Lab 7 Report

Applications of the FFT

Alex Hirzel

Submitted to Yang Liu for EE4252

Due November 8, 2012

Michigan Technological University

Project 1: IDFT from DFT

x	$\texttt{fft}(0.1x)$	$\texttt{fft}(0.1x^\star)$
0	$18 - 4.5j$	$18 + 4.5j$
$4 - 1j$	$-3.5388 - 5.6554j$	$-0.46116 - 6.6554j$
$8 - 2j$	$-2.6882 - 2.2528j$	$-1.3118 - 3.2528j$
$12 - 3j$	$-2.3633 - 0.95309j$	$-1.6367 - 1.9531j$
$16 - 4j$	$-2.1625 - 0.14984j$	$-1.8375 - 1.1498j$
$20 - 5j$	$-2 + 0.5j$	$-2 - 0.5j$
$24 - 6j$	$-1.8375 + 1.1498j$	$-2.1625 + 0.14984j$
$28 - 7j$	$-1.6367 + 1.9531j$	$-2.3633 + 0.95309j$
$32 - 8j$	$-1.3118 + 3.2528j$	$-2.6882 + 2.2528j$
$36 - 9j$	$-0.46116 + 6.6554j$	$-3.5388 + 5.6554j$

Table 1: Values of y_1 and y_2 for $x = n(4 + j), n \in [0, 9]$.

The rule for finding the `ifft` using only the `fft` operation is

$$\texttt{myifft}(x) = \frac{[\texttt{fft}(x^\star)]^\star}{N}$$

where $x^\star$ is the complex conjugation operation. This relation is applied to a random set of data below with the output of `ifft` for comparison:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0.93	0.5	0.53	0.37	0.98	0.76	0.6	0.38	0.37	0.83	0.66	0.95	0.87	0.98	0.39
0.93	0.5	0.53	0.37	0.98	0.76	0.6	0.38	0.37	0.83	0.66	0.95	0.87	0.98	0.39

The built-in MATLAB `ifft` function agrees with `myifft` to very high precision.

Project 2: Filtering a Noisy ECG Signal

The DFT seems to be very effective in filtering out the 60Hz noise, with reconstruction with $N = 600$ being the most effective.

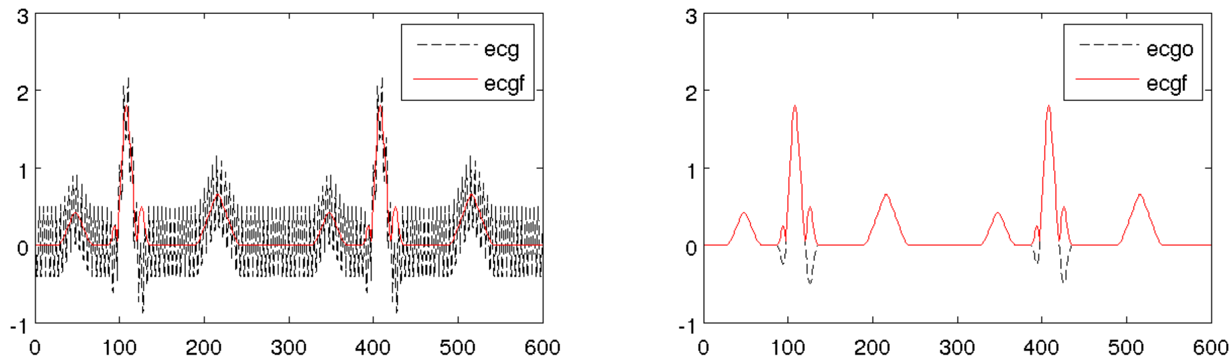


Figure 1: DFT of an ECG signal with $N = 600$ and the 60Hz component removed.

