6.003 Lab 2

Nada Amin
namin@mit.edu

April 6, 2005

Basic Problems

(b)

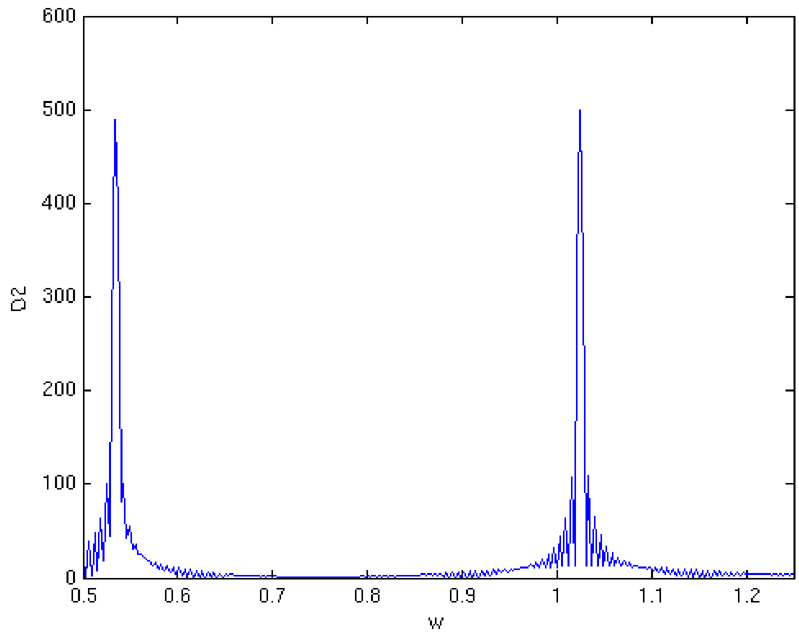

Figure 1: Plot of $D_2(e^{jw})$

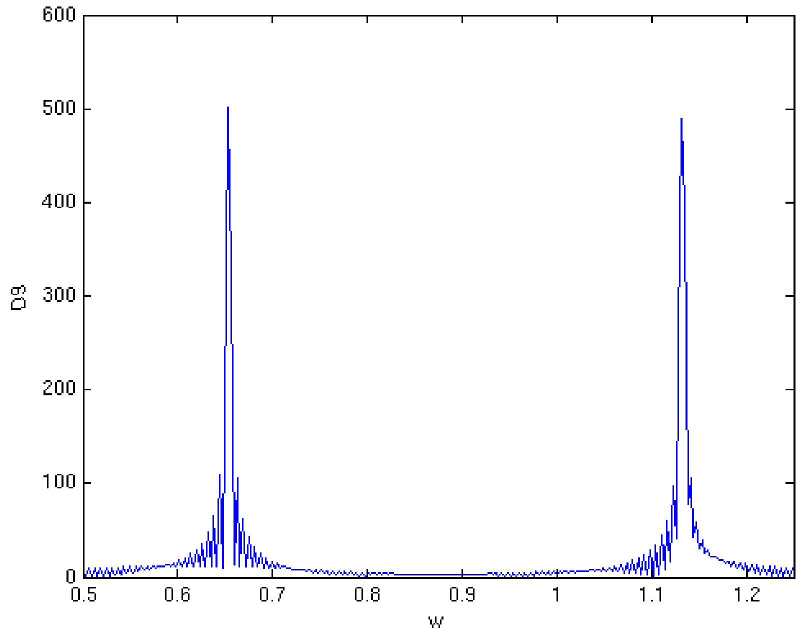

Figure 2: Plot of $D_9(e^{jw})$

Advanced Problems

(f)

Here is a table of indices and their corresponding ω that I found to be the closest to each of the touch-tone frequencies.

touch-tone context	real ω	closest ω	closest index
$\omega_{row}(1)$	0.5346	0.5338	175
$\omega_{row}(2)$	0.5906	0.5921	194
$\omega_{row}(3)$	0.6535	0.6535	214
$\omega_{row}(4)$	0.7217	0.7210	236
$\omega_{column}(1)$	0.9273	0.9265	303
$\omega_{column}(2)$	1.0247	1.0247	335
$\omega_{column}(3)$	1.1328	1.1321	370

(g)

