Laboratory 4: The Discrete Fourier Transform (DTFT)

1. Objectives

- Learn how to compute and interpret the discrete-time Fourier Transform (DTFT) of a DT signal
- Gain real-world experience by examining the frequency distribution within a cardiac signal recording.

2. Introduction

In the previous labs you learned how to investigate the properties of discrete-time signals and systems in the time-domain. Time-domain representations at first appear the most intuitive way to represent signals; they are what we see on an oscilloscope screen. Yet historically the development of powerful control algorithms that guided military manufacturing for World War I were enabled by a fundamentally new way of looking at signals that began gaining popularity at the turn of the 20^{th} century called Frequency Domain Analysis. This is a generic term including Fourier Series, Fourier Transforms, and Laplace Transforms for continuous-time systems, and the Discrete Time Fourier Transform (DTFT), Discrete Fourier Transform (DFT), and Z-transform for discrete-time systems.

In EE230 you already studied two types of transforms that are commonly used to investigate continuous-time signals x(t). In the first half of the semester you studied the Laplace Transform X(s), where the variable s is complex and often graphed on the complex s-plane. In the second half of the semester you studied the trigonometric Fourier Series in which you decomposed a periodic signal x(t) into a sum of sines and cosines of integer multiples k of the fundamental frequency, which were scaled by coefficients a_k and b_k respectively. A less-intuitive but simpler representation (from the perspective of only needing one set of coefficients) that you learned was the complex exponential representation in which the signal x(t) was represented by a sum of complex exponentials $c_k\, e^{jkt/T}$, where again the information was contained in the coefficients c_k that were multiplied by complex exponents having frequencies that are integer multiples of k times the fundamental frequency 1/T. At the end of the course you learned about the more general Fourier Transform $X(e^{j\omega})$ that could be applied to signals that were not periodic, and that intuitively showed how much power from every frequency contributed to a signal. An example you used was the Bode Plot, which is the Fourier Transform of the impulse response h(t) of a system (although you learned about it much earlier in the course as what it physically represents: the magnitude scaling and phase offset that a system applies to a sinusoidal input). The CT transforms of the Fourier Transform, complex exponential Fourier Series, and Laplace transform are in DT called, respectively, the DTFT, the DFT, and the Z Transform. This lab examines the DTFT.

3. Discrete Time Fourier Transform (DTFT)

A. Definition and intuition of the DTFT

The CT transforms you learned about in EE230 have one-to-one correspondence with DT transforms, although their names have been changed to protect the innocent. The DTFT is like the CT Fourier Transform; it is defined and calculated as

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n} \qquad \text{(eqn 1)}$$

Just like its CT counterpart, the DTFT is defined over all values of ω, both positive and negative, and both integers and non-integers. Does it make sense to have a non-integer value of ω for a DT signal? Sure; you can generate a signal x[n] = cos(1.5 n). If you graphed the DTFT of this signal you would find large positive peak at values ω = ± 1.5. (Why at -1.5 too? Because it is the same signal as cos(-1.5 n) since cos is an even function. The DTFT of sin(1.5 n) would have a large negative value at ω = -1.5.) As an undergraduate I was bothered by the notation $X(e^{j\omega})$. Why not simply say $X(\omega)$? While it would be simpler, the unusual notation $X(e^{j\omega})$ is used because
1) for historical reasons; 100 years ago this was the notation used by mathematicians
2) to remind the reader that this is the Fourier Transform, not the Z transform,
3) electrical engineering is hard.

