Ver1.1: 8 Oct 2020

LABORATORY MANUAL

CX1104: Linear Algebra for Computing
Hardware Lab 1 (Location: N4-01a-03)

SESSION 2020/2021
SEMESTER 1
COMPUTER ENGINEERING COURSE

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING
NANYANG TECHNOLOGICAL UNIVERSITY

LAB – Ch 6

LINEAR ALGEBRA FOR COMPUTING

1. OBJECTIVE

The objective of this laboratory is to learn linear algebra using the python programming language as a tool. For this laboratory session, you will:

a) Implement your own functions to calculate norm and dot product.
b) Measure the similarity utilizing norm and dot product.
c) Implement your own functions for orthogonal decomposition.
d) Find the closest point in a column space to a given point.

Specifically, we will work with python 3.7 with Jupyter Notebook under the Anaconda environment [1]. You can choose to use the school's laboratory environment or your own laptop's environment during the laboratory. You can read about python and how to use it for development in [2-3]. You can also read about why python vs Matlab here [4-5,7] and Matlab's response [6].

Snippets of python code which can help in this laboratory is provided in the Jupyter Demo.

2. Expectations of students for the Laboratory sessions.

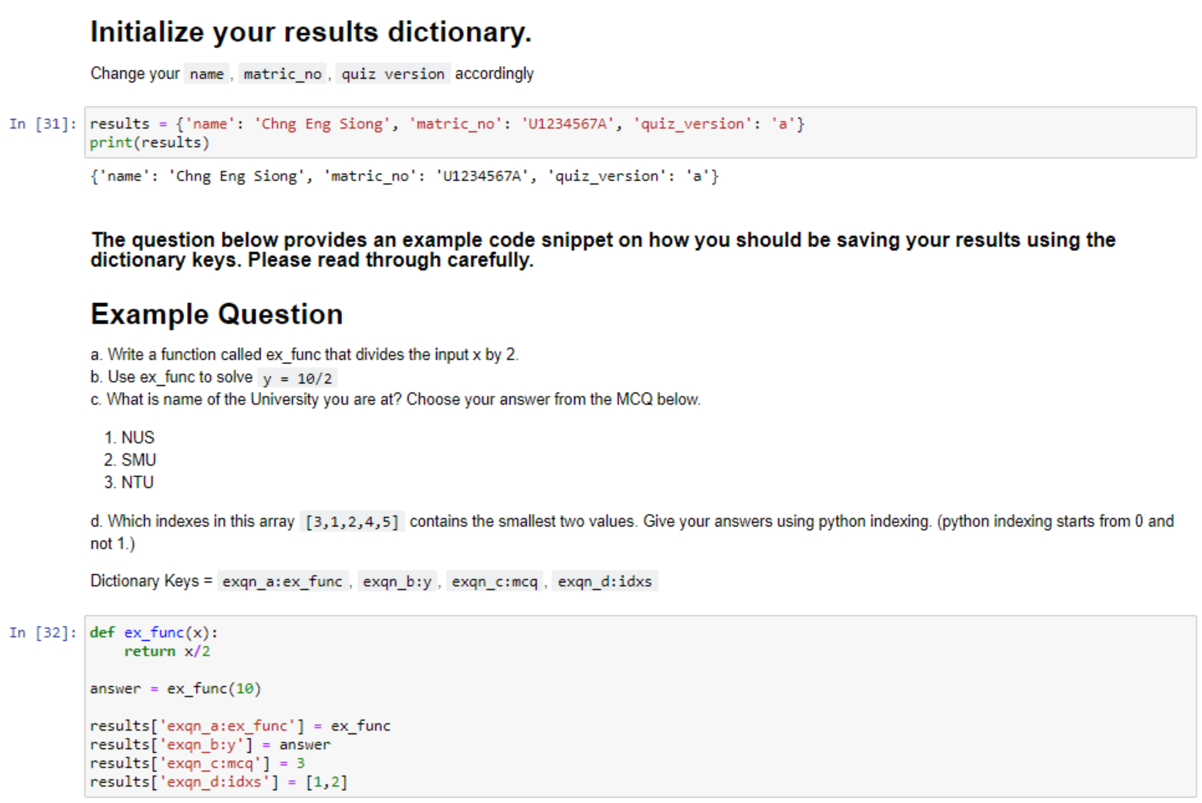
The following rules and regulations apply for all the laboratories pertaining to CX1104.