Here is a list of values of $|D_8(e^{jwk})|^2$ for each value of w_k that I determined in part (f).

touch-tone context	$\lvert D_8(e^{jwk})\rvert^2$
$\omega_{row}(1)$	$3.9811 \cdot 10^0$
$\omega_{row}(2)$	$1.6592 \cdot 10^2$
$\omega_{row}(3)$	$2.5049 \cdot 10^5$
$\omega_{row}(4)$	$5.2769 \cdot 10^1$
$\omega_{column}(1)$	$4.8526 \cdot 10^1$
$\omega_{column}(2)$	$2.5022 \cdot 10^5$
$\omega_{column}(3)$	$9.0996 \cdot 10^0$

We observe that two frequencies ($\omega_{row}(3)$ and $\omega_{column}(2)$) have larger energy values by at least 3 order of magnitudes. Not surprisingly, these are the frequencies that are actually present in the signal for digit 8.

(h-i)

I decoded the provided signals into the following phone numbers:

signal	phone number
`x1`	968-7335
`x2`	324-0716
`hardx1`	968-7335
`hardx2`	324-0716

MATLAB Code

`ttdecode.m`

```
function [phone_number] = ttdecode(phone_signal, verbose)
% TTDECODE
% [phone_number] = ttdecode(phone_signal, verbose)
%
% This functions decodes a touch-tone signal into a phone number.
% It assumes the signal is formatted as a varying number of
% samples of touch-tone for each digit separated by a varying
% number of samples of total silence.
%
% As defined in Section 5.2 (93-96) of Buck, Daniel and Singer
% the touch-tone frequencies are
%
%       w row                w column
% index   (rad)          index   (rad)
% 1       0.5346         1       0.9273
% 2       0.5906         2       1.0247
% 3       0.6535         3       1.1328
% 4       0.7217
%
% and the digits are coded as a sum of two sines, one with a row
% frequency and one with a column frequency:
%
%              | column index
%-----------------------------
%  row index |      1 2 3
%-----------------------------
%       1           1 2 3
%       2           4 5 6
%       3           7 8 9
%       4             0
%
% Inputs:
%   phone_signal -- The signal to decode.
%   verbose      -- Whether the function prints debugging information
%                   (optional, false by default).
%
% Outputs:
%   phone_number -- The decoded phone number, as a vector of digits.
%                   -1 is used in place of a digit when the tone
%                   cannot be decoded.
%
```

```
if ~exist('verbose')
  verbose = 0;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Create a list of the frequency indices that correspond to a
% touch-tone frequency.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

w_row = [0.5346 0.5906 0.6535 0.7217];
w_col = [0.9273 1.0247 1.1328];
sig_ws = [w_row w_col];

sig_ks = find_index(sig_ws);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Identify the regions in the samples that represent a tone.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

N = length(phone_signal);

nzi = find(abs(phone_signal)>0);
gaps = find(diff(nzi)>1);
end_indices = nzi(gaps);
start_indices = [1 end_indices];
end_indices = [end_indices N];
n_digits = length(end_indices);

if verbose
  fprintf(1, '%d digits\n', n_digits);
  fprintf(1, 'start indicies: ');
  fprintf(1, '%d ', start_indices);
  fprintf(1, '\n');
  fprintf(1, 'end indicies: ');
  fprintf(1, '%d ', end_indices);
  fprintf(1, '\n');
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Decode each tone into a digit.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
phone_number = zeros(1,n_digits);
for i=1:n_digits,
  d = phone_signal([start_indices(i):end_indices(i)]);
  D = fft(d, 2048);
  Dabs = abs(D);
```

```
  Dsq = Dabs.*Dabs;
  Den = Dsq(sig_ks);
  max2 = sort(Den, 2, 'descend');
  s = find((max2(1)==Den) | (max2(2)==Den));
  v = -1;
  switch s(1),
    case 1,
      switch s(2),
        case 5,
          v = 1;
        case 6,
          v = 2;
        case 7,
          v = 3;
      end
    case 2,
      switch s(2),
        case 5,
          v = 4;
        case 6,
          v = 5;
        case 7,
          v = 6;
      end
    case 3,
      switch s(2),
        case 5,
          v = 7;
        case 6,
          v = 8;
        case 7,
          v = 9;
        end
      case 4,
        switch s(2),
          case 6,
            v = 0;
        end
    end
  if v==-1
    warning(sprintf('could not decode digit at position %d', i));
  end
  phone_number(i) = v;
end
```