B. Properties of the DTFT

- The DTFT $X(e^{j\omega})$ is complex. It therefore requires two graphs to be represented; either in rectangular coordinates real($X(e^{j\omega})$) which Mitra calls $X_{re}(e^{j\omega})$ and imag($X(e^{j\omega})$) which Mitra calls $X_{im}(e^{j\omega})$, or in polar coordinates the magnitude spectrum $|X(e^{j\omega})|$ and phase spectrum $\angle X(e^{j\omega})$. The most important of these graphs is the magnitude spectrum, which shows the frequency distribution of signal energy.
- The DTFT $X(e^{j\omega})$ is a periodic continuous function in ω with a period 2π. (Why? What is the difference between cos(1.5) and cos(1.5+2π)? Then, what is the difference between x[n] = cos(1.5n) and cos (1.5n + 2πn)?). Furthermore, it only needs to be evaluated between 0 and π, because the DTFT between π and 2π is just the reflection of the value between 0 and π; for instance, see below. Most commonly, it is plotted in the range $-\pi \le \omega \le \pi$.

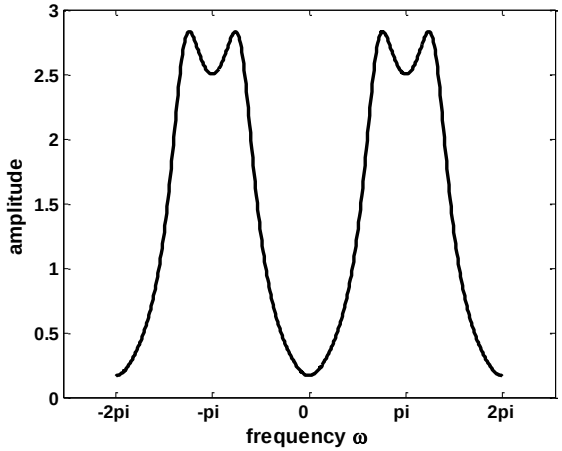

Figure 1: A typical DTFT showing mirror symmetry around π and periodicity every 2π. Usually, only values between 0 and π are plotted. This signal has very little low frequency components and large high frequency components.

- For a real sequence x[n], $X_{re}(e^{j\omega})$ and $|X(e^{j\omega})|$ are even functions of ω, and $X_{im}(e^{j\omega})$ and $\angle X(e^{j\omega})$ are odd functions of ω.
- The inverse DTFT that undoes equation 1 is defined using a complex contour integral (just like for the CT Fourier Transform) and like the CT Fourier Inverse Transform integral definition is never used. Inverses are found by analytically (i.e. in academia) by partial fraction decomposition and recognizing common transform pairs from a table. In industry, you don't need to find the inverse DTFT because you start off with a signal from a sensor that already is in the time-domain; you take it into the frequency domain to analyze it.
- The Fourier Transform $X(e^{j\omega})$ of a sequence x[n] exists if it is absolutely summable, i.e. $\sum_{n=-\infty}^{\infty} |x[n]| < \infty$

  . This is only important in academia; all real-world signals have Fourier Tranforms (or else they'd have infinite energy!).

- Time-shifting property: If the DTFT of x[n] is $X(e^{j\omega})$ then the DTFT of the time-shifted sequence $x[n-n_0]$ is $e^{-j\omega n_0}X(e^{j\omega})$. e.g. x[n-3] has a DTFT of $e^{-3j\omega}X(e^{j\omega})$.
- Frequency-shift property: If the DTFT of x[n] is $X(e^{j\omega})$ then the DTFT of $e^{-j\omega_0 n}x[n]$ is $X(e^{j(\omega-\omega_0)})$
- Convolution property: If the DTFT of g[n] and h[n] are $G(e^{j\omega})$ and $H(e^{j\omega})$ then the DTFT of the sequence g[n]*h[n] is $G(e^{j\omega})\cdot H(e^{j\omega})$.
- Modulation property: Using the above definitions for g[n] and h[n], the DTFT of the sequence $g[n]\cdot h[n]$ is $G(e^{j\omega})*H(e^{j\omega}) = \frac{1}{2\pi}\int_{-\pi}^{\pi} G\left(e^{j\theta}\right)H\left(e^{j(\omega-\theta)}d\theta\right)$. We won't use this much in this course.
- Time-reversal property: If x[n] has a DTFT of $X(e^{j\omega})$ then the DTFT of the time reversed sequence x[-n] is $X(e^{-j\omega})$.

