Due Feb 20 by 11:59pm **Points** 100 **Submitting** an external tool
Available after Feb 9 at 12am

MAS3114

MATLAB Assignment 2

MATLAB (https://ufl.instructure.com/courses/497092/modules/items/10717393)

Introduction

This assignment must be completed individually in MATLAB and submitted on Gradescope. It is worth 100 points (+ 5 extra credit points).

Before Starting the Assignment

1. Download the ZIP (https://ufl.instructure.com/courses/497092/files/85288680?wrap=1) (https://ufl.instructure.com/courses/497092/files/85288680/download?download_frd=1) of the template files, upload it to your M-drive, and un-ZIP it. You should have the following files:
   - Assignment2_Output.mlx [Used to Run the Exercises & EC]
   - dependence.m [Pre-written Function]
   - Exercise1.m [Exercise Template File]
   - Exercise2.m [Exercise Template File]
   - Exercise3.m [Exercise Template File]
   - ExtraCredit.m [Extra Credit Template File]
   - transformation.m [Function Template]
2. Enter your Name & UFID on all template files (see below warning)!

> **WARNING:** If you do not enter your name and UF ID correctly, you will receive a zero for this assignment.

MATLAB 2

There are three exercises as part of this assignment. Exercises 1 and 2 are worth 30 points, and exercise 3 is worth 40 points.

Exercise 1 | Exercise 2 | Exercise 3

Linear Dependence — 30 Points / 10 per Part

In this exercise, you will check if the columns of a matrix are linearly dependent using the m-file function, `dependence.m`.

A. Do the following:
   - (1) Create three 3 x 6 random matrices A1, A2, and A3 so that each entry is a random integer between -7 and 7 (using the command `randi`). Then, call `dependence.m` on each matrix to check whether the columns of the matrix are linearly dependent or independent and set the return value to `dep_A1`, `dep_A2`, `dep_A3`.
   - (2) Repeat the process in A(1) for three 6 x 3 random matrices B1, B2, and B3.

B. Observe the result from A(1). For most cases, the columns of A are linearly dependent. Are the columns of any 3 x 6 matrix always linearly dependent? If yes, provide a reason using the concepts in Linear Algebra. If not, provide a non-trivial example.

C. Observe the result from A(2). For most cases, the columns of B are linearly independent. Are the columns of any 6 x 3 matrix always linearly independent? If yes, provide a reason using the concepts in Linear Algebra. If not, provide a non-trivial example.

Onto/One-to-One Linear Transformations — 30 Points / 10 per Part

Consider a linear transformation so that T(**x**) = A**x**. You will check if T is onto and if T is one-to-one by writing a new m-file function.

A. Write a new function `transformation.m` (you may use `dependence.m` as a guide) so that determines if a linear transformation with a standard `A` is onto and/or one-to-one or neither. `transformation.m` should set `transform_type` to one of the following:
   - `both` if the transformation is onto and one-to-one, or
   - `onto` if the transformation is onto but not one-to-one, or
   - `one_to_one` if the transformation is one-to-one but not onto, or
   - `neither` if the transformation is neither onto nor one-to-one.

> **Hint:** If A is m x n, what is rank(A) when T is onto, and what is rank(A) when T is one-to-one?

B. Do the following:
- Call `transformation.m` on matrix A1 to check if the transformation T(**x**) = A1**x** is onto or/and one-to-one. Repeat the process for matrices A2, A3, B1, B2, and B3.
- Create three 3 x 3 random matrices C1, C2, and C3 so that each entry is a random integer between -7 and 7. Call `transformation.m` on matrices C1, C2, and C3.

C. Observe the result in part B, and you may find that T can be both onto and one-to-one only when m = n. Write a report to discuss the following:
- When m < n, can T be one-to-one? Explain why not using linear algebra concepts.
- When m > n, can T be onto? Explain why not using linear algebra concepts.
- When m = n, can T be neither onto nor one-to-one? Explain why or why not using linear algebra concepts.
  If it can be neither, give a non-trivial example where an m x n matrix is neither onto nor one-to-one when m = n, and then call `transformation.m` to verify your claim.
- When m = n: Is it possible that a linear transformation is onto but not one-to-one? Is it possible that a linear transformation is one-to-one but not onto? Explain why it is not both for either case to be true using concepts in linear algebra.

