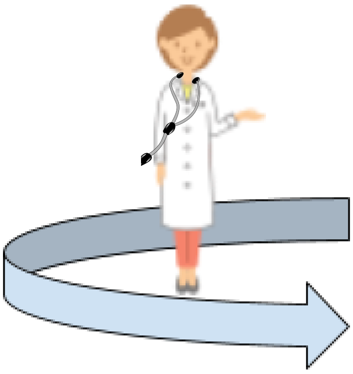
TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

ENGR 121 Laboratory Instructions 2018



Lab 4: Spin Doctor

WHAT IS A MATRIX and WHAT IS IT FOR?

4.1 Aim and Introduction

This lab exercise is designed to help you develop and/or improve some key skills:

1) Understanding rotation, translation, reflection, and the matrices that implement these operations.
2) Working with matrices
3) Developing familiarity and skill with trig identities.

Some of this may be completely new to you. You will be covering matrices in ENGR 121 this week, so this will give you a head start. Also, students taking CGRA 151 and/or ENGR 122 will be using matrices fairly extensively next term, so this should give you a preview.

We will use Python for much of this lab. You will likely see Python again! There is an appendix at the end of the lab that explains how to use it.

4.1.1 Review of Vectors

CORE 1 (10 marks) Take some notes about vectors - you can write by hand and take a photo.

<><><><><><><><><><><><><><><><><><><><><><><><>

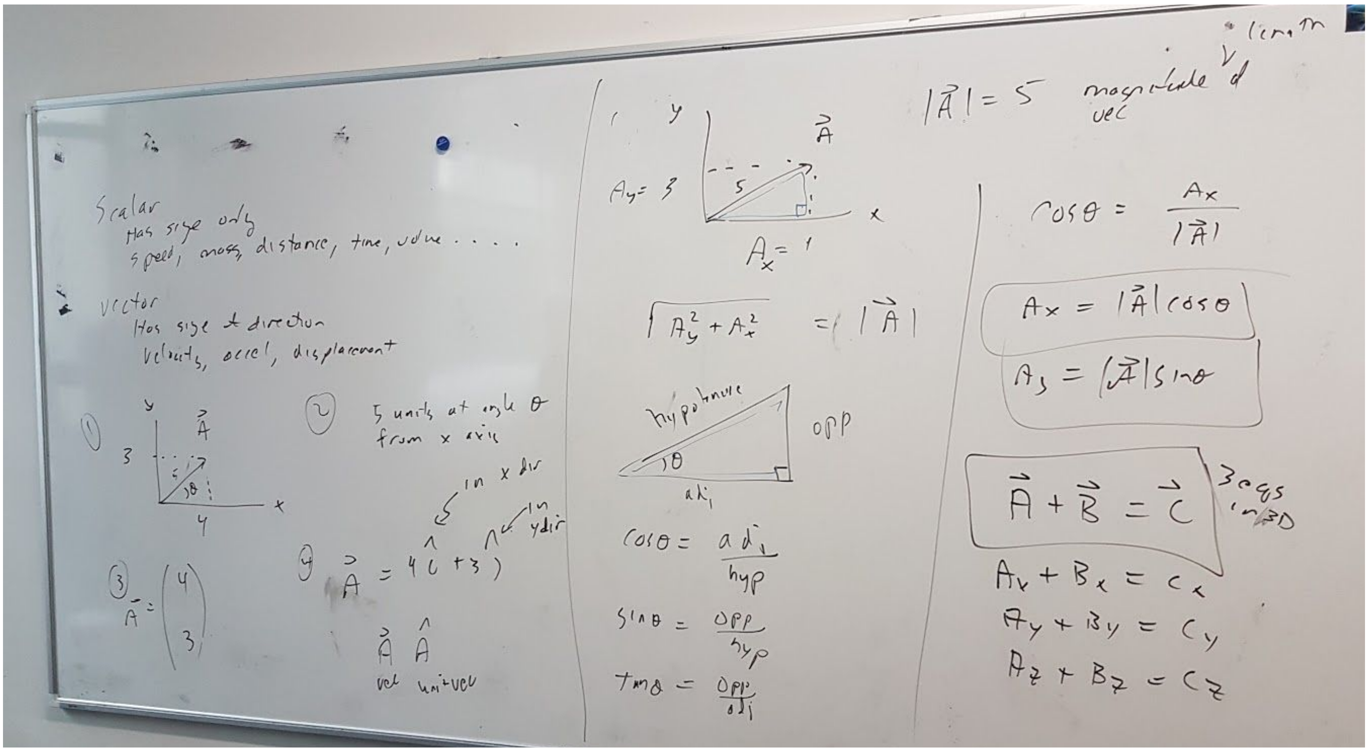
Vectors are comprised of 2 components, a size (magnitude) and a direction.