C. Matlab and the DTFT

The DTFT $X(e^{j\omega})$ of a sequence x[n] is a continuous function of ω. Since Matlab can only work on vectors, Matlab can only evaluate $X(e^{j\omega})$ at a finite number of frequencies given to it as a vector. There are two fundamentally different ways we can plot a DTFT using Matlab; one assumes we have already analytically determined $X(e^{j\omega})$ (it will always be of the following form, with some finite M and N: $X(e^{j\omega}) = \frac{b_0+b_1e^{-j\omega}+...+b_Me^{-j\omega M}}{a_0+a_1e^{-j\omega}+...+a_Ne^{-j\omega N}}$) and the other assumes we begin with our original time-domain sequence x[n].

DTFT analytically known

The easier way first: if we have already determined the analytic value $X(e^{j\omega})$ with the coefficients $b_0 \ldots b_m$ and $a_0 \ldots a_N$ as defined above, then

```
[X, omega] = freqz(b, a, N)
```

returns X as the DTFT evaluated at N points equally spaced between 0 and π, and those frequencies are stored in variable omega. You could plot the DTFT magnitude, for instance, using `plot(omega, abs(X))`. Since the DTFT is continuous, you will want to use a large value of N to closely sample the DTFT, and for fast computation make N a power of 2 such as 512. If you wish to explicitly evaluate the DTFT at a different set of frequencies (perhaps you want to verify the periodicity of $X(e^{j\omega})$ by evaluating it at `w = linspace(-2*pi, 2*pi, 100)`), then type

```
X = freqz(b, a, w)
```

For instance, Figure 2 shows a plot of $X(e^{j\omega}) = \frac{-2-3e^{-j\omega}}{1+2e^{-j\omega}+3e^{-2j\omega}}$ over $0 \le \omega \le 2\pi$, generated as follows:

```
w = linspace(0,2*pi,512);  % Eval at 512 points between 0 and 2π
X = freqz([-2 -3], [1 2 3], w);
plot(w, abs(X))
```

Recall other useful Matlab commands for plotting the DTFT, such as `subplot` to enable both magnitude and phase plots of $X(e^{j\omega})$ in a single figure, `angle(X)` to plot the angle of X in radians (you have to multiply the result by 180/π to plot in degrees), and `real(X)` and `imag(X)` to plot the real and imaginary components.

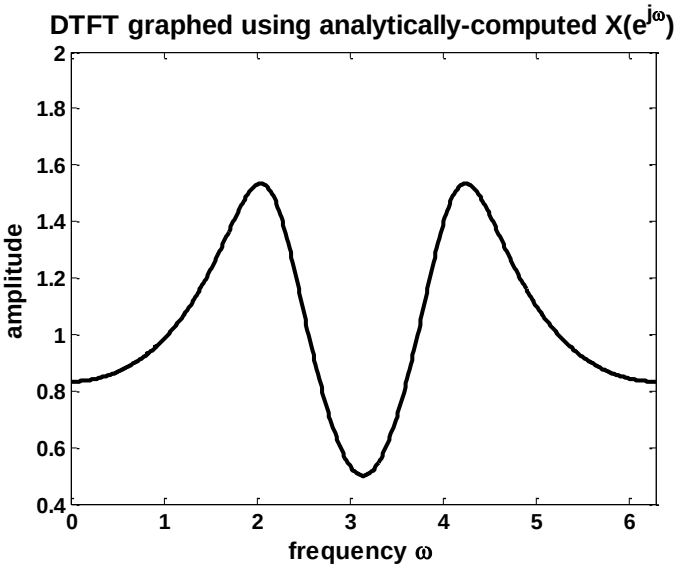

Figure 2: A Matlab-generated plot of a known DTFT. As expected, it is symmetric around π. This example has most of its energy concentrated in the mid-band region, at a frequency of about π/2. Although we have not yet discussed sampling, if this represented a continuous signal sampled at F_s times per second, the digital frequency peak at ω = π/2 would correspond to a continuous frequency peak at f = (π/2) F_s/(2π) Hz.