You are expected to have completed all the exercises of the laboratory before attending the laboratory. The main purpose of the laboratory is to conduct a laboratory quiz.

If you have questions on how to develop the code, check with the TA (Teaching Assistant) before the laboratory class. If you have not complete the laboratory exercises before coming to lab, you are unlikely to finish the quiz questions.

Before you come to the Laboratory Session:

1. Ensure that your python environment and jupyter notebook are working as expected..
2. Have pandas and dill already installed in your python environment. You can do this by running "pip install dill pandas". Dill is an extension library of pickle which helps us serialize our variables. Pandas is a popular tool for loading files. We will require these libraries to facilitate the lab. Ensure that your pandas version is >=1.1.1 and dill version >= 0.3.2
3. Familiarize yourself with python basics, especially dictionaries, and functions.
4. Please see the below image for an example of how to store your results in a dictionary.



Initialize your results dictionary.

Change your `name`, `matric_no`, `quiz version` accordingly

In [31]:
```
results = {'name': 'Chng Eng Siong', 'matric_no': 'U1234567A', 'quiz_version': 'a'}
print(results)
```

```
{'name': 'Chng Eng Siong', 'matric_no': 'U1234567A', 'quiz_version': 'a'}
```

The question below provides an example code snippet on how you should be saving your results using the dictionary keys. Please read through carefully.

Example Question

a. Write a function called ex_func that divides the input x by 2.
b. Use ex_func to solve `y = 10/2`
c. What is name of the University you are at? Choose your answer from the MCQ below.

1. NUS
2. SMU
3. NTU

d. Which indexes in this array `[3,1,2,4,5]` contains the smallest two values. Give your answers using python indexing. (python indexing starts from 0 and not 1.)

Dictionary Keys = `exqn_a:ex_func`, `exqn_b:y`, `exqn_c:mcq`, `exqn_d:idxs`

In [32]:
```
def ex_func(x):
    return x/2

answer = ex_func(10)

results['exqn_a:ex_func'] = ex_func
results['exqn_b:y'] = answer
results['exqn_c:mcq'] = 3
results['exqn_d:idxs'] = [1,2]
```

During the Laboratory Quiz:

1. At the beginning of the laboratory lesson, you will be provided with **a jupyter notebook** and a **.db file**. Place the .db file in the same directory as the jupyer notebook.
2. It is an open book laboratory quiz, i.e., you can use your notes, and the internet to help answer the questions. You are however **not allowed** to receive help from any other person. The questions will be related to the given tasks of the lab as well as the lectures/tutorials you have attended.
3. Once you have completed the lab quiz, you are required to **submit both the dill file and the notebook**. The code and instructions for submission will be provided.
4. If you are absent for the laboratory quiz with valid medical reason, the quiz mark for the absent quiz will be the average of your other quiz marks. Without a valid reason, you will receive 0 marks. Please submit valid MC and exemptions to be away by emailing them to the TA for the laboratory and cc the laboratory technician in charge.

3. Instructions

Develop your own python functions to perform the tasks listed in section 4. In other words, you are not allowed to use the functions of others, public libraries, and etc. You are allow to use the following python libraries for generic functions, such as generating sine, cosine, exponential values, plotting figures for visualization, etc:

```
import numpy as np
import numpy.linalg as la
```

To help students, see the provided codes in Jupyter Demo. Warning: do not write your code in this manner – we expect you to be able to create reusable functions.
In the laboratory, use the preinstalled Jupyter Notebook to write and run your code. You may need to create a virtual environment specifically. See the README file for the configuration.

4. Tasks

Q1. Dot Product, Norm, Cosine Similarity.

Table 1. Data base for Q1.

Name \ Attribute	Weight (Kg)	Height (cm)	Age (year)	Temperature (C°)
Bob	62.2	171.1	17	36.8
Alice	52.2	162.6	19	36.5
Charlie	72.3	178.2	22	36.7
Chunk	80.8	185.2	24	37.9
Unknown Person	72.5	178.3	22	37.8

Suppose that a database has personal information captured from four persons and 1 unknown person. You are required to develop a simple system to identify the unknown person utilizing the norm and dot product.

