ECE 438 Lab 6a

David Dang & Benedict Lee

2.3 Windowing Effects

INLAB REPORT:

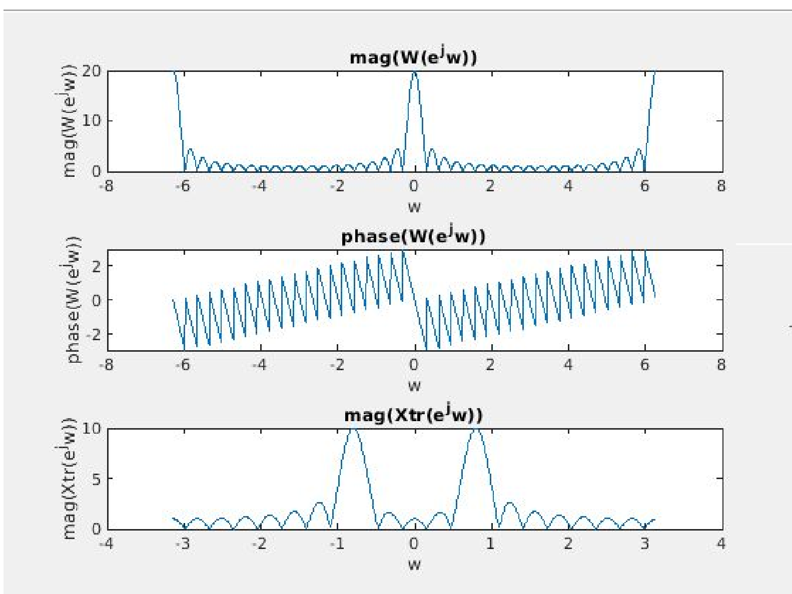
1. Submit the plot of the phase and magnitude of W(ejω).
2. Submit the analytical expression for X(ejω).
3. Submit the magnitude plot of Xtr(ejω).
4. Describe the difference between |Xtr(ejω)| and |X(ejω)|. What is the reason for this difference?
5. Comment on the effects of using a different window for w(n). For example, what would you expect your plots to look like if you had used a Hamming window in place of the truncation (rectangular) window? (See Fig. 1 for a plot of a Hamming window of length 20 and its DTFT.)

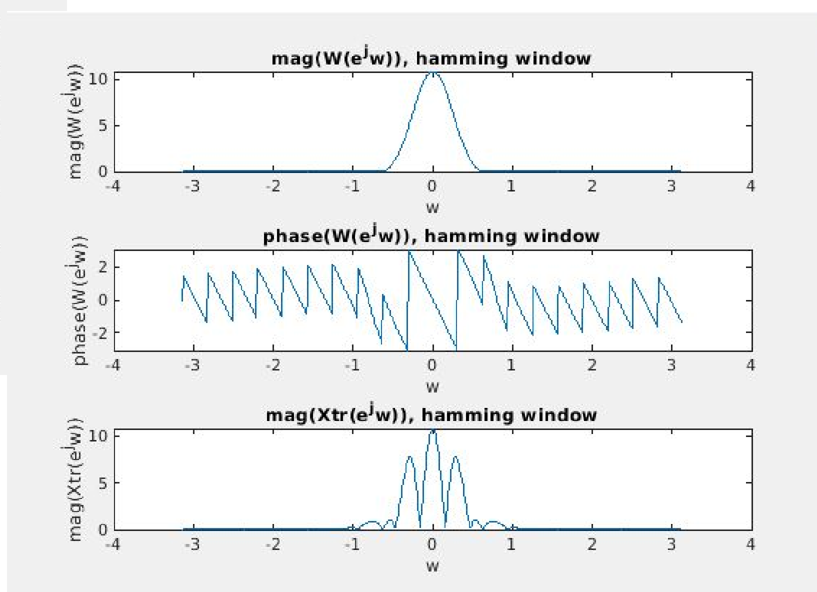
- $x(n) = \cos\left(\frac{\pi}{4}n\right)$

$$X(e^{jw}) = \pi \, rep_{2\pi}\left[\delta\left(w-\frac{\pi}{4}\right)+\delta\left(w+\frac{\pi}{4}\right)\right]$$

$$X(e^{jw}) = \pi \sum_{k=-\infty}^{\infty}\left(\delta\left(w-\frac{\pi}{4}-2\pi k\right)+\delta\left(w+\frac{\pi}{4}-2\pi k\right)\right)$$

- The magnitude plot of X(e^jw) is similar to Xtr(e^jew) except that the peaks grow to infinity since it's not truncated
- If a hamming window was used for w(n), the plots would look like what has been generated below.