A = $(^4{}_3)$

Vectors components are written with the notation:

A_x and A_y (to represent the X and Y components of the vector)

A vectors magnitude is then written |A| which can be calculated through trig.


Scalar
Has size only
speed, mass, distance, time, volume . . . .
vector
Has size + direction
velocity, accel, displacement
1
y
$\vec{A}$
3
5
θ
x
4
2
5 units at angle θ from x axis
in x dir
in y dir
4 $\vec{A} = 4\hat{i} + 3\hat{j}$
3 $\vec{A} = \begin{pmatrix}4\\3\end{pmatrix}$
$\vec{A}$ $\hat{A}$
vec unit vec
y
$\vec{A}$
$A_y = 3$
5
x
$A_x = 4$
$\sqrt{A_y^2 + A_x^2} = |\vec{A}|$
hypotenuse
opp
θ
adj
$\cos\theta = \frac{adj}{hyp}$
$\sin\theta = \frac{opp}{hyp}$
$\tan\theta = \frac{opp}{adj}$
$|\vec{A}| = 5$ magnitude of vec
length
$\cos\theta = \frac{A_x}{|\vec{A}|}$
$A_x = |\vec{A}|\cos\theta$
$A_y = |\vec{A}|\sin\theta$
$\vec{A} + \vec{B} = \vec{C}$
3 eqs in 3D
$A_x + B_x = C_x$
$A_y + B_y = C_y$
$A_z + B_z = C_z$

<><><><><><><><><><><><><><><><><><><><><><><>

4.1.2 Matrices

CORE 2 (10 marks) What is a matrix? Take some notes, particularly about how to multiply a matrix times a vector.

<><><><><><><><><><><><><><><><><><><><><><><>

A matrix is a grid of numbers with a specific purpose in which operations can be done. To multiply matrices you multiply the rows by the columns each individually.

<><><><><><><><><><><><><><><><><><><><><><><>

ter4.2 Python

The appendix at the end of the lab script has some notes about matrices in Python. Include your Python work in your answers for the remainder of the lab. It is probably easiest to work in a text editor and paste into Python so you can reuse chunks of code.

You will need to import math and matrix functions:
Use from numpy import math, from numpy import matrix, from numpy.linalg import inv.

4.2.1 Vectors and Points

Throughout the rest of this lab, all of our vectors will stretch from zero to a point P (x,y). So we will use shorthand notation and speak of rotating a point. Picture rotating a vector from (0,0) to P.

4.3 The Rotation Matrix

This is your basic rotation matrix. It rotates a point or vector from (0,0) to a point counterclockwise in the XY plane by angle ϑ:

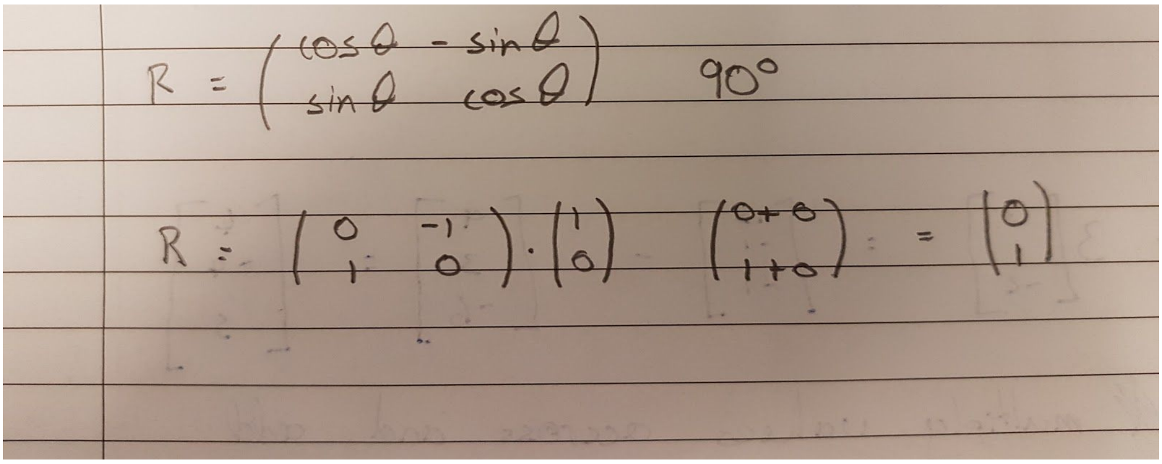
$$R=\begin{pmatrix}\cos\vartheta & -\sin\vartheta \\ \sin\vartheta & \cos\vartheta\end{pmatrix}$$

