Revised 1 February 2017

Project # 2: Convolution and Discrete Fourier Transform

In this lab you will use MATLAB to study the following topics:

- Fourier Matrix and Sampling
- Applications of the Discrete Fourier Transform
- Circulant Matrices and Circular Convolution
- Downsampling and Fast Fourier Transform

Preliminaries

Reading: Before beginning your MATLAB work, study Sections 1.6, 1.7, and Chapter 2 of the textbook.

m-files: For Question 1(b) you will need the m-file `fftgui.m` (*Finite Fourier transform graphic user interface*). Download this file from the Math 642:357 course page to your directory (folder) for your MATLAB work. Set the MATLAB `Path` to include this directory.

Diary File: You will need to record the results of your MATLAB session to generate your lab report. Create a directory (folder) in your computer workspace to save your MATLAB work in. Then use the `Current Directory` field in the desktop toolbar to change the directory to this folder. Now type

```
diary lab2.txt
```

followed by the `Enter` key. Now each computation you make in MATLAB will be saved in your directory in a text file named `lab2.txt`. You can then edit this file using your favorite text editor (such as Emacs, Notepad, or WordPad). When you have finished your MATLAB session you can turn off the recording by typing `diary off` at the MATLAB prompt (*don't do this now*).

Lab Write-up: Now that your diary file is open, type the comment line

```
% Math 357 MATLAB Project #2
```

at the MATLAB prompt. Type `format compact` so that your diary file will not have unnecessary spaces. Put labels to mark the beginning of your work on each part of each question, so that your edited lab write-up has the format

```
% Question 1 (a) ...
      ⋮
% Question 1 (b) ...
```

and so on. Be sure to answer all the questions in the lab assignment. Insert comments in your diary file as you work through the assignment.

Random Seed: When you start your MATLAB session, initialize the random number generator by typing

```
rand('seed', abcd)
```

where *abcd* are the last four digits of your Student ID number. This will ensure that you generate your own particular random vectors and matrices.

BE SURE TO INCLUDE THIS LINE IN YOUR LAB WRITE-UP

The lab report that you hand in must be your own work. The following problems use randomly generated matrices and vectors, so the matrices, vectors, and graphs in your lab report will not be the same as those of other students doing the lab. Sharing of lab report files is not allowed in this course.

Question 1.Fourier Matrix and Sampling

This question explores the results in Sections 2.2 and 2.3 of the textbook.

(a) Fourier Matrix: The $N \times N$ Fourier matrix F_N is symmetric; its kth column (or row) consists of the consecutive powers 1, $\omega^{-(k-1)}$, $(\omega^{-(k-1)})^2$, ..., $(\omega^{-(k-1)})^{(N-1)}$, where $w = \mathrm{e}^{2\pi \mathrm{i}/N}$. If $\mathbf{y} \in \mathbf{C}^N$ is a column vector, then the *Discrete Fourier Transform* of $\mathbf{y}$ is the vector $Y = F_N\mathbf{y}$. In particular, taking $\mathbf{y} = \mathbf{e}_k$ as the kth standard basis vector, we obtain the normalized vector $\mathbf{u}_k = (1/\sqrt{N})F_N\mathbf{e}_k$. The vectors $\{\mathbf{u}_1, \ldots, \mathbf{u}_N\}$ are the orthonormal *Fourier basis* for $\mathbf{C}^N$, and the matrix $(1/\sqrt{N})F_N$ is unitary.

The built-in MATLAB function `fft` takes the Discrete Fourier transform of each column of a matrix argument. Hence `F = fft(eye(N))` generates the matrix F_N, since `eye(N)` generates the $N \times N$ identity matrix (whose columns are the vectors $\mathbf{e}_k$). Check this by the following commands (use the semicolon ; so that the large matrix `F8` doesn't appear on screen or in your diary).

```
F2 = fft(eye(2))
F4 = fft(eye(4))
F8 = fft(eye(8));
```

Compare the matrices `F2` and `F4` with the examples of Fourier matrices in Section 2.3 of the textbook. Use MATLAB to check that the matrix $(1/\sqrt{8})F_8$ is unitary by calculating the distance between the matrix UU^{H} and the identity matrix:

```
norm((1/8)*F8*F8' - eye(8))
```

Remember that the MATLAB notation `A'` means the *conjugate transpose* matrix A^{H}. The computed norm should be less than 10^{-15}, which we consider as zero for numerical purposes.