DTFT not analytically known

If the DTFT is not known, it can be derived directly from x[n] by first selecting the frequencies at which the DTFT value is desired, and then repeatedly evaluating equation 1 for each frequency. Obviously that is inefficient if $X(e^{j\omega})$ is already known, but can still be done for short sequences by calling equation 1 inside a `for` loop. For instance, if x[n] = 1, 1, 2, 1 for n=0, 1, 2, 3, and 0 otherwise, one could evaluate the DTFT at ω = 1.23 rads/s, for instance, using the following code snippet:

```
w = 1.23;
x  = [1 1 2 1]
nx = [0 1 2 3];
X = 0;
for i = 1:length(x)  % make i loop through every value in x
   X = X + x(i)*exp(-j*w*nx(i))
end
```

To have the program evaluate a vector of frequencies w (for instance, 100 evenly-spaced values between 0 and π), you would embed the above code snippet within another for loop to evaluate each different w. The function below does this. It takes the data vector `x` and its index vector `nx`, and computes `w` and `X`, indexed by `k` and `i` respectively. An example of this function's use is shown in Figure 3.

```
function [w,X]=dtftdirect(nx,x)
% DTFTDirect calculates the DTFT of sequence x and its index vector nx
w = linspace(0,pi,100); % evaluate at 100 freqs between 0 and π
X = zeros(size(w));     % set X initially equal to all zeros
for k=1:length(w)       % for every freq in the w vector
   X(k) = 0;
   for i = 1:length(x)  % make i loop through every value in x
         X(k) = X(k) + x(i)*exp(-j*w(k)*nx(i));
   end
end
```

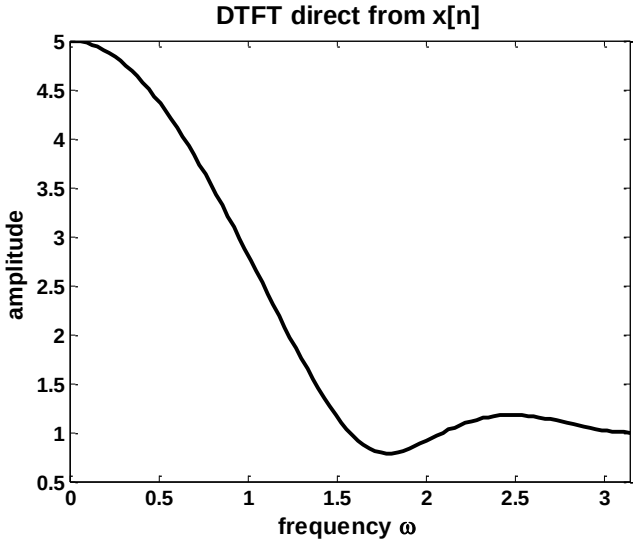

Figure 3: A DTFT computed directly from the sequence x[n] using the above function DTFTdirect, without requiring analytic precomputation of $X(e^{j\omega})$, as would be required by `freqz`. This signal has little high-frequency energy, corresponding to the slowly-changing sequence x[n]. This figure was created by calling

```
[w,X]=dtftdirect([0 1 2],[1 2 3]);
plot(w,abs(X))
```

Problem 1: DTFT analytically known

Consider the signal x[n] whose DTFT is $X(e^{j\omega}) = \dfrac{2 + e^{-j\omega}}{1 - 0.6e^{-j\omega}}$

a) Using Matlab, create a neatly-labeled figure using subplot whose upper plot is the magnitude of $X(e^{j\omega})$ and the lower plot is the phase angle of $X(e^{j\omega})$ for frequencies $-4\pi \le \omega \le 4\pi$. Is the DTFT a periodic function of ω? If so, what is the period? Explain the type of symmetry exhibited by each of the two subplots.

b) **Challenge**: Analytically determine x[n] by computing the inverse DTFT using tables.

c) Now consider a different signal whose DTFT is

$$U(e^{j\omega}) = \frac{0.7 - 0.5e^{-j\omega} + 0.3e^{-j2\omega} + e^{-j3\omega}}{1 + 0.3e^{-j\omega} - 0.5e^{-j2\omega} + 0.7e^{-j3\omega}}$$

and create a figure using subplot whose upper plot is the magnitude of $U(e^{j\omega})$ and whose lower plot is the phase angle of $U(e^{j\omega})$ for frequencies $0 \le \omega \le \pi$. Is this primarily a high frequency or low frequency signal, or something else? Can you explain the jump in the phase angle? The jump can be removed with the Matlab command unwrap. Replot the phase spectrum with the jump removed.

