how objects are transformed in two dimensions. The final part of the tutorial looks at the way in which linear transformations can be used in computer graphics.

As usual, start up MATLAB and load this week's m-files. The files for this week are in the folder
 Computer → Lab-Materials (L:) → MAST10007 Linear Algebra → 10007sheet10 .
 Copy this folder into the MATLAB Current Folder pane and add it to the path.

10.2 Linear transformations of \mathbb{R}^2

Exercise 1: Drawing by hand

We start with a paper and pen exercise. The matrices in the table below represent linear transformations. Fill in the rest of the table explaining what you think the effect of each transformation is. Draw the image of the triangle under each transformation. (The grid lines are at distance 1 apart.) Remember that the image of a straight line under a linear transformation is either another straight line or a point.

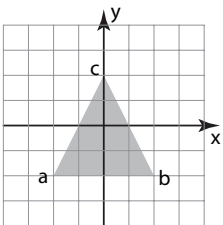
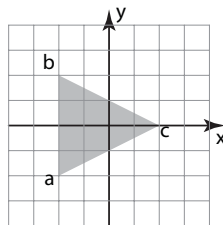
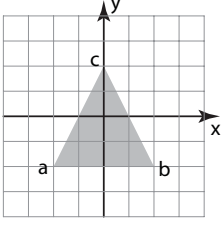
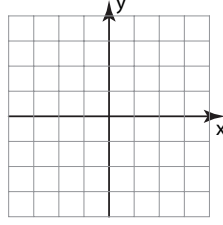
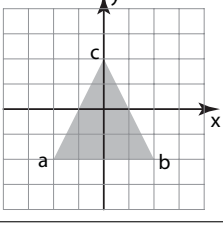
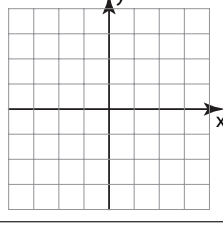
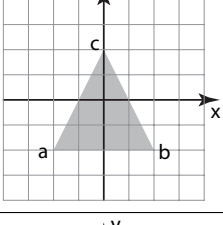
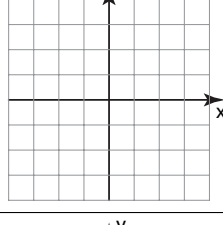
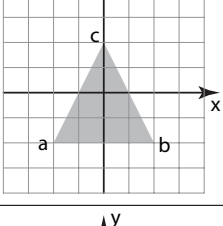
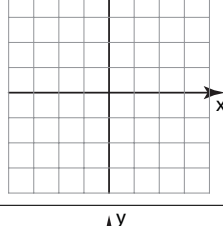
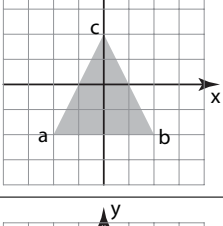
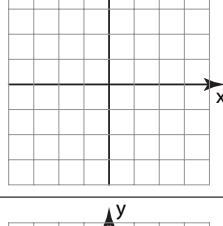
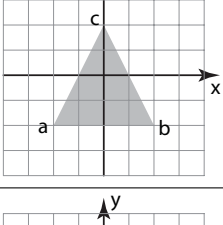
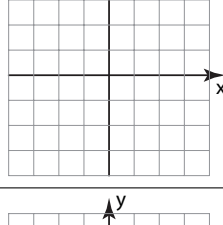
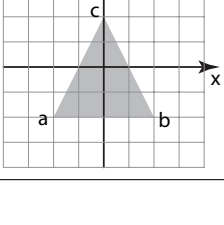
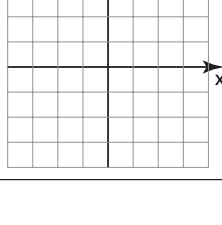
The first row has been completed as an example. The image of the point $a = (-2, -2)$ is the point $(-2, -2)$ since

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} -2 \\ -2 \end{bmatrix} = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$$

Similarly the point $b = (2, -2)$ is sent to the point $(-2, 2)$ and the image of the point $c = (0, 2)$ is the point $(2, 0)$.

Knowing the images of the points a, b, c , we can then fill in the image of the triangle without any further calculation.

Also, try to give a description in words for each transformation. Is it a shear, a rotation, a reflection or something else?

Matrix	Triangle	Image of triangle	Geometric description
$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$			reflection across the line $y = x$
$B = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$			
$C = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$			
$D = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$			
$E = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$			
$F = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$			
$G = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}$			
$H = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{2} \end{bmatrix}$			

Exercise 2: Plotting with MATLAB

Enter the command: `» x=house`

You will see a 2×12 matrix. Each column represents a point. Across the top row are the x -values and across the bottom row are the y -values. When we join these with line segments (i.e., join $(-6, -7)$ to $(-6, 2)$ with the first segment, then join $(-6, 2)$ to $(-7, 1)$ with the second segment, etc.) we obtain a picture of a house.

Try it by entering the following commands.

`»colordef black` ← sets the background colour of the figure
`»plot2dd(x)` ← plots the shape described by x

Now enter the matrices A to H from the table above into MATLAB.

Try plotting Ax to see what happens to the house under this transformation:

`»plot2dd(A*x)`

Now try applying the linear transformations B, C, D, E to x and plotting the result.

Compare the results with your descriptions of the linear transformations.