(b) Sampling and the Nyquist Point: If `y` is a column vector with complex entries, then the MATLAB command `fftgui(y)` generates a window with four subplots: The top two plots indicate the real and imaginary parts of `y`, and the bottom two indicate the real and imaginary parts of the discrete Fourier transform `Y = fft(y)`, all plotted as stem graphs (lollipops). Try this with a vector of length 16 with a single 1 in the first entry and the other entries zero:

```
y = zeros(16, 1); y(2) = 1; fftgui(y)
```

. Here the notation `y(k)` means that we are following the MATLAB indexing convention, numbering the entries from 1 to 16.

> *(i)* Describe the plots of `fft(y)` in each case in terms of sampled sine and cosine waves.

Use the uparrow to repeat this experiment, now taking `y(2) = 1, y(3) = 1` and the other entries zero. In these plots consider the points on the horizontal axis numbered from 0 to $N-1$, where N is the length of `y`. The point numbered $N/2$ on the frequency axis is called the *Nyquist point*. Hence when $N = 16$, then the Nyquist point is the 9th dot from the left.

> *(ii)* What symmetries around the Nyquist point do the graphs of `real(fft(y))` and `imag(fft(y))` have for each of your plots?

Check that equation (2.28) from the textbook holds for your examples (view your plots as periodic functions of period $N = 16$).

Now close the `fftgui` figure window and enter the command `fftgui` at the MATLAB prompt. The figure window will reappear with all plots initialized to `y = zeros(32, 1)`. You can use the mouse to move any of the points in the upper (or lower) two plot windows, Use the mouse in the `real(y)` plot window to draw a random waveform with large and small oscillations. Since the period is now $N = 32$, the Nyquist point is

the 17th dot from the left. Check that the graphs of `real(fft(y))` and `imag(fft(y))` have the symmetries in equation (2.28) around this point.

Question 2. Applications of the Discrete Fourier Transform

(a) Touch-tone Dialing: Telephone dialing is an example of everyday use of Fourier analysis. The basis for touch-tone dialing is the Dual Tone Multi-Frequency (DMF) system. The telephone dialing pad acts as a 4–by–3 matrix. Associated with each row and column is a frequency. Each digit in a telephone number is encoded by a signal which is the sum of two sine waves whose frequencies are the row and column frequencies associated with the digit. Here is the matrix, with the row frequencies indicated on the side and the column frequencies on the bottom:

697	**1**	**2**	**3**
770	**4**	**5**	**6**
852	**7**	**8**	**9**
941	*	**0**	#
	1209	1336	1477

For example, 1 is encoded by the pair of frequencies 697, 1209, while 3 is encoded by the pair 697, 1477. Create the following function m-file to generate dial tones:

```
function y = f(j,k)          % dial tone for button in row j and column k
fr = [697  770  852  941];   % row frequencies
fc = [1209  1336  1477];     % column frequencies
Fs = 32768;                  % sampling rate 2^15
t = 0:1/Fs:0.25;             % vector of sample points in time interval (0, 0.25)
y1 = sin(2*pi*fr(j)*t);      % sine wave of row frequency for button
y2 = sin(2*pi*fc(k)*t);      % sine wave of column frequency for button
y = (y1 + y2)/2;             % tone generated by button
```

(be sure to put semicolons after each MATLAB command, so that these long vectors are not displayed on the screen). Save this function file under the name `tone.m`. Test it by typing

```
y = tone(1,1); Fs = 2^15;   sound(y, Fs)
```

If the sound on your computer is turned on, you should hear a 1/4 second touch-tone for the 1 button (if you are working in one of the Rutgers Computer Labs, you will have to plug your headphones into the computer to hear the sound). If you get an error message, check your typing and also check that you have set the MATLAB path.

Now use MATLAB to generate a random integer r between 0 and 9 by

```
r = fix(10*rand(1))
```

(be sure you have initialized the random number seed using the last four digits of your ID number). Determine the row j and column k of the telephone dial pad for your number r, and use MATLAB to generate the corresponding tone

```
y = tone(j,k);   sound(y, Fs)
```

(here you must replace j and k by the appropriate row/column numbers). Plot 16 milliseconds of the tone wave-form by

```
t = (1/Fs)*[1:512];
figure, plot(t, y(1:512))
```

The wave-form will be a superposition of the sine waves of two different frequencies; you cannot see the frequencies in the graph, however. Click on `Insert` in the Figure toolbar, label the horizontal axis as time (seconds) and put a title for the figure Fig. 1: DMF Tone for r (here replace r by your particular randomly-generated number). Save and print the graph.