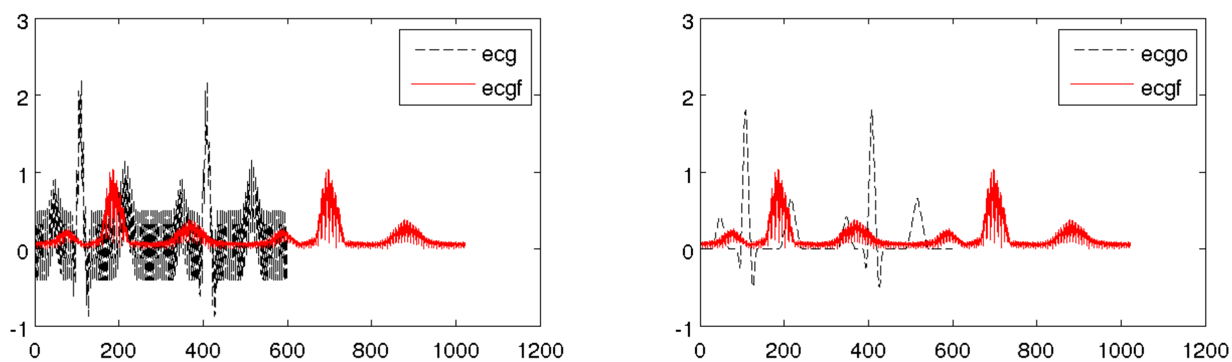


Figure 2: DFT of an ECG signal with $N = 1024$ and the 60Hz component removed.

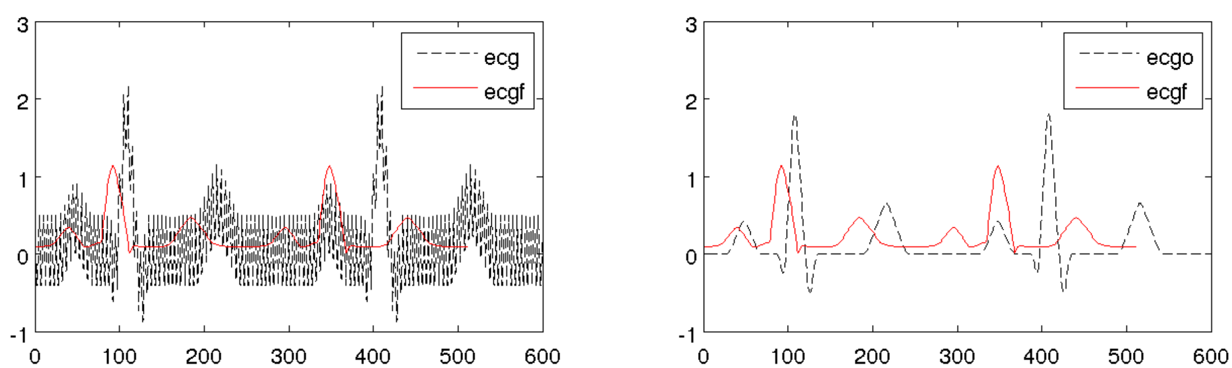
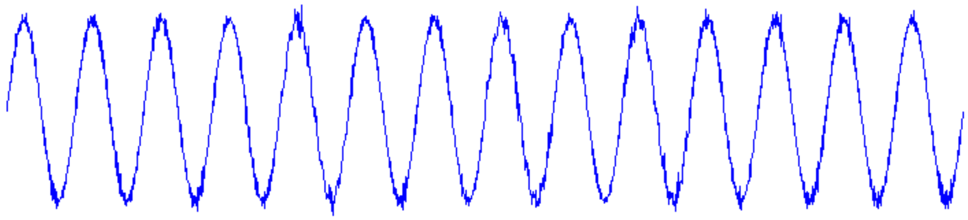


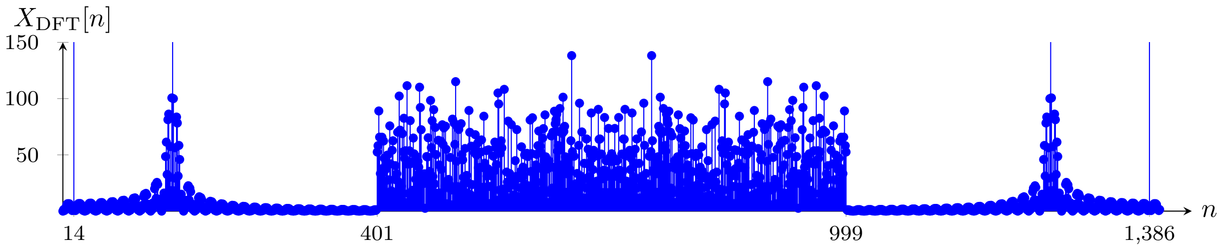
Figure 3: DFT of an ECG signal with $N = 512$ and the 60Hz component removed.