Problem 2: DTFT not analytically known

a) Consider the signal g[n] = [1 3 5 7 9 11 13 15 17] that begins at n=0 (i.e. g[0]=1, etc.). Using the method given in the section "DTFT not analytically known" plot the magnitude spectrum and phase spectrum of the sequence over 0 to π. Comment on the distribution of signal energy. Can you explain the jumps in the phase spectrum?

b) How would you modify your code to plot the phase spectrum in degrees instead of radians? You need only show the new "plot" single-line command.

c) **Challenge**: Analytically determine $G(e^{j\omega})$. Now use freqz to generate a magnitude spectrum. Compare with the magnitude spectrum generated in part a.

Problem 3: Time-shift property of the DTFT

Consider the following program available for download on the course webpage:

```
% Lab4_Prob3
% Time-shifting Properties of the DTFT

clf;
w = linspace(0, pi, 256);
D=4;
num = [1 2 2 1];
H1 = freqz(num, 1, w);
H2 = freqz([zeros(1,D) num], 1, w);
subplot(2,2,1)
plot(w/pi, abs(H1)); grid
title('Magnitude Spectrum of Original Sequence')
subplot(2,2,2)
plot(w/pi, abs(H2)); grid
title('Magnitude Spectrum of Time-Shifted Sequence')
subplot(2,2,3)
plot(w/pi, unwrap(angle(H1))*180/pi); grid
title('Phase Spectrum of Original Sequence')
subplot(2,2,4)
plot(w/pi, unwrap(angle(H2))*180/pi); grid
title('Phase Spectrum of Time-Shifted Sequence')
```

a) Although the programmer received top grades from his professor for clever plotting, he was boned by his TAC for not including comments or axis labels. What does variable D do? What should the horizontal axis and vertical axis labels read (what are their units)?

b) Examine the two systems H1 and H2 by plotting the impulse response of h1[n] and the impulse response of h2[n] (which is just time-delayed h1[n]). Remember, these are discrete systems, so use stem plots.

c) Examine the DTFT of H1 and H2 using the above code. Explain what adding the delay in the time domain did to the magnitude and phase spectrum plots. Qualitative comments are good, but quantitative comments are better.

d) Experiment with a shorter and longer-length signal (i.e. invent a different, shorter signal, and compare how its DTFT changes when not delayed vs. when delayed, and then do the same for a different, longer signal you invent). Does the length of the signal affect how its phase varies with a delay?

e) **Challenge**: If the phase of the DTFT of a real sequence x[n] is a constant that does not change with frequency (e.g. $\angle X(e^{j\omega})$ = a constant vs. something like $\angle X(e^{j\omega}) = \omega$) , what can you say about the sequence x[n]? What values could the phase be?

Problem 4: Convolution property of the DTFT

Consider the following program available for download on the course webpage:

```
% Lab4_Prob4
% Convolution Properties of the DTFT
%
clf;
w = linspace(-pi, pi, 256);
x1 = [1 3 5 7 9 11 13 15 17];
x2 = [1 -2 3 -2 1];
X1 = freqz(x1, 1, w);
X2 = freqz(x2, 1, w);
XP = X1.*X2;
y = conv(x1,x2);
Y = freqz(y, 1, w);
%
subplot(2,2,1)
plot(w/pi, abs(XP)); grid
title('Product of Magnitude Spectra')
%
subplot(2,2,2)
plot(w/pi, abs(Y)); grid
title('Magnitude Spectrum of Convolved Sequence')
%
subplot(2,2,3)
plot(w/pi, unwrap(angle(XP))); grid
title('Sum of Phase Spectra')
%
subplot(2,2,4)
plot(w/pi, unwrap(angle(Y))); grid
title('Phase Spectrum of Convolved Sequence')
title('Magnitude Spectrum of Original Sequence')
```

a) Run the program and comment on the program results. What does it show happens to the DTFT magnitude and phase when two signals are convolved in the time domain?