Now you will use the Discrete Fourier Transform to find the pair of frequencies in your dial tone, and hence the number r that the tone encodes. Create the following MATLAB m-file that will plot the absolute value of the Fourier transform Y of a signal y as a function of frequency over a specified range of frequencies:

```
function powergraph(y, Fs)
n = length(y);
Y = fft(y);
pow = abs(Y);
pmax = norm(pow, inf);
freq = (0:n-1)*(Fs/n);
figure, plot(freq,pow); axis([500 1700 0 pmax])
```

Save this function file under the name `powergraph.m`. Here `pmax` is the *maximum* of the absolute values of the entries in the vector `pow` (this number is called the ℓ^∞ norm of `pow`).

At the MATLAB prompt type `powergraph(y, Fs)` (this uses the variables `y` (the signal) and `Fs` (the sampling frequency) that were already defined). The horizontal axis in the graph are the values of `freq` (the frequency), and the range is the DTMF frequency range. The vertical axis shows the *intensity* of the Fourier transform at each frequency (the square of the intensity is usually called the *power*). The graph should show two sharp peaks, corresponding to the two frequencies used to encode your number. Use the zoom feature (the magnifying glass in the toolbar) to determine the two frequency peaks. Confirm that the pair of frequencies are the correct ones for your original random number. Then restore the graph to the original version and put labels on the graph indicating the frequency peaks, label the horizontal axis frequency (Hertz), and title the figure as Fig. 2: Powergraph of DMT Signal for r (replace r by your particular randomly-generated number). Save and print the graph.

(b) Analyzing a Train Whistle: Type `load train` at the MATLAB prompt. This will give you a long vector `y` and a scalar `Fs` whose value is the number of samples per second. The time increment is `1/Fs` seconds. The command

```
sound(y, Fs)
```

will play the signal – one short and one long pulse of a train whistle. Generate the time vector `t` and plot the signal as a function of time by the commands

```
n = length(y); t = (1/Fs)*[1:n];
figure, plot(t,y)
```

The graph clearly shows the two short-long whistle pulses, but it gives no information about the frequencies of the tones in the whistle. Label the horizontal axis time (seconds) and add the title Fig. 3: Train Whistle. Save the graph and print a copy.

Now use the function file `powergraph` that you wrote in part **(a)** to obtain part of the other graph in Example 2.10 of the textbook. Notice that the powergraph gives detailed frequency information about the whistle, but it does not show that there were two pulses (no time information). Zoom in on the peaks in the powergraph to find approximate values of the three main frequencies less than 1,250 Hertz in the whistle. The characteristic mournful sound of the whistle is due to the fact that the ratio of each pair of these frequencies is very close to a rational number p/q, where $p < q$ are small positive integers (5 or less), and when sounded together these frequencies give a minor triad. Find these ratios (one of them is $3/5$; in musical terminology a major sixth). Put labels on the graph indicating the frequency peaks and the best choice of such ratios p/q for the ratios of frequencies of the three pairs of these peaks. Label the horizontal axis frequency (Hertz), and title the graph as Fig. 4: Powergraph of Train Whistle. Save and print the graph. See Example 2.10 of the textbook for more discussion of this example.

Question 3. Circulant Matrices and Circular Convolution

This question explores the results in Sections 2.4 and 2.5 of the textbook.

(a) Shift Operator: Create the following MATLAB function m-file that generates the $n \times n$ *shift matrix* S:

```
function S = shift(n)
S = zeros(n,n); S(1,n) = 1;
for k = 1:n-1
S(k+1,k) = 1;
end
```

Save this function file as `shift.m`. Then test it by typing `S = shift(4)`. You should get the 4×4 shift matrix (the 3×3 shift matrix is shown in Example 2.5 of the textbook). Generate a random integer vector $\mathbf{v}$ by

```
v = fix(10*rand(4,1))
```

Then calculate `S*v` to see that S shifts the components of $\mathbf{v}$ cyclically.