`find_index.m`

```
function [k] = find_index(w_0)
% FIND INDEX
%
% [k] = find_index(w_0)
%
% This functions finds an index, in a 2048-samples FT, which is
% closest to the given frequency.
%
% Inputs:
%   w_0 -- The frequency to find
%
% Outputs:
%   k   -- An index, between 1 and 2048, such that 2*pi*(k-1)/2048
%          is one of the closest frequencies to w_0.
%
k = round((2048*w_0)/(2*pi))+1;
```

Script for Basic Problems

```
% (a)
n_from = 0;
n_to = 999;

n = [n_from:n_to];

w_col = [0.9273 1.0247 1.1328];
w_row = [0.5346 0.5906 0.6535 0.7217];

%  1 2 3
%1 1 2 3
%2 4 5 6
%3 7 8 9
%4   0
w0 = [w_row(4) w_col(2)];
w1 = [w_row(1) w_col(1)];
w2 = [w_row(1) w_col(2)];
w3 = [w_row(1) w_col(3)];
w4 = [w_row(2) w_col(1)];
w5 = [w_row(2) w_col(2)];
w6 = [w_row(2) w_col(3)];
w7 = [w_row(3) w_col(1)];
w8 = [w_row(3) w_col(2)];
w9 = [w_row(3) w_col(3)];
```

```
d0 = sin(n*w0(1)) + sin(n*w0(2));
d1 = sin(n*w1(1)) + sin(n*w1(2));
d2 = sin(n*w2(1)) + sin(n*w2(2));
d3 = sin(n*w3(1)) + sin(n*w3(2));
d4 = sin(n*w4(1)) + sin(n*w4(2));
d5 = sin(n*w5(1)) + sin(n*w5(2));
d6 = sin(n*w6(1)) + sin(n*w6(2));
d7 = sin(n*w7(1)) + sin(n*w7(2));
d8 = sin(n*w8(1)) + sin(n*w8(2));
d9 = sin(n*w9(1)) + sin(n*w9(2));

% (b)
D2 = fft(d2, 2048);
D9 = fft(d9, 2048);
k = [0:2047];
omega = 2*pi*k/2048;
figure;
plot(omega, abs(D2));
xlabel('w'); ylabel('D2');
axis([0.5 1.25 0 600]);
% peaks are at 0.55 and 1.05
figure;
plot(omega, abs(D9));
xlabel('w'); ylabel('D9');
axis([0.5 1.25 0 600]);
% peaks are at 0.65 and 1.15

% (c)
space = zeros(1,500);
% my phone is 225-7326
phone = [d2 space d2 space d5 space d7 space d3 space d2 space d6];
```

Script for Advanced Problems

```
% (f)
k_row = find_index(w_row);
k_col = find_index(w_col);
wk_row = omega(k_row);
wk_col = omega(k_col);

% (g)
all_ks = [k_row k_col];
D8 = fft(d8, 2048);
D8abs = abs(D8);
D8sq = D8abs .* D8abs;
D8en = D8sq(all_ks);
```

```
% (h-i)
dphone = ttdecode(phone);
load touch;
dx1 = ttdecode(x1);
dx2 = ttdecode(x2);
dhardx1 = ttdecode(hardx1);
dhardx2 = ttdecode(hardx2);

fprintf(1, 'My phone numberis ');
fprintf(1, '%d%d%d-%d%d%d%d', dphone);
fprintf(1, '\nphone number of x1 is ');
fprintf(1, '%d%d%d-%d%d%d%d', dx1);
fprintf(1, '\nphone number of x2 is ');
fprintf(1, '%d%d%d-%d%d%d%d', dx2);
fprintf(1, '\nphone number of hardx1 is ');
fprintf(1, '%d%d%d-%d%d%d%d', dhardx1);
fprintf(1, '\nphone number of hardx2 is ');
fprintf(1, '%d%d%d-%d%d%d%d', dhardx2);
fprintf('\n');

%My phone numberis 225-7326
%phone number of x1 is 968-7335
%phone number of x2 is 324-0716
%phone number of hardx1 is 968-7335
%phone number of hardx2 is 324-0716
```