b) **Challeng**e: The phase of both signals look quite smooth, except for a couple of small jags. You may have noticed similar small jags in the phase spectra of other signals. These are *artifacts*, meaning they shouldn't be there, and are caused by some inaccuracy in the way Matlab calculates those particular values. What causes them? Why are they difficult to overcome? Hint: take a look at the magnitude spectrum.

4. Real-World Problem: Frequency Distribution in the EKG

A. The EKG

The heart from a mechanical viewpoint

The heart has four chambers that pump in a paired fashion; the upper two chambers are called atria and the lower two are called ventricles. There are four valves in the heart; two regulate flow direction between each of the atria and their paired ventricles, and two regulate blood flow out of the ventricles. Blood drains into the atria from large veins under comparatively little pressure for the majority of the cardiac cycle. Then the atria contract from the top down, forcing blood into the lower ventricles. The ventricles contract the moment the atria are done filling them, closing valves at the atrial/ventricular interface and forcing blood under high pressure into the body's arteries. This occurs in about 200ms, after which the valves at the exit of the ventricles close to prevent reverse flow. The "lub-dub, lub-dub" sound heard in a stethoscope are the sounds of the closing of the atrial/ventricular valves ("lub") followed a split second later by the closing of the ventricular exit valves ("dub").

The heart from an electrical viewpoint

Every muscle *except* the heart is directly enervated by a nerve; to contract the muscle, the nerve fires. A severed nerve causes loss of actuating ability of a particular muscle bundle. The cardiac cycle is unique in that it requires a carefully-orchestrated firing sequence of muscle fibers to be effective; if the ventricles contract even a millisecond before the atria, the pumping action is ruined. The heart has thousands of different muscle fibers each of which must be actuated in the correct sequence, from the top of the heart downwards. To guarantee this, cardiac muscle is specialized in that unlike any other muscle it can directly conduct electrical stimuli. Thus, only a single signal is needed to initiate the entire cardiac cycle. Further, there is a region of cardiac muscle at the top of the right atrium called the sinoatrial node that is the body's equivalent of a 555 timer. About once a second it spontaneously contracts. The frequency of contraction can be modified by a variety of feedback mechanisms sensitive to blood oxygen content, breathing rate, and the presence of adrenaline. The contraction causes the muscle to depolarize; the depolarization spreads out in a circular fashion, conducting down the heart and causing successively lower regions to contract until the wave meets itself at the heart's bottom. It takes several tens of milliseconds for cardiac tissue to repolarize, and so when the depolarization wave meets itself at the bottom of the heart it cannot reflect back up the already-depolarized cardiac tissue. About a second later the sinoatrial node fires again, and the cycle repeats.

One obvious problem with this system is that should the wave not meet exactly at the bottom of the heart, the cardiac tissue may have a chance to repolarize and thus the depolarization wavefront may chase itself forever around the heart's walls. This is called "ventricular fibrillation" (Latin for "wormlike movement of the ventricles") and is typically called a heart attack in a young person. The only cure is a complete depolarization of the entire heart at once, which is accomplished by applying a ~300V pulse to the chest wall through defibrillator paddles. (The rubbing of the paddles you see in the movies is the spreading of conductive gel on each paddle to get better conduction through the skin.)

The depolarization wave through the heart causes about a 1mV signal difference to appear on the surface of the chest at opposite ends of the heart. It could be measured between fingertips on opposite hands (as exercise monitors frequently do), between a finger and a toe (as early cardiac detection machines did), or directly on the chest just below the nipples (as modern EKG machines do). The word "EKG" is an acronym for the electro-cardiogram – the "K" occurs because the Germans were the first to build one, and their word for "cardio" is "kardio".

