FFT Octave Codes (1A)

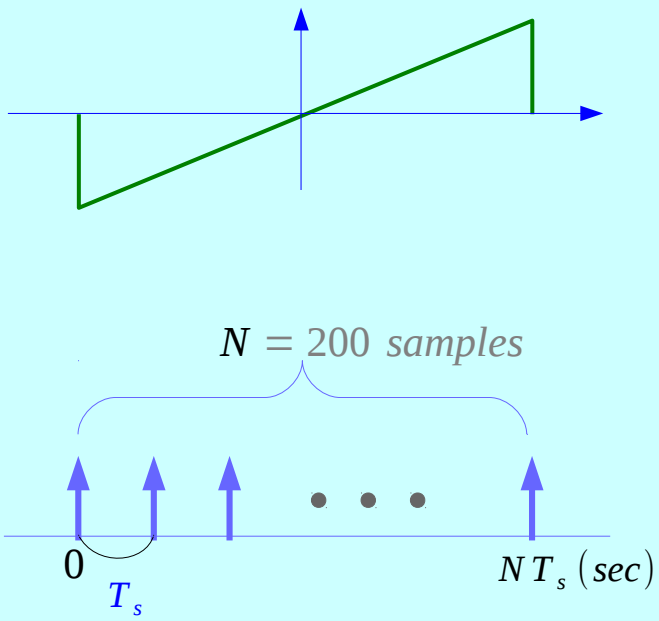
Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

