Program 1 Foundations of Computational Math 2 Spring 2019

Due date: 11:59PM on Monday 4 February 2019

Problem 1

The Codes: Implement the following codes:

1.1. Using the unitary form of F and F^H matrices defining the DFT and IDFT respectively, implement a routine that takes as input, v, a complex vector of length $n = 2^t$, and outputs, w, the vector resulting from the DFT of v or the IDFT of v, i.e., either $w = Fv$ or $w = F^H v$. Your code will require complex arithmetic and variables. This type should be supported in your Fortran or C as a basic data type or as a template class in C++. This code should not use the FFT idea, i.e., it should work with the matrix vector product definition directly. **It must be efficient in computations and in storage. Make sure you exploit the structure of the matrix and explain clearly how your computations and storage are handled.**

1.2. Implement an FFT and IFFT routine that takes as input, v, a complex vector of length $n = 2^t$, and outputs, w, as either $w = Fv$ or $w = F^H v$. As with the DFT/IDFT routine, explain clearly how your computations and storage are handled to maximize efficiency.

1.3. Implement a routine that solves linear systems of equations involving an $n \times n$ non-singular circulant matrix with $n = 2^t$, i.e., solve $C_n x = b$.

1.4. Implement the image processing routine described in the Tasks section using the routines above.

The Tasks:

1.1. Provide systematic evidence of the correct execution of your routines.

1.2. Provide detail on how efficiency in computations and storage is achieved.

1.3. Determine how to time the execution of your codes on your machine and investigate the performance benefit of the FFT/IFFT implementation over the DFT/IDFT implementation for the tasks required. Note you should explain the accuracy of your timer, the type of "time" reported, the overhead of reading the timer, the design of your timing experiments to amortize overhead and assure accuracy, the design of your timing experiments to avoid or account for timing effects of system activity such as virtual memory servicing.

1.4. Let $C \in \mathbb{R}^{n\times n}$ be a circulant matrix defined by the elements $\alpha_0, \ldots, \alpha_{n-1}$ in its first row. Let $\mathcal{I} \in \mathbb{R}^{n\times n}$ and $M \in \mathbb{R}^{n\times n}$ be two matrices whose entries can be interpretted as pixels in $n \times n$ images. The image in M is a blurred version of the image in $\mathcal{I}$ give by the blurring process specified by C as

$$M = C^T \mathcal{I} C.$$

Use your circulant matrix and Disrete Fourier (inverse) transform routines to recover the deblurred image $\mathcal{I}$ given M and C. The class homework webpage has links to files containing M, $\mathcal{I}$ and C matrices that you can use as a test case to debug your code. Your code must therefore be able to read matrices from files and output matrices in a manner that allows it to be processed by Matlab or Python or your prefered graphics system for display. You may find it convenient to use Julia for this programming assignment.

1.5. Consider sending an image $\mathcal{I}$, an $n \times n$ matrix, via radio signal. To encode the image given in the spatial domain, the frequency spectrum S given by

$$S = \mathcal{F}\mathcal{I}\mathcal{F}^H$$

is often used. Notice that S is a complex matrix. Suppose we have a modulator M, that can encode S into n radio signals $\{X_i\}_n$ without any loss of accuracty and a demodulator M^{-1} that convert the $\{X_i\}_n$ back to S exactly, i.e.

$$M(S) = (X_1, \cdots, X_n)$$
$$M^{-1}(X_1, \cdots, X_n) = S$$

Therefore, no noise was introduced by modulation-demodulation.

Now suppose there were strong signals N_i that corrupted our signals X_i and we received

$$\tilde{X}_i = X_i + N_i$$
$$\tilde{S} = M^{-1}(\tilde{X}_1, \cdots, \tilde{X}_n)$$

Suppose these N_i were combinations a few simple harmonics so they introduced spikes into S, i.e.

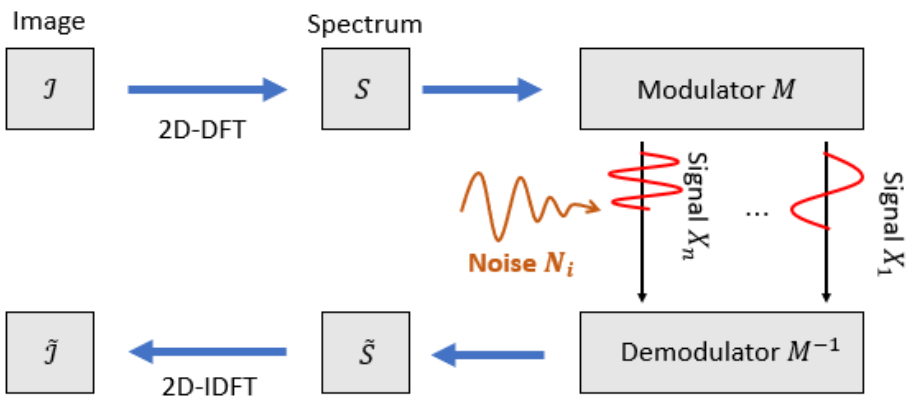
$$\tilde{S} - S = n_{i_0,j_0} E_{i_0,j_0} + n_{i_1,j_1} E_{i_1,j_1} + \cdots$$

where $E_{i,j}$ has zero entry everywhere except (i, j). The number of spikes and their positions are assumed unknown. Here is an illustration. Numerical noise due to finite precision is ignored, i.e., the effect of the corruption is the problem of interest.

Suppose, in addition, we know that $|n_{i,j}|$ are at the level of the total energy of S:

$$|n_{i,j}| \approx 0.01 \times E \sim E$$

where $E = \sqrt{\mathrm{Tr}(SS^H)}$.



Given a $\tilde{\mathcal{I}}$, design an algorithm to recover $\mathcal{I}$ as accurately as possible. Explain how you detect the spikes and what you would do after finding a spike to create a new spectrum. Investigate multiple detection and restoration strategies. You may find it useful to check the energy distribution of the spectrum of some normal pictures before you design the way of detecting spikes and repairing the corrupted spectrum.

After you have submitted your solution for this programming assignment you will be asked to demonstrate your code on other source/blurred/corupted image pairs in a meeting with the TA. For grading purposes, you will be required to compress and reconstruct an image and to compute the error for source images not in the provided test set; and to successfully produce and display a deblurredor recovered image $\mathcal{I}$ given M and C pairs not in the provided test set. Make sure your code is designed to facilitate this activity.

Submission of Results

Expected results comprise:

- A document describing your solutions as prescribed in the notes on writing up a programming solution posted on the class website.
- The source code, makefiles, and instructions on how to compile and execute your code including the Math Department machine used, if applicable.
- Code documentation should be included in each routine.
- All text files that do not contain code or makefiles must be PDF files. **Do not send Microsoft word files of any type**.

These results should be submitted by 11:59 PM on the due date. Submission of results is to be done via FSU Dropbox at `https://dropbox.fsu.edu`. Drop the files off for Zhifeng Deng using his MyFSU email `zd16d@my.fsu.edu`. (Note: **do not use** `zdeng@math.fsu.edu` **to drop off the files**) You should login to FSU Dropbox using your MyFSU login. If for some reason you cannot use FSU Dropbox please email the files to Zhifeng at the email address above.