Follow the steps below to achieve this goal.

a. **Initiatise the 5×4 data matrix raw_A1 from Table 1 using $numpy.array([[]], dtype = float)$.** (Explanation: We have five examples (persons), each of which has four features (attributes). The first step is to input all these data as a matrix.) Remove the last column of raw_A1 (as it is not a physical attribute) to form **A1.**

b. How will you measure the similarity between two persons based on the norm and dot product? Describe your idea abstractly.

c. Implement your own routines for norm,dot product and cosine similarity **Name your routines as $my\_norm(x)$, $my\_dot(x, y)$, my_cosSimilarity(x,y).**

d. **Report the results of your own implementations (my_norm, my_dot and my_cosSimilarity) and corresponding numpy routines on Bob and Alice.** [Hint: your results should match numpy routines numpy.dot and numpy.linalg.norm.]

e. Obtain a new data matrix $A1\_norm$ by normalizing each row of $A1$ to unit vector using your own routine of norm. **Report the matrix *A1_norm* you obtained. (Explanation**: this ensures that each example is presented by a unit vector)

f. Repeat Q1d using normalized Matrix *A1_norm*.

g. Using A1_norm, compute the cosine similarity of person row 5 (unknown) against row1-4. **Report the similarity vector you obtained.**

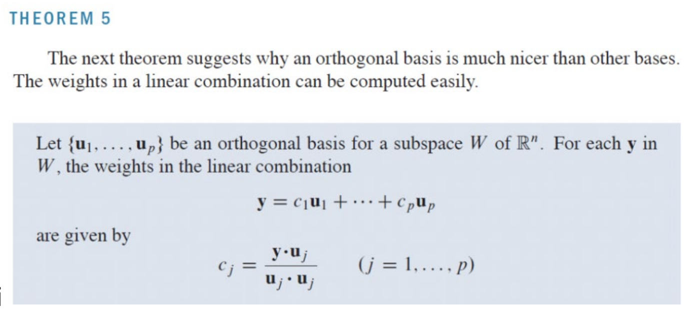
h. Which person on the table is most similar to the unknown person? **Write your answer according to your similarity measure vector b.**

Q2. Projection, Orthogonality.

Let $A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix}$, $B = \begin{bmatrix} 3 & -1 & -1/2 \\ 1 & 2 & -2 \\ 1 & 1 & 7/2 \\ 0 & 1 & 0 \end{bmatrix}$

a. Which matrix's (A or B) columns form an orthogonal set? Why?

b. Express $y_1 = [2.5, 11, -7.5, 2]^T$ as a linear combination of columns of matrix B (i.e find the vector x such that $y_1 = Bx$) by Theorem 5 (Lay, pg 399). Show that your solution Bx is equals to y_1.



THEOREM 5

The next theorem suggests why an orthogonal basis is much nicer than other bases. The weights in a linear combination can be computed easily.

Let $\{\mathbf{u}_1, \dots, \mathbf{u}_p\}$ be an orthogonal basis for a subspace W of $\mathbb{R}^n$. For each $\mathbf{y}$ in W, the weights in the linear combination

$$\mathbf{y} = c_1\mathbf{u}_1 + \cdots + c_p\mathbf{u}_p$$

are given by

$$c_j = \frac{\mathbf{y}\cdot\mathbf{u}_j}{\mathbf{u}_j\cdot\mathbf{u}_j} \qquad (j = 1, \dots, p)$$

i

c. The vector $y_2 = 1a_1 + 2a_2 - 3a_3$ where a_i are columns i of A. Evaluate the elements of y_2. Attempt to solve for the weights $Ax = y_2$ using Theorem 5. What is the found x and Ax? Why is it wrong to use Theorem 5 to solve for x in this case as compared to Q2b? What should the correct x values be?

d. Normalize the columns of B (Q2a) to have length 1 and assigned it to B_n. Show that the columns of B_n forms an orthonormal set. Solve $y_1 = B_n x_n$ $(where\ y_1$ as in Q2b). How is x_n related to x found in Q2b.

