MAU11601:Introduction To Programming - Tutorial 4
Basic Matrix Operations

Liam Burke
School Of Mathematics, Trinity College Dublin

October 31, 2022

Q1

Write a MATLAB function *mat_mul* which inputs an $m \times n$ matrix $\mathbf{A}$ and a $n \times p$ matrix $\mathbf{B}$ and returns their product in a new $m \times p$ matrix $\mathbf{C}$. Your code should perform the task on the individual elements of the matrix and should not use the built in MATLAB matrix multiplication given by the * operation ($\mathbf{A}^*\mathbf{B}$).

Recall that the element in row i and column j of $\mathbf{C}$ is given by

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

where a_{ik} is the element in the i^{th} row and k^{th} column of the matrix $\mathbf{A}$, and b_{kj} is the element in the k^{th} row and j^{th} column of the matrix $\mathbf{B}$. Use error checking to ensure the matrices are of compatible sizes for multiplication.

Create a run script *run.m* which loads the matrices $\mathbf{A}$ and $\mathbf{B}$ from the files *A.mat* and *B.mat* in the Tutorial 4 folder on Blackboard. Test your code on these matrices by showing that the 2 norm of the matrix $\mathbf{C} - \mathbf{A}^*\mathbf{B}$ is zero.

Q2

We define the $n \times n$ identity matrix $\mathbf{I}_n$ as a $n \times n$ matrix of zeros with $1's$ along the main diagonal. For example, the 6×6 identity matrix is shown below.

$$\mathbf{I}_6 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{6\times 6}$$

We define the *addition* of two $n \times n$ matrices $\mathbf{A}, \mathbf{B}$ as the matrix $\mathbf{C}$ who's entry in row i and column j is given as $c_{ij} = a_{ij} + b_{ij}$, where a_{ij} and b_{ij} are the elements in row i and column j of the matrices $\mathbf{A}$ and $\mathbf{B}$ respectively.

We define the *scalar multiplication* of a matrix $\mathbf{A}$ with scalar $\gamma \in \mathbb{C}$ as the matrix $\gamma\mathbf{A}$ who's entry in row i and column j is γa_{ij}, where a_{ij} is the entry in row i and column j of the matrix $\mathbf{A}$.

(a) Write a MATLAB function *add_scaled_identity1.m* which takes in an $n \times n$ matrix $\mathbf{A}$ and returns the matrix $\mathbf{D}$ given by

$$\mathbf{D} = \mathbf{A} + 106\mathbf{I}_n \tag{1}$$

The function should *explicitly* construct and store the $n \times n$ identity matrix. Your code should contain one or more *for* loops to perform the task on the individual elements a_{ij} of the matrix, i.e. without using

MATLAB built in matrix operations on the full matrix (such as, e.g. $\mathbf{A}+\mathbf{B}$). Add error checking to ensure that only a square matrix can be input. Test your code on the matrix $\mathbf{A}$ loaded from the previous question and demonstrate you obtain the correct result by showing the 2 norm of the matrix

$$\mathbf{D} - (\mathbf{A} + 106^{*}\mathbf{I}_{40})$$

is zero.

(b) Write another function *add_scaled_identity2.m* which performs the same task as part (a), except this time, it should *implicitly* add the scaled identity matrix i.e. should avoid constructing and storing the identity matrix.

Theory Questions/ Class Discussion

(a) Which method in Question 2, implicit or explicit, may be more preferable in practice? Particularly as the size of the matrices gets very large?
(b) In practice it may not always be possible to use an implicit approach. What feature of problem (1) allows us to use an implicit approach ?