Let's take this matrix out for a spin. We will start with the vector from the origin to the point (1,0) and rotate it 90 degrees, and find out where it ends up.

4.3.1 First by hand

CORE 3 (10 marks):

<><><><><><><><><><>



$$R = \begin{pmatrix}\cos\theta & -\sin\theta \\ \sin\theta & \cos\theta\end{pmatrix} \quad 90°$$

$$R = \begin{pmatrix}0 & -1 \\ 1 & 0\end{pmatrix}\cdot\begin{pmatrix}1 \\ 0\end{pmatrix} \quad \begin{pmatrix}0+0 \\ 1+0\end{pmatrix} = \begin{pmatrix}0 \\ 1\end{pmatrix}$$

<><><><><><><><><><><><><><><><><><><><><>

<><><><><><><><><><><><><><><><><><><><><><><>

4.3.2 Using Python

Skip ahead to the appendix and work through the examples there.

4.3.3 Now by Python

Hints: Try this. Use a text editor. You can copy and paste this and then modify it for other parts of the lab. Remember to use ctrl-shift-v to paste in Python.

R = (math.cos(math.pi/2), -math.sin(math.pi/2), math.sin(math.pi/2), math.cos(math.pi/2))

R=matrix(R).reshape(2,2)

P=(1,0)

P=matrix(P).reshape(2,1)

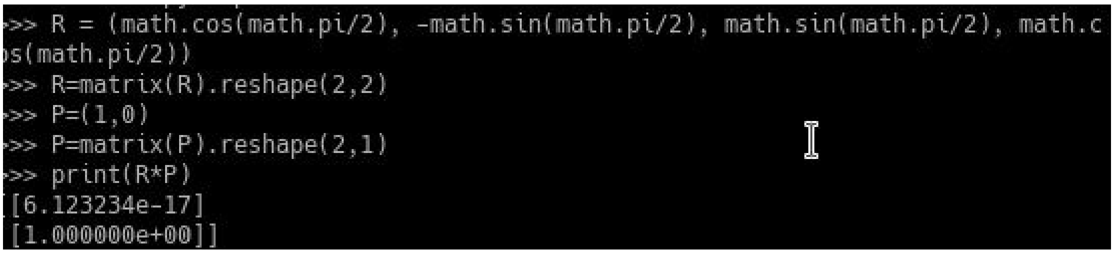
print (R*P)

Hint: Don't forget to use radians rather than degrees throughout the lab.

CORE 4 (10 marks):
Did this result in a point on the *y* axis (a vector along *y*) as expected? Include your python code.

<><><><><><><><><><><><><><><><><><><><><><><>

Yes it did what I expected



```
>>> R = (math.cos(math.pi/2), -math.sin(math.pi/2), math.sin(math.pi/2), math.c
os(math.pi/2))
>>> R=matrix(R).reshape(2,2)
>>> P=(1,0)
>>> P=matrix(P).reshape(2,1)
>>> print(R*P)
[[6.123234e-17]
 [1.000000e+00]]
```

<><><><><><><><><><><><><><><><><><><><><><><>

CORE 5 (5 marks)
What should happen if we rotate a point counterclockwise and then clockwise by the same angle?

<><><><><><><><><><><><><><><><><><><><><><><>

It will return to its initial position.