Matrix Operations — 40 Points / 10 per Part

In this exercise, you will first use *for loops* or *while loops* to create a matrix, and then you will practice some matrix operations and verify some properties for matrices. You may need the following commands for this exercise.

- The command `inv(A)` returns the inverse of A when A is invertible.
- The command `A'` returns the transpose of A when all entries in A are real numbers.

You also need to be familiar with both *for loops* and *while loops*.

for loop

The *for loop* executes a group of statements in a loop for a specified number of times.

```
for index = initial_val:end_val
    statements
end
```

```
for index = initial_val:step:end_val
    statements
end
```

Note: The first *for loop* has an implicit step of 1.

while loop

The *while loop* executes an expression and repeats the execution of a group of statements in a loop while the expression is true (see the example in the handout).

```
while expression
    statements
end
```

A. Compare *for loop (i)* and *for loop (ii)* in the template file.
   - Do you have the same output for matrix A?
   - For an n x n matrix A, calculate the number of FLOPs (additions) used in each *for loop*. Express the number of FLOPs in terms of n for each *for loop*.
   - Which *for loop* uses fewer FLOPs?

B. Practice *while loops*: Rewrite both *for loop (i)* and *for loop (ii)* using *while loops* only. (You should get the same output as part A. If not, check your code in *while loops*.)
   Note: Do NOT forget to put semi-colons in commands in loops (causes excessive printing)!

C. Check some properties for matrix multiplications and do the following operations using matrix A from 3A:
   - Create a matrix `B` with size n x (n-2) whose entries are random integers between -7 and 7.
   - Try `A*B` and `B*A` by setting them to `ABBA`.
     Note: Only one command will work. Leave the one that works and give a reason why the other does not work.
   - Create a matrix `C` with size n x n whose entries are random integers between -7 and 7.
   - Calculate `A*C` and `C*A` and set them to `AC` and `CA`, respectively. Are they the same? Why or why not?
   - Calculate `A*eye(n)` and `eye(n)*A` and set to `AI` and `IA`. Are they the same? Why or why not?

D. Check some properties involving the inverse and transpose of a matrix.

- Use matrix A from 3A and find the inverse of A, `inverse_A`, using `inv(A)`. Is A invertible?

- Let . Create matrix D and find the inverse of D using

  - `inv(D)`, and
  - RREF of D -- Write the code using exactly two lines to calculate the inverse of D using `rref`.
- Calculate the following in MATLAB.
  - `inv(inv(D))`, does it equal to D? Generalize your observation.

  - Create matrix .

  - `inv(D*E)`, `inv(D)*inv(E)`, `inv(E)*inv(D)`, which two give you the same result? Finish the generalization.
  - `inv(D')`, `(inv(D))'`, are they the same? Finish the generalization.

> **Extra Credit** (5 points): See instructions in the template file, ExtraCredit.m.

Guidelines and Grading

This assignment must be submitted to Gradescope for grading. You must upload the following files:

- Exercise1.m
- Exercise2.m
- Exercise3.m
- transformation.m
- ExtraCredit.m [Optional]

Before submission, be sure to check that your Name & UFID is on all template files.

After submission, wait till the autograder runs and verify that your submission passes test cases "0.1) Check files were submitted" and "0.2) Check files run" case. If you did not, you may have not uploaded all the necessary files or your files do not run correctly. Make sure to fix any issues and resubmit before the deadline.

Note: You will not see the Autograded portion of your score until after grades are released.

> **WARNING:** There is a late penalty of 25% deduction per day, and therefore, we do not accept any submission after four days.

This tool needs to be loaded in a new browser window

The session for this tool has expired. Please reload the page to access the tool again