Project 3: Decoding a Mystery Message

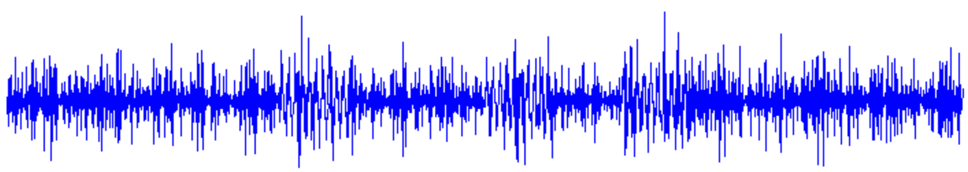
(a) The "mystery" message is displayed below. At face value, it seems to be a high-frequency signal riding on a low-frequency sinusoid.



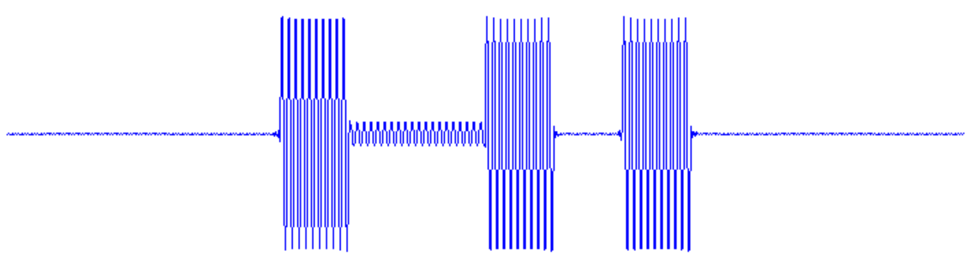
(b) Upon closer inspection of the FFT of the signal, however, two anomalous frequency components are noted at the indices $n = 14$ and $n = 1386$.



These frequency components are four orders of magnitude stronger than the others. **(c)** These low-frequency components correspond to an analog frequency of $\pm 14/(1400 t_s)$. Eliminating these yields the following signal:



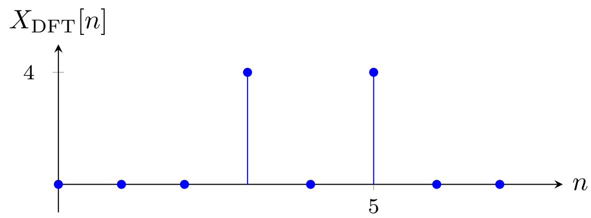
Upon cursory inspection, this signal doesn't seem to contain any useful information due to the presence of high-frequency noise between $n = 401$ and $n = 999$. **(d) (e)** By zeroing out these frequency components and reconstructing the signal, we arrive at



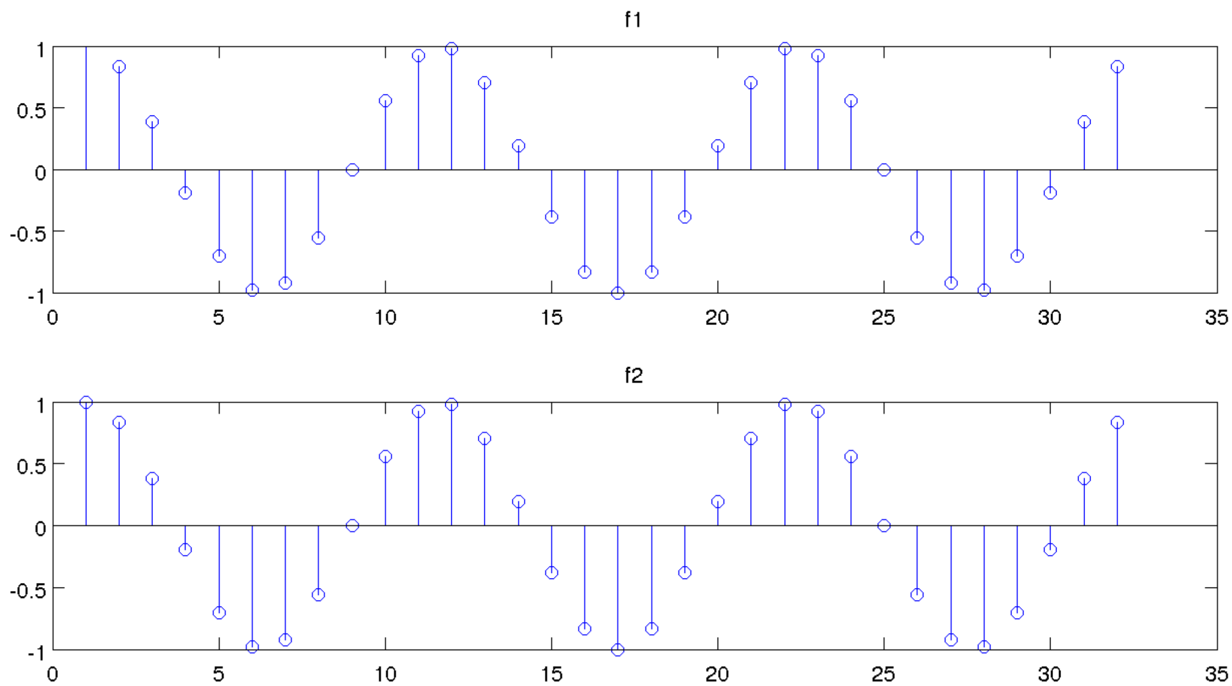
which is a much more useful-seeming signal.