<><><><><><><><><><><><><><><><><><><><><><><>

CORE 6 (10 marks)
Try that! Rotate the point (1, 2) by 45 degrees counterclockwise and then clockwise. Include your Python code.

<><><><><><><><><><><><><><><><><><><><><><><>

```
>>> R = (math.cos(math.pi/4), -math.sin(math.pi/4), math.sin(math.pi/4),
math.cos(math.pi/4))
>>> R = matrix(R).reshape(2,2)
>>> P = (1,2)
>>> P = matrix(P).reshape(2,1)
>>> PR = R*P
>>> print PR
[[-0.70710678]
 [ 2.12132034]]
>>> R2 = (math.cos(-(math.pi/4)), -math.sin(-(math.pi/4)),
math.sin(-(math.pi/4)), math.cos(-(math.pi/4)))
>>> R2 = matrix(R2).reshape(2,2)
>>> NP = R2*PR
>>> matrix(NP).reshape(2,1)
matrix([[1.],
        [2.]])
```

<><><><><><><><><><><><><><><><><><><><><><><>

COMPLETION 1 (10 marks)
Rotate your point (1,2) by 300 degrees and then rotate the result by a further 100 degrees. Then rotate your point (1,2) by an equivalent angle that is between 0 and 360 degrees. Do you get the same results?

<><><><><><><><><><><><><><><><><><><><><><><>

Rotating by 300, then by 100:

```
>>> R = (math.cos(10*(math.pi/6)), -math.sin(10*(math.pi/6)),
math.sin(10*(math.pi/6)), math.cos(10*(math.pi/6)))
>>> R = matrix(R).reshape(2,2)
>>> P = (1,2)
>>> P = matrix(P).reshape(2,1)
>>> R2 = (math.cos(10*(math.pi/18)), -math.sin(10*(math.pi/18)),
math.sin(10*(math.pi/18)), math.cos(10*(math.pi/18)))
>>> R2 = matrix(R2).reshape(2,2)
>>> P1 = R*P
>>> P2 = R2*P1
>>> print P2
[[-0.51953078]
 [ 2.1748765 ]]
```

Rotating by 40:

```
>>> R = (math.cos(4*(math.pi/18)), -math.sin(4*(math.pi/18)),
math.sin(4*(math.pi/18)), math.cos(4*(math.pi/18)))
>>> R = matrix(R).reshape(2,2)
>>> print R*P
[[-0.51953078]
 [ 2.1748765 ]]
```

They output the same answer.

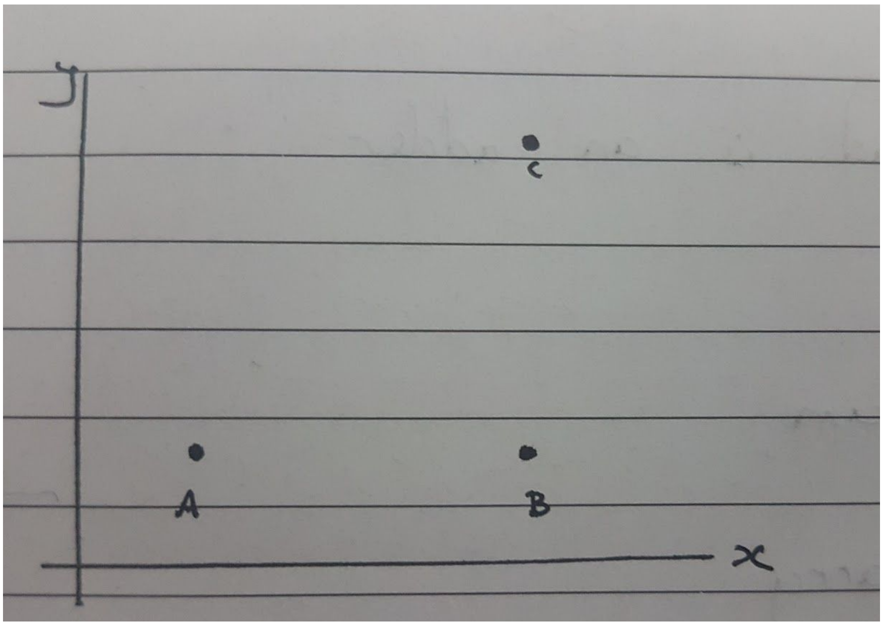
<><><><><><><><><><><><><><><><><><><><><><><>