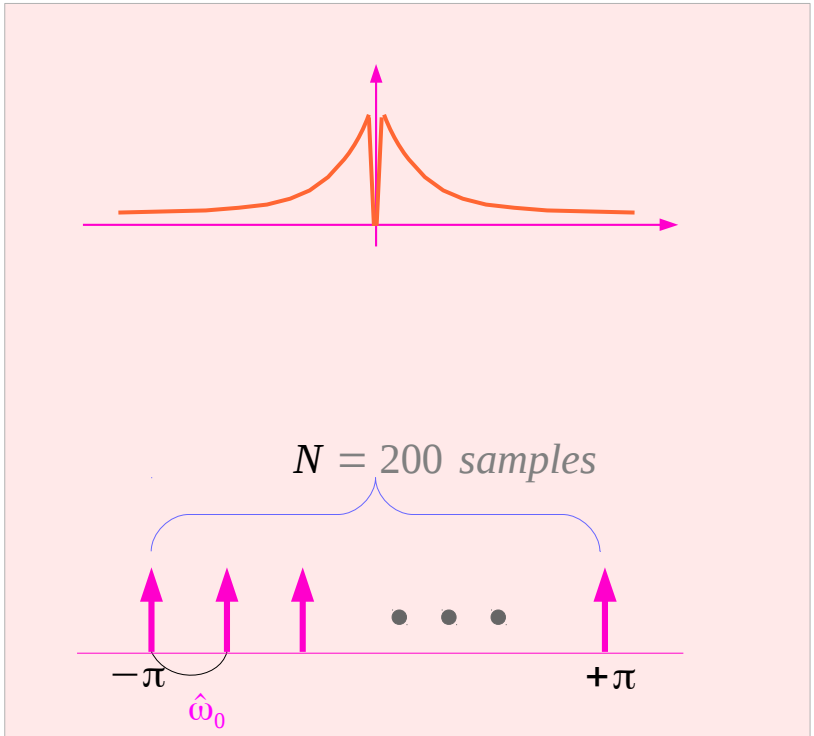
Based on

B Ninness, Spectral Analysis using the FFT

Time Scales and Frequency Scales



$$T_s = \frac{1}{f_s} = 0.1\ (sec)$$



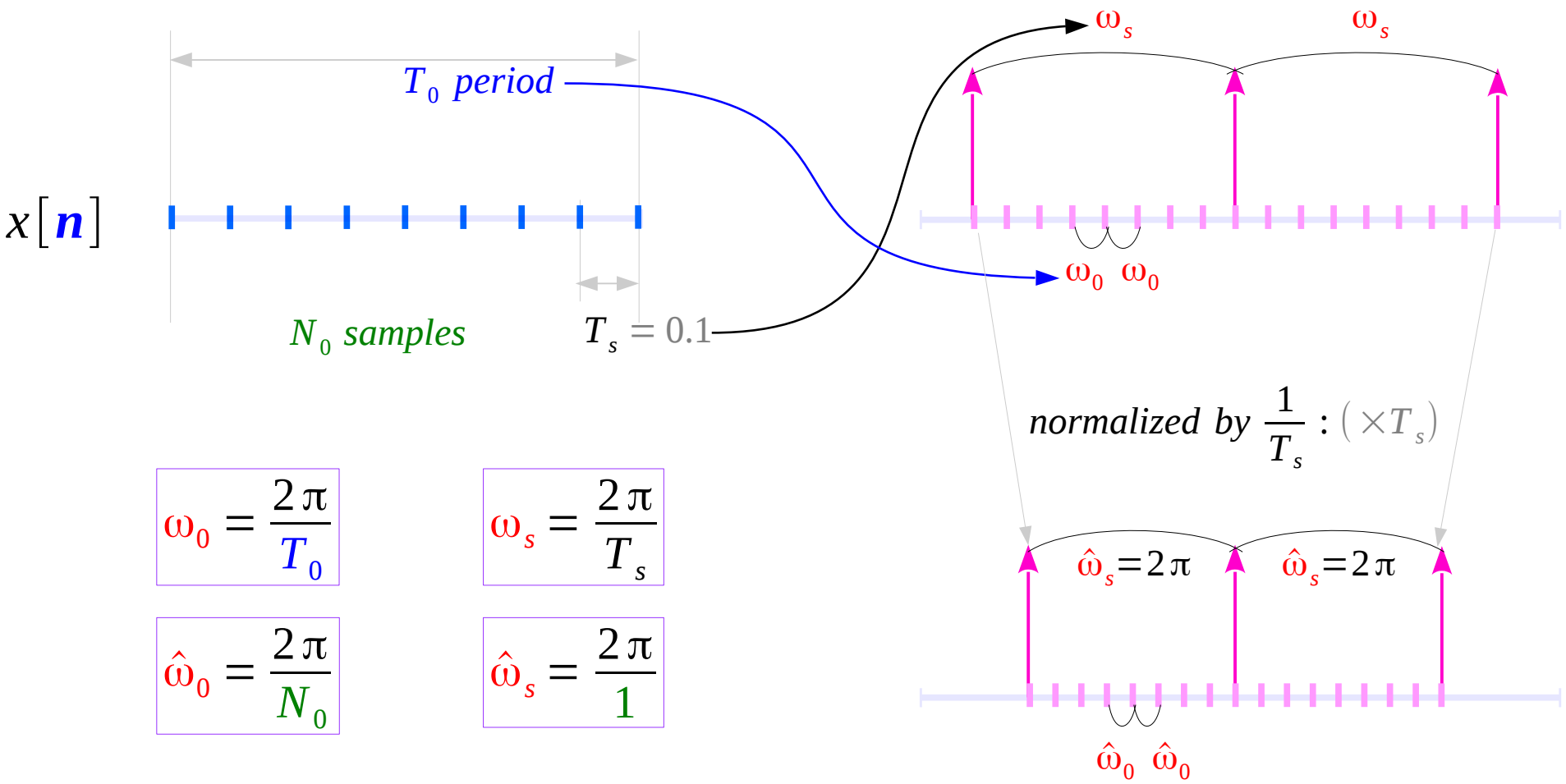
$$\hat{\omega}_0 = \frac{2\pi}{N} = \frac{2\pi}{200} = 0.031416\ (rad)$$