Project 4: Band-Limited Interpolation

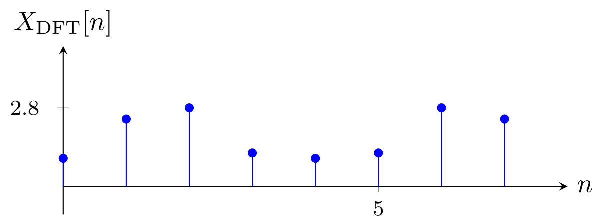
(a) Yes, the sampling rate $S = 200\text{Hz}$ is high enough to prevent aliasing because $x(t)$ has a largest frequency $f_{\max}$ of only 75Hz. Leakage occurs, however, because the sampling range does not even include a full period of the underlying signal. The DFT does have a pair of nonzero samples at the correct frequency (as shown below), but this is essentially a matter of chance.



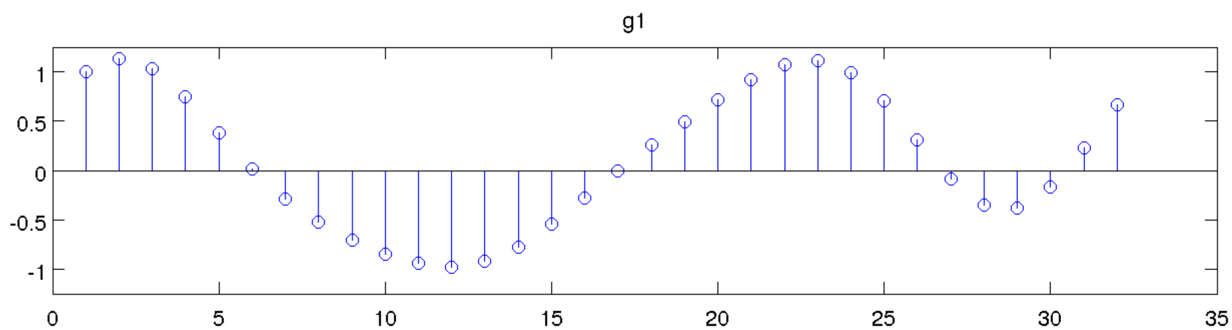
This matches the analog signal because $f_0 = 200 \cdot 3/8 = 75$, but again this is a matter of chance because of the leakage. The following plot shows f_1 and f_2. They are approximately equal (up to machine roundoff error).

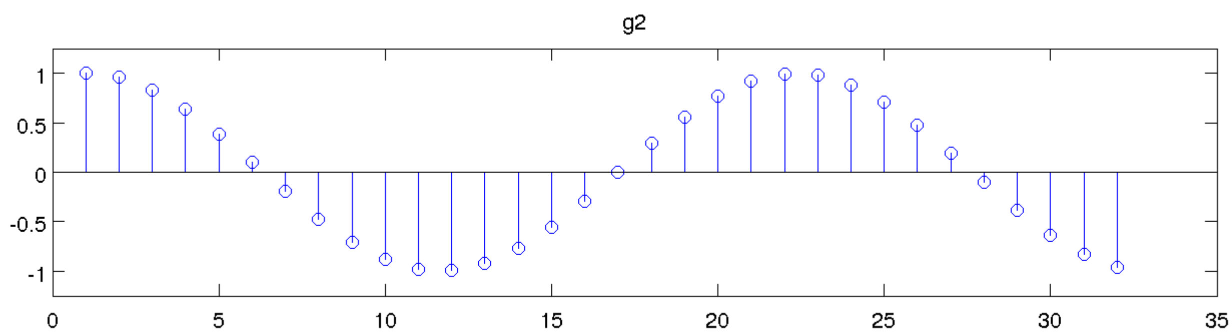


(b) Yes, the sampling rate $S = 400\text{Hz}$ is high enough to prevent aliasing because $x(t)$ has a largest frequency $f_{\max}$ of only 75Hz. Leakage occurs, however, because the sampling range does not even include a full period of the underlying signal. The DFT does not consist of a simple pair of nonzero samples; this is not unexpected.