e. Find the projection of y_1 on the columns of B. Calculate the error of approximation for each column. Hence, decide the order of importance of B_n columns to approximate y1.

f. If you are only allow to use 2 columns of B to approximate y_1, which 2 columns should you choose. You will have to exhaustively calculate ALL pairs to determine the least error. **Report the selected columns and residual error obtained.**

Q3. Gram-Schmidt Process

Let $A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix}$ (size mxn matrix)

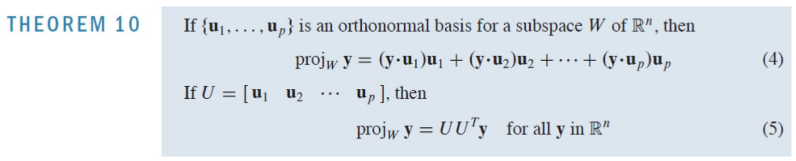
a. In the Jupyter notebook the Gram-Schmidt process for QR factorization has been implemented. **Apply it (either the classical or modified version is okay) on $\boldsymbol{A}$ so that $\boldsymbol{A = QR}$**. Report the Q and R matrixes obtained. Report using the economy size Q (i.e, number of columns of Q == A).

b. What are the characteristics of columns of Q? Explain your answer and compare Q'*Q vs Q*Q'. What is the matrix Q*Q' and its relationship to A?

c. Manually find the ranks of A with row reduction. Compare your result against $numpy.linalg.matrix\_rank()$ from Numpy.

d. Report the rank of the augmented matrix $[A \quad Q]$ using Numpy and compare with that of A. What does it imply?

e. Given an arbitrary vector $\boldsymbol{y}$ and the orthonormal matrix U in R^{mxn}, the orthogonal decomposition aims to first find the projection $\boldsymbol{p}$ of $\boldsymbol{y}$ onto $Col\ U$, and then obtain the orthogonal component $\boldsymbol{z} = \boldsymbol{y} - \boldsymbol{p}.$ There are two ways to obtain the projection $\boldsymbol{p}$. The first one is to take $U = \{\boldsymbol{u_1}, \boldsymbol{u_2}, \ldots, \boldsymbol{u_n}\}$ as a set of basis, find the projection of $\boldsymbol{y}$ onto each $\boldsymbol{u_i}$, and sum them up (Theorem 10). The second one is to calculate the projection matrix P, and apply P to $\boldsymbol{y}$. **Implement your own Python routines for both methods**.


THEOREM 10

If $\{\mathbf{u}_1, \ldots, \mathbf{u}_p\}$ is an orthonormal basis for a subspace W of $\mathbb{R}^n$, then

$$\text{proj}_W\, \mathbf{y} = (\mathbf{y}\cdot\mathbf{u}_1)\mathbf{u}_1 + (\mathbf{y}\cdot\mathbf{u}_2)\mathbf{u}_2 + \cdots + (\mathbf{y}\cdot\mathbf{u}_p)\mathbf{u}_p \tag{4}$$

If $U = [\,\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_p\,]$, then

$$\text{proj}_W\, \mathbf{y} = UU^T\mathbf{y} \quad \text{for all } \mathbf{y} \text{ in } \mathbb{R}^n \tag{5}$$

f. Find the closest point in $Col\ A$ space to $\boldsymbol{y} = (1,2,3,4)$ using your 2 routines. Clearly write the equations you used, the coordinate of the found point, and the error (i.e., the distance between $\boldsymbol{y}$ and the found point).

5. References

[1] https://www.anaconda.com/download/

[2] "A Crash Course in Python for Scientist" http://nbviewer.jupyter.org/gist/rpmuller/5920182

[3] "Scientific Computing with Python", http://nbviewer.jupyter.org/url/atwallab.cshl.edu/teaching/QBbootcamp3.ipynb

[4] "I used Matlab. Now I use Python", Steve Tjoa, Sep 2010. https://stevetjoa.com/305/

[5] http://phillipmfeldman.org/Python/Advantages_of_Python_Over_Matlab.html

[6] "Matlab vs Python: Top reasons to choose Matlab" https://www.mathworks.com/products/matlab/matlab-vs-python.html

[7] http://www.pyzo.org/python_vs_matlab.html