B Ninness, Spectral Analysis using the FFT

Replication Frequency ω_s and Resolution ω_0

$$T_0 = T_s N_0 = 200 \cdot 0.1$$

$$\omega_0 = \frac{2\pi}{200 \cdot 0.1} = 0.1\pi \qquad \omega_s = \frac{2\pi}{0.1} = 20\pi$$



$$\omega_0 = \frac{2\pi}{T_0} \qquad \omega_s = \frac{2\pi}{T_s}$$

$$\hat{\omega}_0 = \frac{2\pi}{N_0} \qquad \hat{\omega}_s = \frac{2\pi}{1}$$

$$\hat{\omega}_0 = \omega_0 T_s = 0.1\pi \cdot 0.1 = 0.01\pi$$

Comparison of variable names

```
function s1( N )
% N = 200;

fs = 10;
t = (0:1:N-1)/fs;
x = cos(pi*t);

ws = 2*pi/N;
wnorm = -pi:w0:+pi;
wnorm = wnorm(1:N);
w = wnorm * fs;
X = fftshift(fft(x));

subplot(2,1,1);
plot(t, x);
subplot(2,1,2);
plot(w, abs(X));
```

Variables used in octave	Variables used in formulas
fs	$f_s = \frac{1}{T_s}$
ws	$\omega_0 = 2\pi f_0 = \frac{2\pi}{T_0}$
wnorm	$k\hat{\omega}_0$
w	$k\omega_0 = 2\pi f_0 = \frac{2\pi}{T_0}$
	$\hat{\omega}_0 = 2\pi\left(\frac{f_0}{f_s}\right) = 2\pi\left(\frac{T_s}{NT_s}\right) = \frac{2\pi}{N}$
2*pi*fs	$\omega_s = 2\pi f_s = \frac{2\pi}{T_s}$

