MA2210 Linear Algebra

MATLAB assignment

You will work on two tasks. Task A is interpreting Matlab code already written, while Task B asks you to write your own code, given some sample code for a similar problem.

The early support material for Matlab is embedded in the weekly folder from about week 6. The material directly related to this task is accessed through the Assessment folder in another folder called “Matlab assignment”. **You will need to submit three files to complete this task. See below.**

**Task A.** You will complete this task after reading through the page labelled “Gram Schmidt” in the Matlab assignment folder. You will need to run through the code in the file master.m as suggested in the embedded video.

To help you do this I suggest you work through the Gram Schmidt procedure as outlined in the week eight folder (recording 3b for the week, associated with section 6.3 of the text). Do the process, by hand, on the following three unit vectors starting with as the first vector in the orthogonal basis:

* Note for these hand calculations it is easiest to leave vector components in surds, rather than write decimal approximations of them.
* What is the relationship between the vectors listed above and the matrix U at line 27 of the code in master.m?Answering this question helps you unpack what is being done in the code.

**Your task is to write one to two pages** indicating what the code in the file master.m does. **You need to indicate that you have interpreted the code in terms of the linear algebra concepts developed in the MA2210.**  You need to indicate how those concepts appear in the code, indicating the line numbers over which those concepts appear. I expect you to mention lines 11 to 13, line 47, line 55, line 59, lines 80 to 120. There are no new linear algebra concepts developed in the remainder of this code. The file gramSchmidt\_errorcheck.m does introduce one more characteristic of the Gram Schmidt procedure, which you should identify.

**Task B.** After working through the pages labelled “Singular Value Decomposition” and “SVD for image compression” we ask you to use MATLAB to do two items that when combined present a reconstructed version of an image using the SVD.

1. Write a MATLAB function called SVDcompress that will take two arguments: a filename and the number of singular values with which to reconstruct the image in the file. This time the function will reconstruct a colour image, rather than a greyscale image. The function will be defined like this:

function outimage = SVDcompress(filename, Nretain)

While you can use your code for any image, for this task the filename will be one of

* + prague-astronomical-clock-detail-871291743639AGq.jpg
  + boat-in-caribbean-14884763094mZ.jpg

1. Display images that are constructed using 5, 10, 50 and 100 singular values from the original dataset.

You should be able to base your code for this task on that given in compress.m. The array outimage contains the information that is needed to generate the reconstructed image, which will be done outside the function. It will have three indicies, the last one, colour\_ind, takes the value 1, 2 or 3 and caters for the three different base colours that are combined to form the image. Each base colour will have an SVD computed for it. The SVD is truncated by taking Nretain singular values and outimage is calculated by multiplying the retained components of the SVD: outimage(:,:,colour\_ind) = Uret\*Sret\*Vret'. Here Uret, etc are the retained components of the original U, etc.

**You need to include your code, (appropriately commented) and the output from the code (as in step 2 above). The code will be submitted as a** .m **file, while the output can be submitted as a pdf file (Matlab figures can be saved as pdf files).**