From the theory of the shift matrix `(1/4)*F4'*S*F4` is a diagonal matrix, where F_4 is the Fourier matrix from Question 1 – remember that the conjugate-transpose A^H is given in MATLAB by `A'` (see Theorem 2.3 in the textbook). Check this by MATLAB. From this calculation, what are the eigenvalues of S?

(b) Circulant Matrices: Use your random vector `v` from **3(a)** and powers of the matrix `S` to construct a 4×4 *circulant matrix* `C` whose first column is `v` (see Theorem 2.2 in the textbook).

(i) From the theory of circulant matrices `(1/4)*F4'*C*F4` is a diagonal matrix. Check this by MATLAB.

(ii) From the MATLAB calculation, what are the eigenvalues of `C`?

(iii) How are the eigenvalues of `C` obtained from the eigenvalues of S? Give the general algebraic relation (see Theorem 2.4 of the textbook). Then carry out a hand calculation to obtain the same values as in (ii)

(c) Circular Convolution: Consider the vector space of real periodic signals of length 32. The value of the signal $\mathbf{y}$ at discrete time j is denoted by $\mathbf{y}[j]$. The periodicity means that $\mathbf{y}[j+32] = \mathbf{y}[j]$ for all integers j. For purposes of calculation we identify $\mathbf{y}$ with the column vector

$$\begin{bmatrix} \mathbf{y}[0] \\ \mathbf{y}[1] \\ \vdots \\ \mathbf{y}[31] \end{bmatrix} \in \mathbf{R}^{32}$$

(see Section 2.4 of the textbook).

Let C be the *three-step moving average* operator defined by

$$C\mathbf{y}[j] = (1/3)(\mathbf{y}[j-1] + \mathbf{y}[j] + \mathbf{y}[j+1]) \quad \text{for } j = 0, 1, \ldots, 31\,. \tag{1}$$

Notice that when $j = 0$ or $j = 31$ the periodicity of $\mathbf{y}$ is used in this definition as in Section 2.4 of the textbook. We will study the effect of C and C^2 on a signal.

(i) Find the periodic function $\mathbf{f}[j]$ of period 32 such that $C\mathbf{y}[j] = (\mathbf{f} \star \mathbf{y})[j]$, where $\star$ means *circular convolution* (see Definition 2.4 in the textbook). The function $\mathbf{f}$ is called the *filter*.

(ii) Write out by hand an explicit symbolic formula for $C^2\,\mathbf{y}[0]$ in terms of the values $\mathbf{y}[0], \ldots, \mathbf{y}[31]$. How many sample values of $\mathbf{y}$ are used to calculate $C^2\,\mathbf{y}[0]$?

(iii) Use your formula from (ii) and the shift-invariance property to obtain an explicit formula for $C^2\mathbf{y}[j]$ from the formula for $C^2\mathbf{y}[0]$.

Use MATLAB to create a 32×32 circulant matrix `C` which acts on column vectors `y` according to (1), as follows: Generate the shift operator `S = shift(32)` and the identity matrix `I = eye(32)`; then write `C` as a linear combination of `I` and powers of `S` (see Theorem 2.2 in the textbook); in this case the inverse of `S` is `S^31`.

Note: *Put ; at the end of each* MATLAB *command so that the matrices* `S, I, C` *do not display on screen.*

Generate a time vector `t` and random signal `y` and plot the signal signal as a solid red line:

```
t = [0:31]; y = rand(32,1);
figure, plot(t,y, 'r-'); hold on
```

Compare the signal with its three-step moving average (plotted in blue with a dotted line):

```
plot(t, C*y, 'b:')
```

Now compare with a double three-step averaging (plotted in green with a dashed line):

```
plot(t, C*C*y, 'g--')
```

Notice that the averaging process acts as a *low-pass filter* to reduce the variability in the signal: the graph of $C\,\mathbf{y}$ has less oscillation than the graph of $\mathbf{y}$, and the graph of $C^2\,\mathbf{y}$ has even less oscillation. Put the title Fig. 5: Random Signal with 3-Step Moving Averages on the graph. Insert arrows and labels to identify the signal `y` and the moving averages `C*y` and `C^2*y`. Save and print the graphs.

Question 4. Fast Fourier Transform

This question explores the results in Section 2.6 of the textbook.