B Ninness, Spectral Analysis using the FFT

FFT of a cosine wave

x ⟺ X

t w

$\cos(\pi t)$

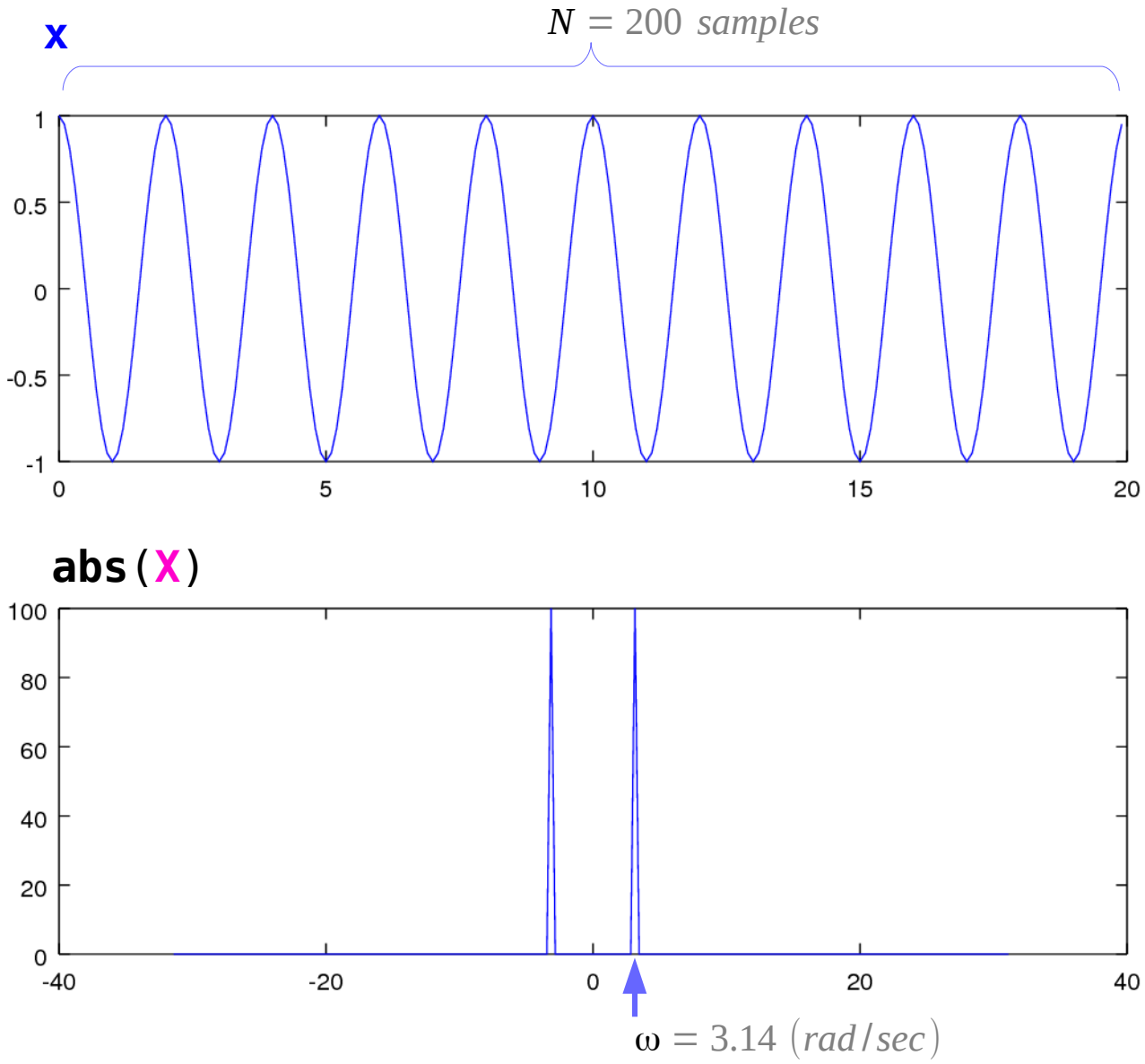
$\cos(\pi n T_s)$

$\cos(0.1\pi n)$

$\hat{\omega} = 0.1\pi$

$\omega = \hat{\omega} \cdot f_s = 0.1\pi \cdot 10 = 3.14$

$\hat{\omega} = \omega \cdot T_s = \pi \cdot 0.1 = 0.314$

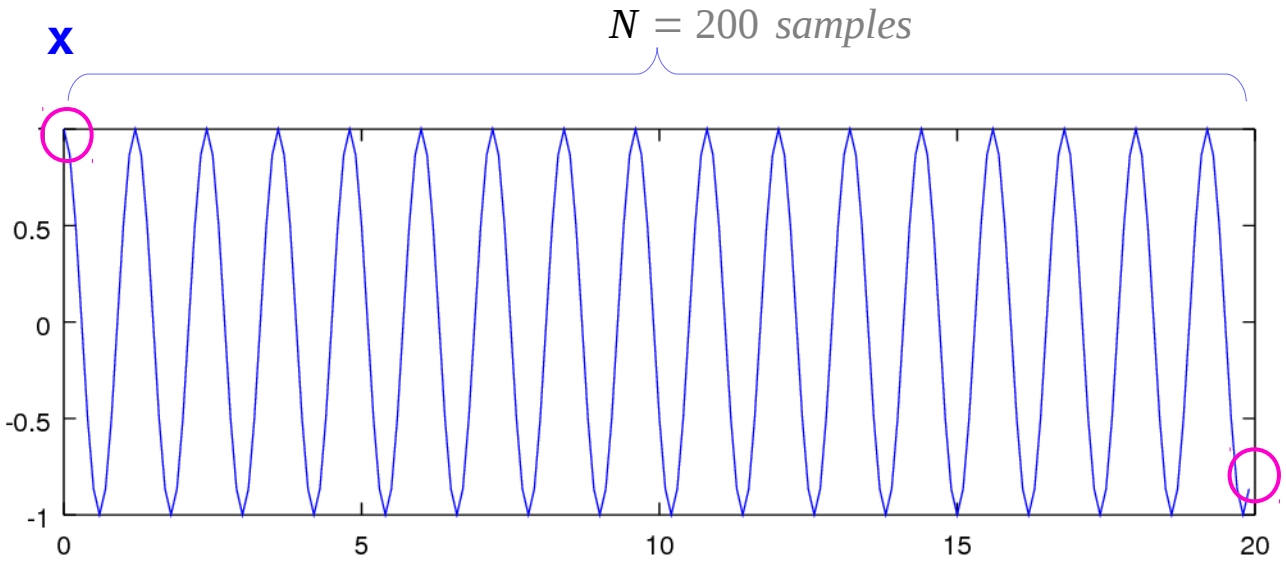


FFT of a cosine wave – non-integer multiple period

$\cos(5\pi/3 \cdot t)$

$\cos(5\pi/3 \cdot nT_s)$

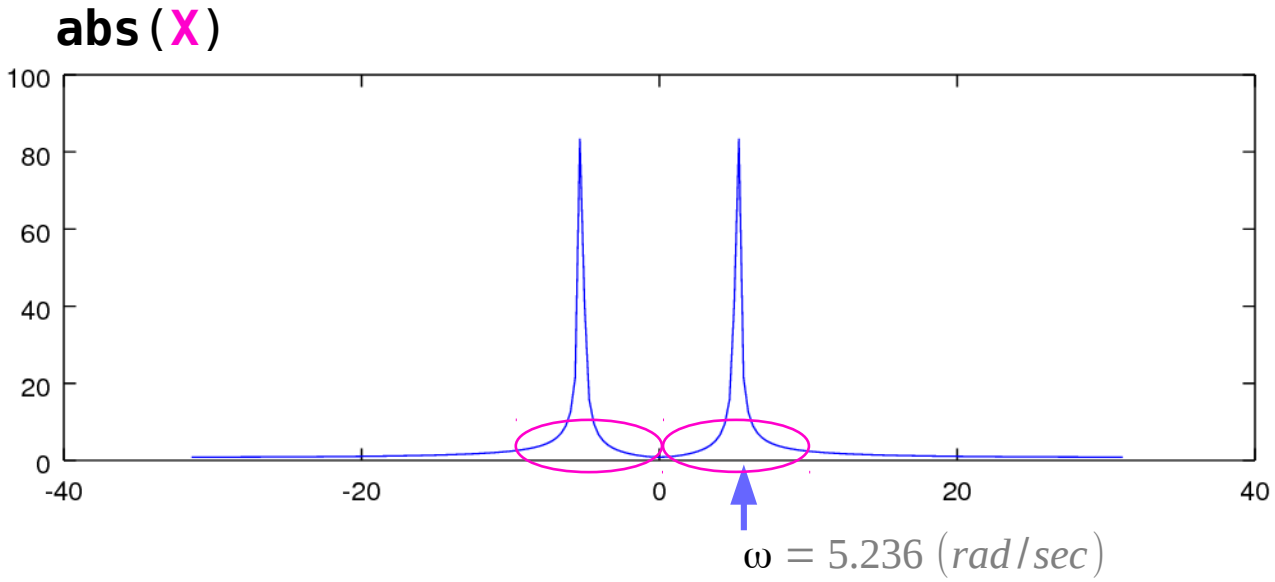
$\cos(0.1667\pi n)$



$\hat{\omega} = 0.1667\pi = 0.5236$

$\omega = \hat{\omega} \cdot f_s = 0.1667\pi \cdot 10 = 5.236$

$\hat{\omega} = \omega \cdot T_s = 1.667\pi \cdot 0.1 = 0.5236$



$\omega = k\,\omega_0$