In this exercise we will examine the frequency distribution of a cadet's EKG. Frequency distribution is a good indicator of cardiac health. If a person has a perfectly regular 60bpm (beats per minute) heart rate then this will show up as a spike at exactly 1Hz (it is a periodic signal whose fundamental frequency is 1Hz). Of course, since even this ideally healthy person's EKG form

does not look like a sine wave, there will be harmonics at exactly 2Hz, 3Hz, and so on. A real person's heart rate will not be perfectly regulated, and the changes in rate will show up on the magnitude spectrum as a blurring of the fundamental and harmonic peaks. The more blurring, the more heart rate change, which could be indicative of a variety of diseases. Similarly, a healthy individual's EKG should contain the bulk of its energy in the frequency range below 20Hz. Someone experiencing ventricular fibrillation may have an EKG with significant high-frequency energy; half or more of its energy may be greater than 20Hz.

The DTFT of a sampled signal

Although we will not rigorously examine digitally-sampled signals until the end of this course, you can intuitively appreciate the concepts already. The highest frequency a digital signal can represent corresponds to $\omega = \pi$ rads/sample, or in degrees 180° per sample. An example of this is the sequence [1 -1 1 -1]; every sample shows a 180° phase inversion. Try to capture a faster-changing signal, say with $\omega = 2\pi$ rads/sample, and it changes 360° every sample to [1 1 1 1]…and looks like a 0 frequency signal. Similarly, the slowest that you can sample the signal $\cos(2\pi f_0 t)$ is at $F_s=f_0/2$. This would give you the [1 -1 1 -1] signal. If you sampled it any more slowly, for instance at $F_s=f_0$, you would end up getting [1 1 1 1] and not see the f_0 frequency component at all. So, the highest discrete frequency of $\omega = \pi$ rads/sample in the sampled signal corresponds to a frequency component of $F_s/2$ in the original continuous signal. A discrete frequency of $\omega = 0$ corresponds to the 0 frequency in the original continuous signal.

The code below (and on the course webpage) is a simple extension of the code presented in the earlier section *DTFT Not Analytically Known* and determines the DTFT of a sampled signal. Notice that the only change is that when graphed, rather than graph between 0 and π rads/sample it is graphed between 0 and $F_s/2$ Hz. This is a quite useful although slow; it works inputs that are both native digital sequences and with sequences sampled from continuous-time processes. You may find use for this in future years, although wait until the following lab when you develop a way to do this much faster using the DFT.

```
function [w,X]=DTFTsample(x,Fs)
% DTFTsample takes the DTFT of a sampled signal
% [f,X]=Lab4_dtft(x,Fs) returns the DTFT of sampled signal x
%    sampled at Fs Hz, and the frequencies f of each DTFT sample
% If no output arguments are requested, the function plots the
%    result.

% create the discrete frequency vector that goes from 0 to pi
L=length(x);
omega = linspace(0,pi,L);

% create a DTFT result vector
X1 = zeros(1,L);
for k=1:L        % for every freq in the w vector
   for i = 1:L  % make i loop through every value in x
         X1(k) = X1(k) + x(i)*exp(-j*omega(k)*(i-1));
   end
end

% rescale frequency vector from 0-pi to 0-Fs/2
omega = omega*Fs/(2*pi);

% if no output requested, plot it
if nargout == 0
    plot(omega, abs(X1))
    xlabel('frequency')
    ylabel('amplitude')
    title('magnitude of the DTFT of x[n]')
    figure(gcf)
else
    w = omega;
    X = X1;
end
```

Problem 5: Real-World Applications of the DTFT

a. Examine the code given above for `DTFTsample`. Why does it only compute the DTFT of a sampled signal at discrete-time frequencies corresponding to continuous-time frequencies of $F_{sample}/2$? That is, why does it not compute the DTFT at higher discrete-time frequencies that correspond to continuous-time frequencies of F_{sample} or higher?