(a) Downsampling: The first step in the FFT is to split a signal vector $\mathbf{y}$ (of length $2n$) into two vectors $\mathbf{y}_{\text{even}}$ and $\mathbf{y}_{\text{odd}}$, each of length n. This step is called *downsampling*. The following function m-file generates the permutation matrix P_{2n} that does this:

```
function P = downsamp(n)
P = zeros(2*n, 2*n);       % Create 2n by 2n matrix of zeros
for j = 1:n
    P(j, 2*j - 1) = 1;     % Sort entries 1,3,...,2n-1 into rows 1,...,n
    P(n + j, 2*j) = 1;     % Sort entries 2,4,...,2n into rows n+1,...,2n
end
```

Create and save this m-file under the name `downsamp.m`. Test it by creating a random vector `y = fix(20*rand(8,1))` and the 8×8 downsampling matrix `P = downsamp(4)`. Calculate `P*y` and check that the entries in positions 1, 3, 5, and 7 of `y` are the first four entries in `P*y`, and the entries in positions 2, 4, 6, and 8 of `y` are the last four entries in `P*y`.

(b) Block decomposition of the Fourier matrix: The block form in the FFT uses a diagonal matrix D_n (see equation (2.33) in the textbook), which can be generated by the following function m-file:

```
function D = fftdiag(n)
D = zeros(n,n);
w = exp(pi*i/n);
for j = 1:n
  D(j, j) = w^(1-j) ;
end
```

Create and save this m-file under the name `fftdiag.m`.

With the two function m-files you have made you can now create the matrix F_4^{H} from the matrix F_2^{H}, and then the matrix F_8^{H} from F_4^{H}. Type

```
P4 = downsamp(2); D2 = fftdiag(2); F2 = fft(eye(2));
F4H = [F2,  D2*F2; F2,  -D2*F2]*P4
```

Compare `F4H` with the matrix `fft(eye(4))'`. Are they the same?

Now repeat the process to generate the 8×8 matrix F_8^{H}. First generate `P8 = downsamp(4)` and `D4 = fftdiag(4)`. Use these, the matrix `F4H` you just generated, and equation (29) in the supplementary notes to obtain a matrix `F8H`. Compare your `F8H` with the MATLAB-generated matrix F_8^{H} by calculating

```
norm(F8H - fft(eye(8))')
```

This should be of size 10^{-15}, thus zero to the limits of machine computation.

(c) Speed comparison: Generate the Fourier matrix F_N for $N = 2^{12} = 4096$ and a random vector $\mathbf{x} \in \mathbf{R}^{4096}$ by the following commands:

```
F = fft(eye(4096)); x = rand(4096,1);
```

Calculate the computation time to obtain Fourier transform of $\mathbf{x}$ in two ways: first by direct matrix multiplication and then by the fast Fourier transform (be sure to type the semicolons as indicated, so the vectors will not appear on the screen), as follows.

```
tic; F*x; matrixtime = toc
tic; fft(x); ffttime = toc
speedup = floor(matrixtime/ffttime)
```

You can compare this observed speedup using the FFT with the theoretical prediction. If $N = 2^k$, then computing the matrix $\times$ vector `F*x` requires $N^2 = 2^{2k}$ scalar multiplications. Computing `fft(x)` needs at most $k2^{k-1}$ scalar multiplications by Equation (2.35) of the textbook. The theoretical speed advantage of the FFT over the plain (matrix $\times$ vector) multiplication, in terms of the scalar multiplications required, is thus

$$2^{2k}/[k2^{k-1}] = 2^{k+1}/k\,. \qquad (\star)$$

A similar speed advantage holds for the scalar additions required (see Section 2.6 of the textbook). Scalar multiplications require much more computing time than scalar additions, so it is the reduction in the number of multiplications that is crucial for the speedup given by the FFT. Compute the theoretical speed advantage $(\star)$ for the value of k that you have used, and compare it with the observed value `speedup`.

Final Editing of Lab Write-up: After you have worked through all the parts of the lab assignment, you will need to edit your diary file. Remove all errors and other material that is not directly related to Questions 1–4. Your write-up should only contain the keyboard input and the MATLAB output (including Figures 1-7), together with the answers to the questions that you have written.

Preview the document before printing and remove unnecessary page breaks and blank space. Put your name and four-digit ID number on each page. (If you have difficulty doing this using your text editor, you can write this information by hand after printing the report.)

Do not include the codes for the m-files in your lab writeup, but be sure to save these m-files on your disc for future use.