COMPLETION 2 (10 marks)

Now let’s work on a triangle. We will define our triangle by the three points (1,1), (3,1), and (3,4). Make a quick sketch of the triangle (you can make a sketch on paper and insert a photo). Rotate your triangle by 75 degrees using Python and sketch the result. Hint: rotate each point. Did we successfully rotate the triangle? Hint: label the points in the original triangle A, B, C and the points in the rotated triangle A’, B’, C’. This rotation may not have been what you had in mind! What point did we rotate the triangle around?

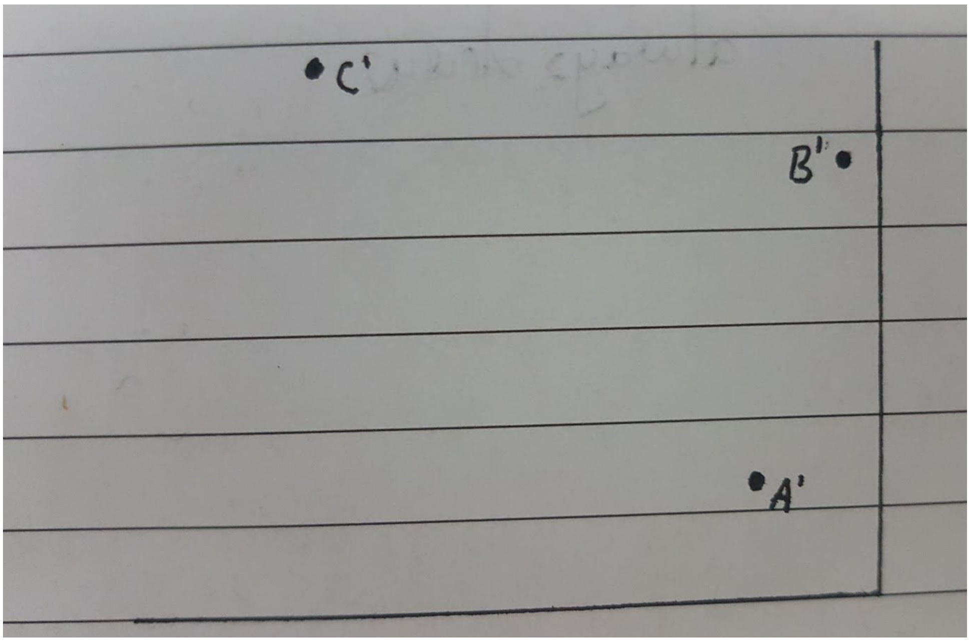
<><><><><><><><><><><><><><><><><><><><><><><>

```
R = (math.cos(75*(math.pi/180)), -math.sin(75*(math.pi/180)),
math.sin(75*(math.pi/180)), math.cos(75*(math.pi/180)))
R = matrix(R).reshape(2,2)

A = (1,1)
A = matrix(A).reshape(2,1)

B = (3,1)
B = matrix(B).reshape(2,1)

C = (3,4)
C = matrix(C).reshape(2,1)

A1 = R*A
B1 = R*B
C1 = R*C

print A1
print B1
print C1

print A1
[[-0.70710678]
 [ 1.22474487]]
print B1
[[-0.18946869]
 [ 3.15659652]]
print C1
[[-3.08724617]
 [ 3.93305366]]
```



Original triangle

A = (1,1)
B = (3,1)
C = (3,4)



Rotated triangle

```
print A1
[[-0.70710678]
 [ 1.22474487]]
print B1
[[-0.18946869]
 [ 3.15659652]]
print C1
[[-3.08724617]
 [ 3.93305366]]
```

We have rotated each point of the triangle around the point (0,0)

<><><><><><><><><><><><><><><><><><><><><><><>

SUPERCHALLENGE (0 marks - do this if you want to get ahead of the class a bit after you finish the rest of the lab)
Now we will work with our rotation matrix algebraically. You can work on a sheet of paper and upload a photo of your work. Use the definition of the rotation matrix, matrix multiplication, and trig identities show that a rotation by an angle ϑ in the counterclockwise direction followed by a rotation of the same magnitude in the clockwise direction results in no rotation at all.
<><><><><><><><><><><><><><><><><><><><><><><>