b. A 5 minute sample of a volunteer's EKG was taken and posted on the network as ekgdata.mat. The EKG was sampled at 360Hz. The algorithm given above is accurate, but too slow to analyze the entire 5min * 60sec/min * 360samples/sec = 108,000 long data vector. Rather than analyze all the data, just analyze the first 2000 samples (recall you can extract a subvector in Matlab from a large vector by indexing it, like this: x([1:2000])). It may still take a while to compute. Include a plot of the first 2000 samples of the EKG in the time-domain and a plot of the DTFT spectrum magnitude (i.e. the energy distribution vs. frequency) of the first 2000 samples of the EKG.

c. Most of the EKG energy should be in the under-20Hz region. Use the zoom-in and zoom-out tools of the Matlab plot window to examine the area from 0 to 20Hz in greater detail. To do this, select the zoom-in icon (the magnifying glass with the plus) and drag it in the white plot area to select. Select the same-sized vertical region (otherwise the signal will not be visible) and the 0 to 20Hz horizontal area. If you select the wrong area you can select the zoom-out tool (the magnifying glass with the minus) and repeatedly click the plot until the original view is restored. Alternatively, you can zoom-in precisely by selecting the menu options Tools $\rightarrow$ Edit Plot, and then double-click the horizontal axis. Do this until you can precisely locate the first frequency peak indicating the fundamental heartbeat; this will probably be somewhere in the 0.8 to 1.5 Hz region. What is it for the given data? What is the heart rate in beats per minute?

d. You identified the fundamental frequency in part c. Since the EKG of a cardiac cycle does not look exactly like a sine wave, you know from your EE230 studies of the Fourier Series that a first harmonic will be present at a frequency twice the fundamental, a second harmonic at frequency 3 times the fundamental, and so on. How many of these harmonics can you identify in the band below 10 Hz?

e. Zoom back out by selecting the zoom-out key and clicking in the white area a number of times (or faster, by using the menu option Tools $\rightarrow$ Reset View). The EKG is a small signal of about 1mV intensity, and can be affected by capacitively-coupled (through the air) 60Hz powerline noise from wires running overhead and in the floor. Fluorescent lights often add a 120Hz harmonic. Using the zoom-in or direct plot editing as described in part c, identify whether each of these noise sources are present, and if so what their peak power is compared to the peak power of the fundamental heartbeat. That is, find the peak in the magnitude DTFT corresponding to the 60Hz noise (if present), find the peak corresponding to fundamental of the heart beat, and divide the two to find the strength of the 60Hz noise relative to the signal (the inverse of the signal to noise ratio). It will probably be somewhere in the 1-25% area for the 60Hz noise.

f. **Challenge**: Heart rate is directly influenced by breathing rate; it speeds up when you breathe in and slows on the exhale. Using the zoom-in and out capabilities of Matlab, use the EKG to identify the breathing rate of our test subject in breaths-per-minute. The sampling rate is a little coarse here for a precise response; what could be done to get a more accurate reading?

5. Matlab Commands Used in Laboratory 4

Extracting parts of complex signals

abs(X)	returns the magnitude of complex signal X
angle(X)	returns the phase angle of complex signal X
unwrap(angle(X))	returns the phase angle of X, "unwrapped" so that as a complex number travels smoothly counterclockwise around the complex plane, rather than having the phase suddenly jump from $1\angle 179.9°$ to $1\angle -179.9°$ it is unwrapped to go from 179.9° to 180.1°.
real(X)	returns the real part of complex signal X
imag(X)	returns the imaginary part of complex signal X

Signal processing commands

freqz(num,den,w)	evaluates the DTFT given by num and den at the frequencies in w. For example, to evaluate $X(e^{j\omega}) = 1/(3-2e^{-j\omega})$ at 256 points from $0 \le \omega \le \pi$, enter `X=freqz(1, [3 -2], linspace(0,pi,256));`

General Matlab commands

y=x([1:100])	extracts the sequence from index 1 to 100 from x and stores it in y