Reconstructing this signal will result in a complicated sinusoid that does not resemble the original signal. The cause of this is the fact that the samples taken of the signal do not represent a full period of the underlying signal. Also, the samples chosen are not symmetric, so they have a DC value which is a zero-frequency component in the frequency domain. Qualitatively, this messes *everything* up.





(c) Leakage.

Appendix: MATLAB Source Code

What follows is a listing of the MATLAB source code (Listing 1 and Listing 2) used to generate the figures and other information presented in this report.

Listing 1: The MATLAB script used for this report, `Lab07_ahirzel.m`.

```
addpath ../../ClassWorkspace;


% Project 1: IDFT from DFT %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

x = @(n) (4 + 1j)*n;
myifft = @(x) conj(fft(conj(x)))./max(size(x));

xs = x(0:9);  y1 = fft(0.1*xs);  y2 = fft(0.1*conj(xs));
csvwrite('generated/p1-data.txt', [xs', y1', y2']);

rand('seed', 324324); Y = rand(1, 15);
csvwrite('generated/p1-inverses.txt', [abs(myifft(fft(Y))); abs(ifft(fft(Y)))]);


% Project 2: Filtering a Noisy ECG Signal %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load ecg; load ecgo;

ecgf_fft = fft(ecg);
ecgf_fft(60/300       * 600 + 1) = 0;
ecgf_fft((1 - 60/300) * 600 + 1) = 0;
plotecg(ecgf_fft, 'p2-600', ecg, ecgo);

ecgf_fft = [ecgf_fft zeros(1, 1024-600)];
plotecg(ecgf_fft, 'p2-1024', ecg, ecgo);

plotecg(ecgf_fft(1:512), 'p2-512', ecg, ecgo);


% Project 3: Decoding a Mystery Message %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load mystery1;
plot(mystery1); axis off; mysaveas('p3-mystery', 6, 1.5);
dm = fft(mystery1); csvwrite('generated/p3-mystery-dft.txt', abs(dm)');

dm(15) = 0; dm(1400-13) = 0;
plot(ifft(dm)); axis off; mysaveas('p3-mystery-corrected', 6, 1.5);

dm(402:1000) = 0;
csvwrite('generated/p3-mystery-dft-corrected2.txt', abs(dm)');
plot(ifft(dm)); axis off; mysaveas('p3-mystery-corrected2', 6, 1.5);


% Project 4: Band-Limited Interpolation %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

x = @(t) cos(2*pi*75*t);

S = 200; F = fft(x((0:7) / S));
csvwrite('generated/p4a-dft.txt', abs(F)');
subplot(211); stem(real(ifft(4*[abs(F(1:4)) zeros(1, 24) abs(F(5:8))]))); ylim([-1 1]); title('f1');
subplot(212); stem(x((0:31) / (4*S)));                                     ylim([-1 1]); title('f2');
mysaveas('p4a-both', 10, 5);

S = 400; G = fft(x((0:7) / S));
csvwrite('generated/p4b-dft.txt', abs(G)');
subplot(211); stem(real(ifft(4*[G(1:4) zeros(1, 24) G(5:8)]))); ylim([-1.25 1.25]); title('g1');
subplot(212); stem(x((0:31) / (4*S)));                          ylim([-1.25 1.25]); title('g2');
mysaveas('p4b-both', 10, 5);
```

Listing 2: The MATLAB script used to plot the ECG plots shown in this report, `plotecg.m`.

```
function plotecg(t, na, ecg, ecgo)

ecgf = abs(ifft(t));
n = 0:(length(ecgf) - 1);
subplot(1, 2, 1); plot(0:599, ecg,  '--k', n, ecgf, 'r'); ylim([-1,3]); legend('ecg',  'ecgf');
subplot(1, 2, 2); plot(0:599, ecgo, '--k', n, ecgf, 'r'); ylim([-1,3]); legend('ecgo', 'ecgf');
mysaveas(na, 10, 2.5);
```