Were your answers about what the linear transformations do correct?

What happens if we combine two of these linear transformations?

Apply B to x , and then E to the result and then plot it as follows:

`»x1=B*x »x2=E*x1 »plot2dd(x2)`

Which of the following would give the same result, that is, applying the linear transformation B to x and then applying E ?

$BEx?$

☐ YES ☐ NO

$EBx?$

☐ YES ☐ NO

Check your answer by doing the following.

First create two new figures by entering: `»figure(2); figure(3)`

to obtain two new graphics windows, then enter the following commands:

`»figure(2) »plot2dd(B*E*x) »title('BEx') »figure(3) »plot2dd(E*B*x) »title('EBx')`

(The commands `figure(2)` and `figure(3)` set which window is to be used for the following plot.)

Compare the pictures of the house with the picture in Figure 1 to see if your answers were correct.

Exercise 3: Composition of three linear transformations

Close **all** graphics windows before continuing.

In this exercise we will find a matrix, S , that represents a shear in the y -direction of factor -2 , followed by a rotation of $-3\pi/4$, followed by a reflection in the line $x = 0$. (Write your answers correct to 2 decimal places.)

Matrix representing shear in y -direction
with factor -2 :

$$U = \begin{bmatrix} & \\ & \end{bmatrix}$$

Matrix representing rotation of $-3\pi/4$:

$$V = \begin{bmatrix} & \\ & \end{bmatrix} = \begin{bmatrix} & \\ & \end{bmatrix}$$

Matrix representing a reflection in the line $x = 0$:

$$W = \begin{bmatrix} & \\ & \end{bmatrix}$$

Required matrix for the composition of
the three linear transformations:

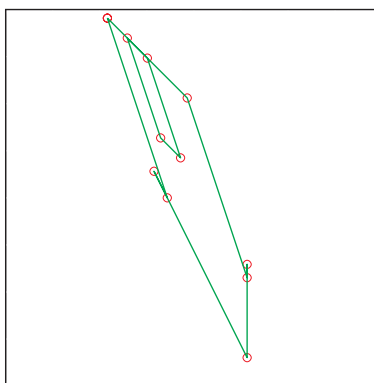
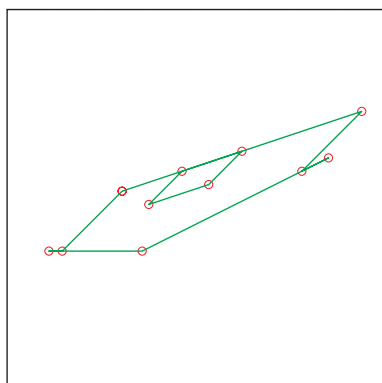
$$S = \quad \times \quad \times$$

From MATLAB this is:

$$S = \begin{bmatrix} & \\ & \end{bmatrix}$$

Now use MATLAB to apply the transformation S to the vector x and graph the result.

Which of the following is the image of the house under the transformation S ?



10.3 A simple animation

Computer generated graphics are a major feature in films and on television, to say nothing of computer games. These special effects are made up of a sequence of discrete pictures. How can we easily generate such a sequence of pictures?

Usually an object in a picture is made up of many dots (called pixels) each of which corresponds to a vector in \mathbb{R}^2 . One way of changing or moving the object is to apply linear transformations to it. We are going to try a very simple example. We will take **house** and, by applying a sequence of linear transformations, make it appear as if it is being sucked into a black hole.

The commands in the box below will be used to create the frames that make up the movie.

Arrange your windows on the screen so that you can enter commands while seeing the *Figure Window*. Then type in the commands listed below. Try to understand what each command is doing.

Commands	What they do
<code>clear</code>	Removes all previous material.
<code>x=house+10;</code>	Displaces house by (+10,+10) from the origin.
<code>plot(x)</code>	Does an initial plot of house in green.
<code>axis tight</code>	Says that the specified axis is not to be varied.
<code>set(gca,'nextplot','replacechildren')</code>	Tells MATLAB to replace each plot by the next.
<code>A=[cos(pi/6) -sin(pi/6);sin(pi/6) cos(pi/6)];</code>	A rotation of $\pi/6$.
<code>B=[.001^(1/24) 0;0 .001^(1/24)];</code>	A scaling.
<code>C=A*B;</code>	Composes the two linear transformations.
<code>axis([-10 20 -10 20])</code>	Sets the axis size.
<code>colordef black</code>	Sets the background colour of the figures.
<code>for i=1:24 x=C*x; plot(x) F(i)=getframe; end</code>	Applies the linear transformation, C , 24 times. Plots the transformed house. Stores the plot as a movie frame.

Now type in

`»movie(F)`

to run your movie.

11 Computer Lab 11: Test

11.1 Test

This week the computer laboratory session will be a test held in your regular weekly lab session. Details are available from the subject web page.

In the test you will be expected to use MATLAB as a tool in answering problems about the content of the lecture course.

Some questions will be multiple choice, and some will require you to, for example, write down a basis for a given vector space.

Please note the following:

- The test is 45 minutes (no reading time).
- You may not attend at a different class time unless you have a valid reason and have made arrangements with the tutor coordinator. Unless the reason is unexpected, such arrangements should be made at least two weeks before the test date.
- The test is not "open book". The only materials allowed are writing materials (i.e., pens and pencils).
- You must bring your student card!