Homework 3 - Sparse Matrices and FFT

Submission open until 11:59:59pm Thursday November 12, 2020

(1) The time evolution of the vorticity $\omega(x, y, t)$ and the streamfunction $\psi(x, y, t)$ are given by the governing equations:

$$\omega_t + [\psi, \omega] = \nu \nabla^2 \psi \tag{1}$$

where $[\psi, \omega] = \psi_x \omega_y - \psi_y \omega_x$ and $\nabla^2 = \partial_x^2 + \partial_y^2$. The streamfunction satisfies

$$\nabla^2 \psi = \omega. \tag{2}$$

Boundary Conditions: Assume periodic boundary conditions for both the vorticity and the streamfunction.

(a) Using the `spdiags` command, generate the three matrices $A = \partial_x^2 + \partial_y^2$, $B = \partial_x$ and $C = \partial_y$ which take derivatives in two dimensions. Use the discretization of the domain and its transformation to a vector as in class.

ANSWERS: With $x, y \in [-10, 10]$, $n = 8$ save the matrices A, B and C as A1, A2 and A3 respectively.

NOTE: You can't write out sparse matrices to ASCII files so be sure to first make the matrices full, i.e. you can use `A=full(A)` in MATLAB and it will turn a sparse matrix into a full matrix.

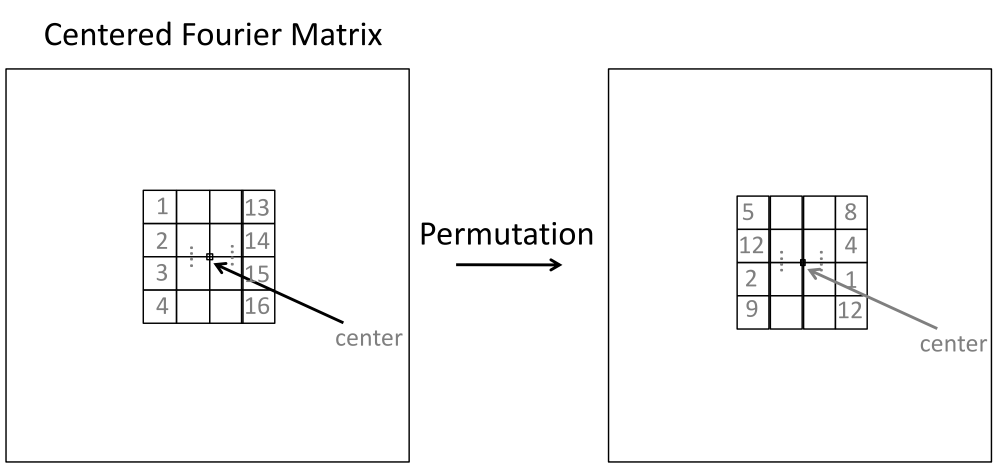

Figure 1: Encryption of the Fourier matrix.

(2) Fourier Transform (FFT) of an image is sometimes used for image encryption. The idea for the encryption is to divide the image in Fourier space into blocks. Permuting the blocks and applying the inverse FFT will result in a scrambled image instead of the original one. The image can be decrypted with the inverse permutation (key). The motivation for the encryption is based on the fact that guessing the permutation takes a long time, since each guess involves an inverse FFT operation. Most images include low frequency components so it is enough to choose squared blocks in the center of the (centered) Fourier matrix of the

image and permute them as shown in Fig. 1.

(a) Write a decryption algorithm that gets as an input a (centered) **Fourier** matrix 400×400 and an inverse permutation vector that specifies how 16 blocks of 20×20 should be permuted back to their original placement. The algorithm will permute the blocks, shift the Fourier matrix from the center to the corners and perform inverse FFT to recover the image. Load the Fourier matrix and the permutation vector from the files `Fmat.mat` and `permvec.mat`. **Do not turn in these files to gradescope** and do not assume that on gradescope the matrix/vector will have the same elements, thereby plan your algorithm to be general.

ANSWERS: Save the absolute values of the decrypted (centered) Fourier matrix as A4 and the reconstructed image matrix (absolute value without uint8) as A5.

NOTE: To plot the image matrix (for checking how your decryption works) use the `set(gcf,'colormap',gray);imagesc(...);` or `imshow` commands. For the plotting, don't forget to take absolute values of the reconstructed image and to use `uint8`. The `ind2sub` command can be useful for indexing.