$$x(t) = \cos(\pi \cdot t) = \frac{1}{2}\left(e^{+j\pi t} + e^{-j\pi t}\right) \qquad N = 200, \quad T_s = 0.1$$

$$\omega = \frac{2\pi}{T} = \pi \qquad T = \frac{2\pi}{\omega} = 2$$

$$\omega_0 = \frac{2\pi}{T_0} = 0.1\,\pi \qquad T_0 = N\,T_s = 200 \cdot 0.1 = 20$$

$$k = 10$$

$$C_{10} = \frac{1}{2} \quad (\omega = 10\,\omega_0 = \pi)$$

Fourier Series and Spectrum

$$x(t) = \cos(\pi \cdot t) = \frac{1}{2}\left(e^{+j\pi t} + e^{-j\pi t}\right)$$

$$\omega = \frac{2\pi}{T} = \pi \qquad \omega_0 = \frac{2\pi}{T_0} = 0.1\,\pi$$

$$T = \frac{2\pi}{\omega} = 2 \qquad T_0 = N\,T_s = 200 \cdot 0.1 = 20$$

$$T_0 = N\,T_s$$

$$\omega_0 = \frac{2\pi}{T_0} = \frac{2\pi}{N\,T_s} = \left(\frac{2\pi}{N}\right)\left(\frac{1}{T_s}\right) = \hat{\omega}_0 f_s$$