3.1 Computing the DFT

INLAB REPORT:

1. Submit a listing of your code for DFTsum.
2. Submit the magnitude plots.
3. Submit the corresponding analytical expressions.

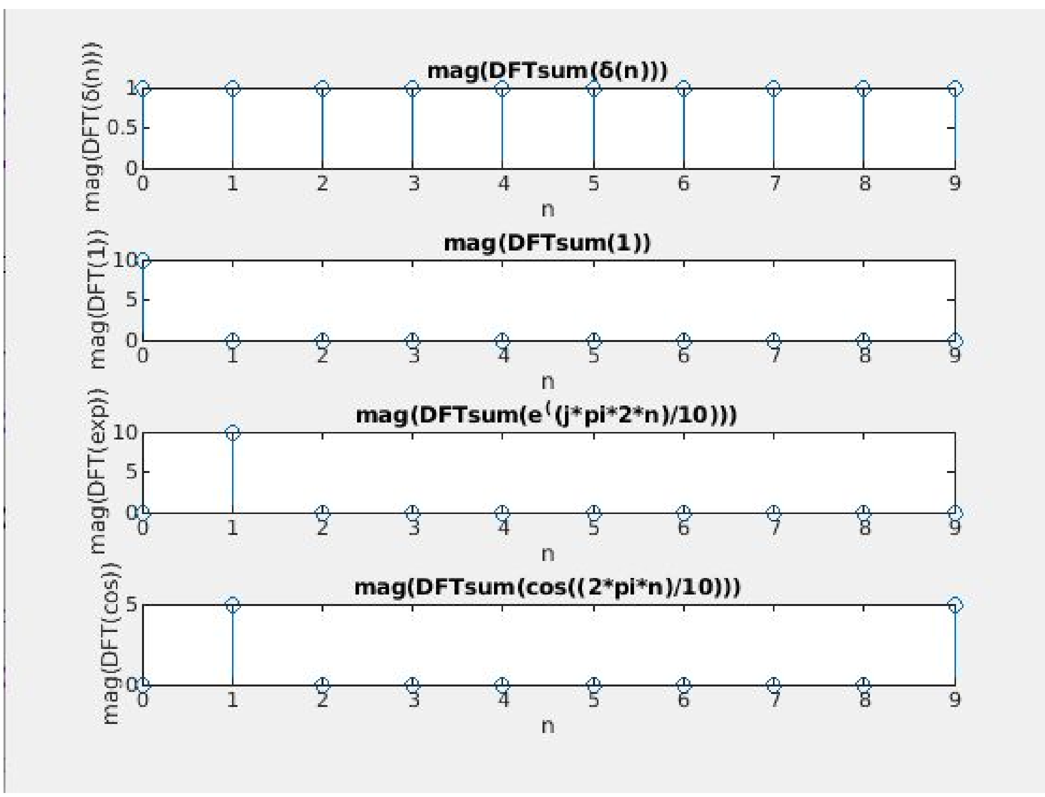
```
function [ X ] = DFTsum( x )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here

j = sqrt(-1);
N = length(x);
X = zeros(1,N);

for k = 1:N
    for n = 1:N

       X(k) = X(k) + x(n)*exp((-j*2*pi*(k-1)*(n-1))/N);
    end
end

end
```



- $x(n) = \delta(n)$, $N=10$

  $X(k) = 1, \quad 0 \le k \le 9$

- $x(n) = 1$, $N=10$

  $X(k) = 10\delta[k], \quad 0 \le k \le 9$

- $x(n) = e^{j2\pi n/10}$, $N=0$

  $X(k) = 10\delta[k-1], \quad 0 \le k \le 9$

- $x(n) = \cos(2\pi n/10)$, $N=10$

  $X(k) = 5\{\delta[k-1] + \delta[k-9]\},$
  $0 \le k \le 9.$

INLAB REPORT:

1. Submit the listing of your code for IDFTsum.
2. Submit the four time-domain IDFT plots.

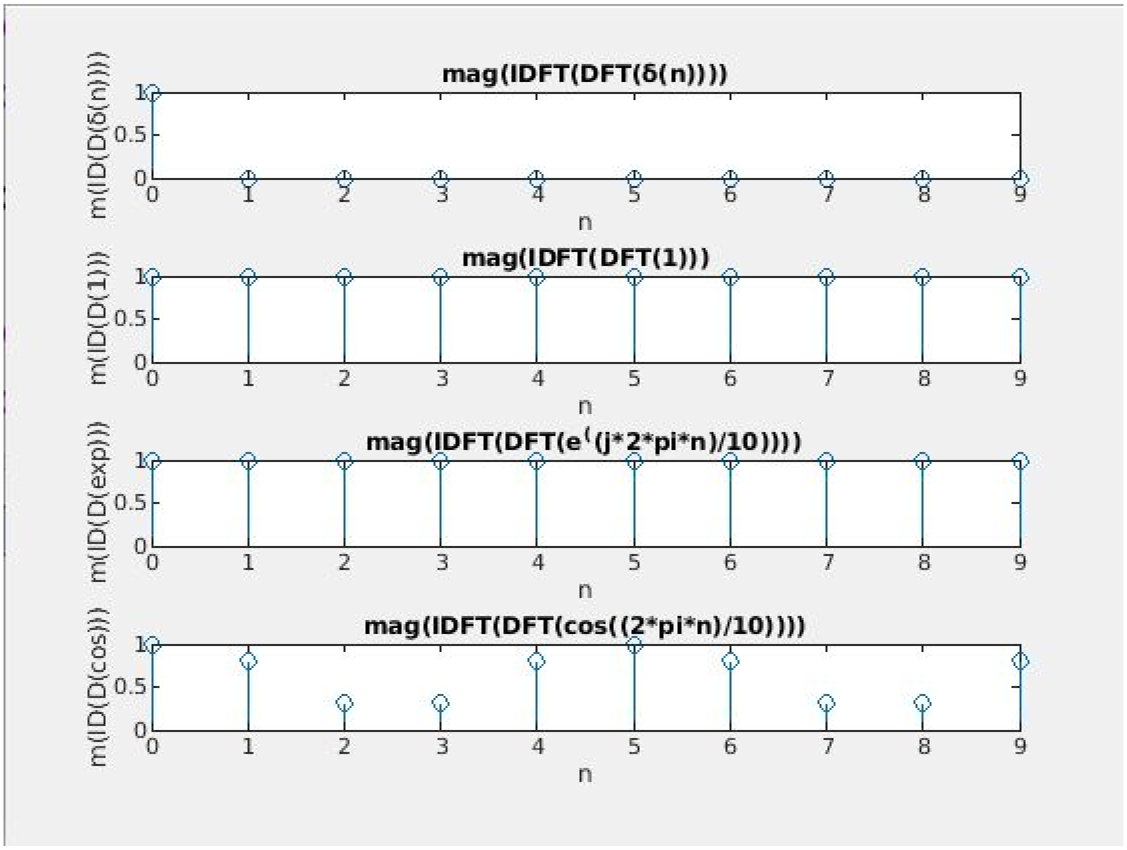
```
function [ x ] = IDFTsum( X )
%UNTITLED5 Summary of this function goes here
%   Detailed explanation goes here

j = sqrt(-1);
N = length(X);
x = zeros(1,N);

for n = 1:N
    for k = 1:N

        x(n) = x(n) + (1/N)*(X(k)*exp((j*2*pi*(k-1)*(n-1))/N));
    end
end

end
```



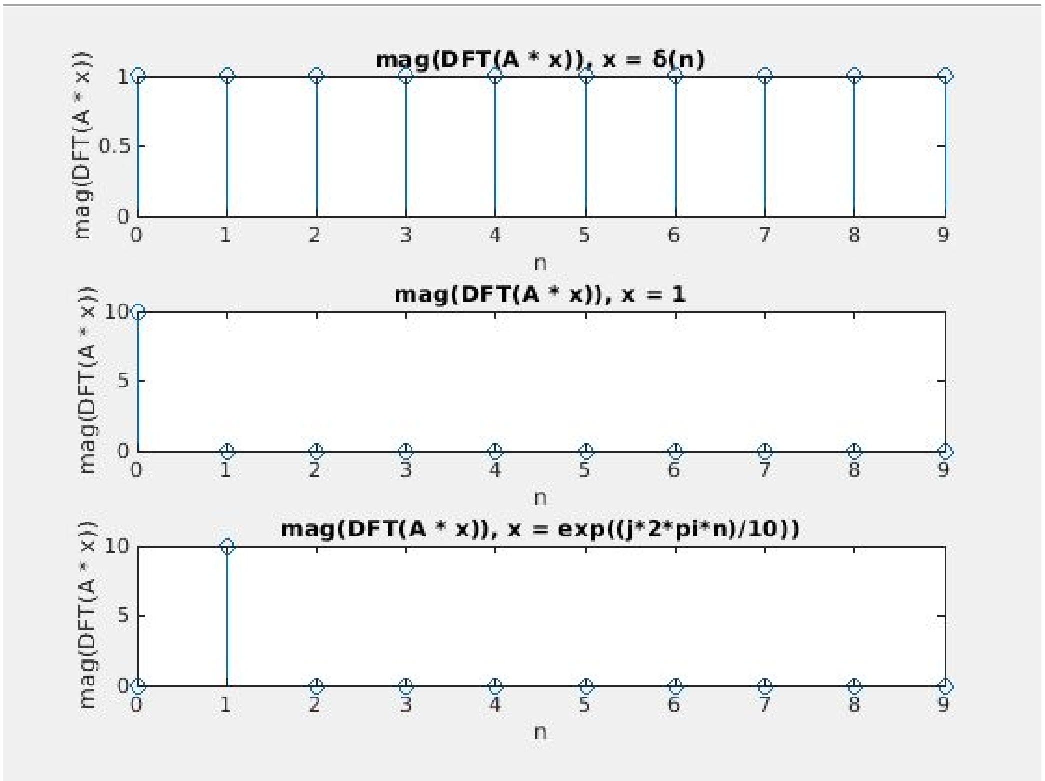
<u>3.2 Matrix Representation of the DFT</u>