<><><><><><><><><><><><><><><><><><><><><><><>

SUPERCHALLENGE (0 marks - do this if you want to get ahead of the class a bit after you finish the result of the lab.)
Find the inverse of $R(\vartheta)$ and compare the result to $R(-\vartheta)$. How is this related to our results in Completion 1?
<><><><><><><><><><><><><><><><><><><><><><><>

<><><><><><><><><><><><><><><><><><><><><><><>

4.4 The Reflection Matrix

To reflect a point across a line through the origin and (x,y) we use

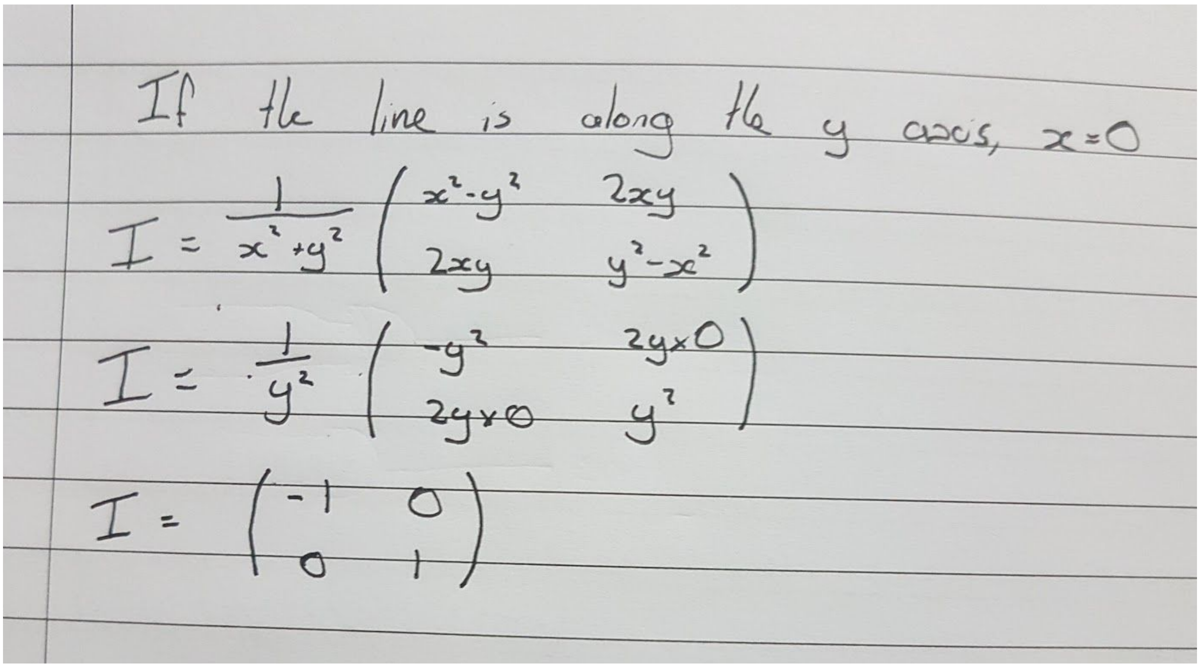
$$I=\frac{1}{(x^2+y^2)}\begin{pmatrix} x^2-y^2 & 2xy \\ 2xy & y^2-x^2 \end{pmatrix}$$

COMPLETION 3 (5 marks)

For a line along the y axis, show this reduces to

$$I=\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$$

<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>



If the line is along the y axis, $x=0$

$$I = \frac{1}{x^2+y^2}\begin{pmatrix} x^2-y^2 & 2xy \\ 2xy & y^2-x^2 \end{pmatrix}$$

$$I = \frac{1}{y^2}\begin{pmatrix} -y^2 & 2y\times 0 \\ 2y\times 0 & y^2 \end{pmatrix}$$

$$I = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$$

<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>

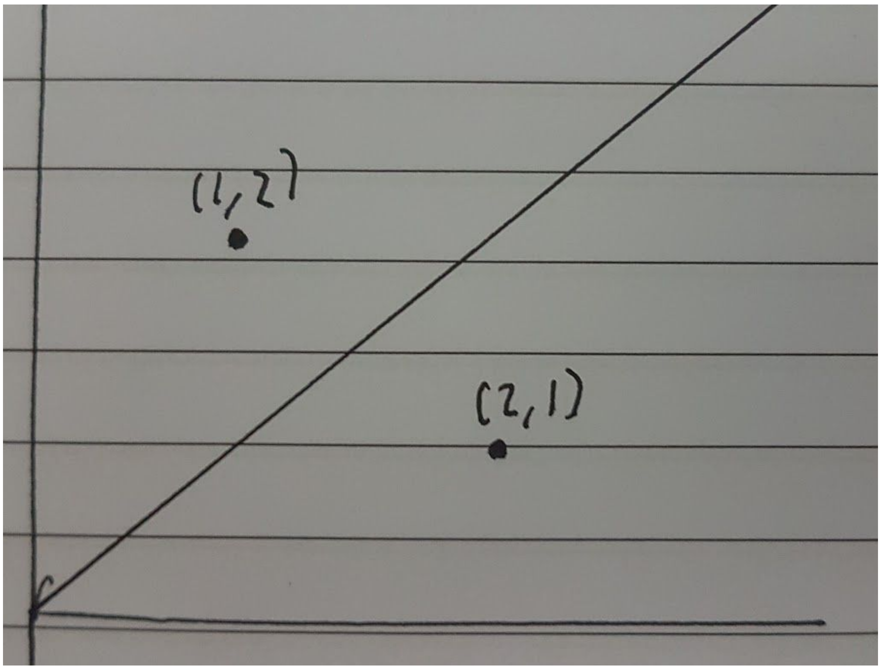
CHALLENGE 1 (10 marks)

Reflect your point (1,2) across a line at 45 degrees to the *x* axis. Make a quick sketch of the point before and after reflection. Is the operation behaving as expected? You can do this by hand and/or use Python.

<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>

If reflected by 45 degrees, x = y, therefor the matrix becomes

```
[0 1]
[1 0]

P = (1,2)
P = matrix(P).reshape(2,1)

R = (0,1,1,0)
R = matrix(R).reshape(2,2)

P = R*P
print P

[[2]
 [1]]
```



<><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><><>

4.5 Combining Operations

The stretch matrix below will stretch an object along the x axis by factor s.

$$S=\begin{pmatrix} s & 0 \\ 0 & 1 \end{pmatrix}$$

CHALLENGE 2 (10 marks)
Stretch your triangle (the original) by a factor of 2 along the y axis. You can do this by hand and/or use Python.

<><><><><><><><><><><><><><><><><><><><><><><>

```
A = (1,1)
A = matrix(A).reshape(2,1)

B = (3,1)
B = matrix(B).reshape(2,1)

C = (3,4)
C = matrix(C).reshape(2,1)

S = (2,0,0,1)
S = matrix(S).reshape(2,2)

print S*A
[2]
[1]

print S*B
[6]
[1]

print S*C
[6]
[4]
```

<><><><><><><><><><><><><><><><><><><><><><><>

APPENDIX

Using Python to Work With Matrices

To launch Python just get a terminal and type python.

The first step is to import the maths libraries you will need:
Hint: you actually have to type the word “from”.

```
from numpy import matrix
from numpy.linalg import inv
```

To set up a matrix with nine elements and then display it, for example, type

```
A = (1, 2, 2, 4, 3, 2, 3, 2, 1)

A = matrix(A).reshape(3, 3)

print A
```

You should see

$$\begin{pmatrix} 1 & 2 & 2 \\ 4 & 3 & 2 \\ 3 & 2 & 1 \end{pmatrix}$$

To create a column vector use, for example,

```
b = (1, 2, 5)

b=matrix(b).reshape(3, 1)
```

To find the inverse of a matrix use

```
InvA = inv(A)
```

this creates a matrix named InvA which is the inverse of A.

To multiply two matrices just use * as usual. For example, type

```
print InvA*A
```

you should see the identity matrix (with some slight rounding errors – keep that in mind!)