INLAB REPORT:

1. Print out the matrix A for N= 5.
2. Hand in the three magnitude plots of the DFT's.
3. How many multiplies are required to compute an N point DFT using the matrix method? (Consider a multiply as the multiplication of either complex or real numbers.)

```
A =

   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i   1.0000 + 0.0000i
   1.0000 + 0.0000i   0.3090 - 0.9511i  -0.8090 - 0.5878i  -0.8090 + 0.5878i   0.3090 + 0.9511i
   1.0000 + 0.0000i  -0.8090 - 0.5878i   0.3090 + 0.9511i   0.3090 - 0.9511i  -0.8090 + 0.5878i
   1.0000 + 0.0000i  -0.8090 + 0.5878i   0.3090 - 0.9511i   0.3090 + 0.9511i  -0.8090 - 0.5878i
   1.0000 + 0.0000i   0.3090 + 0.9511i  -0.8090 + 0.5878i  -0.8090 - 0.5878i   0.3090 - 0.9511i
```



- N^2 multiplies are required to compute an N point DFT using the matrix method

INLAB REPORT:

1. Hand in your analytical expression for the elements of B.
2. Print out the matrix B for N= 5.
3. Print out the elements of C=BA. What form does C have? Why does it have thisform?

- $$B_{nk} = \frac{1}{N} e^{j\frac{2\pi(k-1)(n-1)}{N}}$$

- C is in the form of the identity matrix. Any matrix multiplied by its inverse yields the identity matrix.

```
B =

   0.2000 + 0.0000i   0.2000 + 0.0000i   0.2000 + 0.0000i   0.2000 + 0.0000i   0.2000 + 0.0000i
   0.2000 + 0.0000i   0.0618 + 0.1902i  -0.1618 + 0.1176i  -0.1618 - 0.1176i   0.0618 - 0.1902i
   0.2000 + 0.0000i  -0.1618 + 0.1176i   0.0618 - 0.1902i   0.0618 + 0.1902i  -0.1618 - 0.1176i
   0.2000 + 0.0000i  -0.1618 - 0.1176i   0.0618 + 0.1902i   0.0618 - 0.1902i  -0.1618 + 0.1176i
   0.2000 + 0.0000i   0.0618 - 0.1902i  -0.1618 - 0.1176i  -0.1618 + 0.1176i   0.0618 + 0.1902i


C =

   1.0000 + 0.0000i  -0.0000 + 0.0000i  -0.0000 - 0.0000i   0.0000 - 0.0000i   0.0000 - 0.0000i
  -0.0000 + 0.0000i   1.0000 + 0.0000i  -0.0000 - 0.0000i   0.0000 - 0.0000i   0.0000 - 0.0000i
  -0.0000 + 0.0000i  -0.0000 + 0.0000i   1.0000 + 0.0000i  -0.0000 - 0.0000i  -0.0000 - 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i  -0.0000 + 0.0000i   1.0000 - 0.0000i  -0.0000 - 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i  -0.0000 + 0.0000i  -0.0000 - 0.0000i   1.0000 + 0.0000i
```

3.3 Computation Time Comparison

INLAB REPORT:

Report the cpu time required for each of the two implementations. Which method is faster? Which method requires less storage?

```
X = DFTsum(x)
ans =

    0.3400

X = A * x
ans =

     0
```

- The matrix method for calculating the DFT was faster. Using DFTsum to calculate the DFT required less storage since it didn't require an intermediate variable (i.e. matrix 'A') to calculate the DFT.