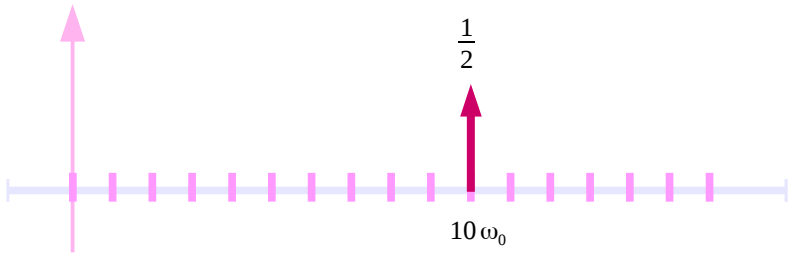
$$C_k = \frac{1}{2T_0}\int_0^{T_0}\left(e^{+j\omega t} + e^{-j\omega t}\right) e^{-jk\omega_0 t}\,dt$$

$$= \frac{1}{2T_0}\int_0^{T_0}\left(e^{+j(\omega - k\omega_0)t} + e^{-j(\omega + k\omega_0)t}\right) dt$$

$$C_k = \frac{1}{T_0}\int_0^{T_0} x(t)\, e^{-jk\omega_0 t}\,dt$$

$$C_{10} = \begin{cases} 0.5 & (\omega = \pm 10 \cdot \omega_0) \\ 0 & (\omega \neq \pm 10 \cdot \omega_0) \end{cases} \rightarrow k = 10$$

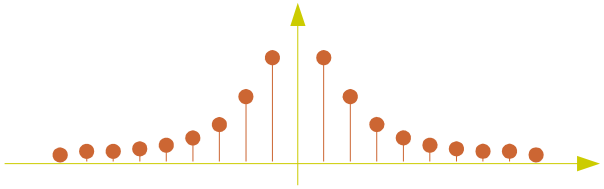
Only when ω is an integer multiples of ω_0



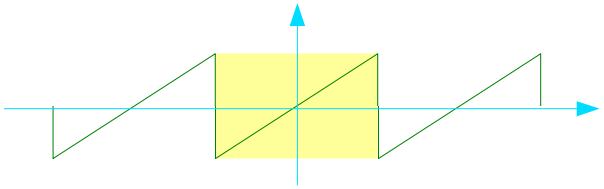
Continuous Time – **CTFS** Computation

Continuous Time Fourier Series

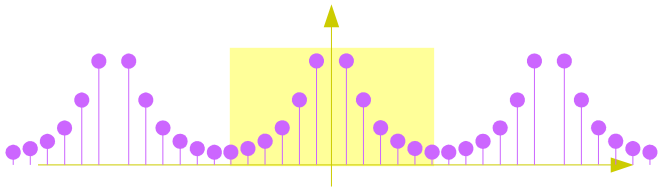
$$C_k = \frac{1}{T}\int_0^T x(t)\, e^{-jk\omega_0 t}\, dt \quad \Leftrightarrow \quad x(t) = \sum_{k=-\infty}^{+\infty} C_k\, e^{+jk\omega_0 t}$$

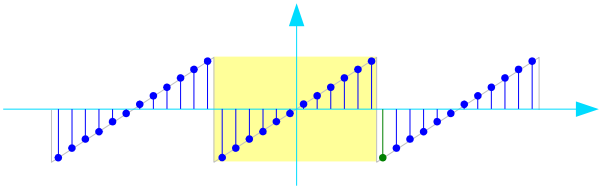


$$\omega_0 = \frac{2\pi}{T}$$



$$C_k \approx \frac{1}{N}\,\mathbf{DFT}\{x(nT_s)\} \qquad x(nT_s) \approx N\,\mathbf{IDFT}\{C_k\}$$





$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j\frac{2\pi}{N}kn} \quad \Leftrightarrow \quad x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]\, e^{+j\frac{2\pi}{N}kn}$$

X[k] and C_k

$$C_k \approx \frac{1}{N} \boldsymbol{DFT}\{x(nT_s)\}$$

$$C_k \approx \frac{1}{N} X[k]$$

Scale down

$$N C_k \approx \boldsymbol{DFT}\{x(nT_s)\}$$

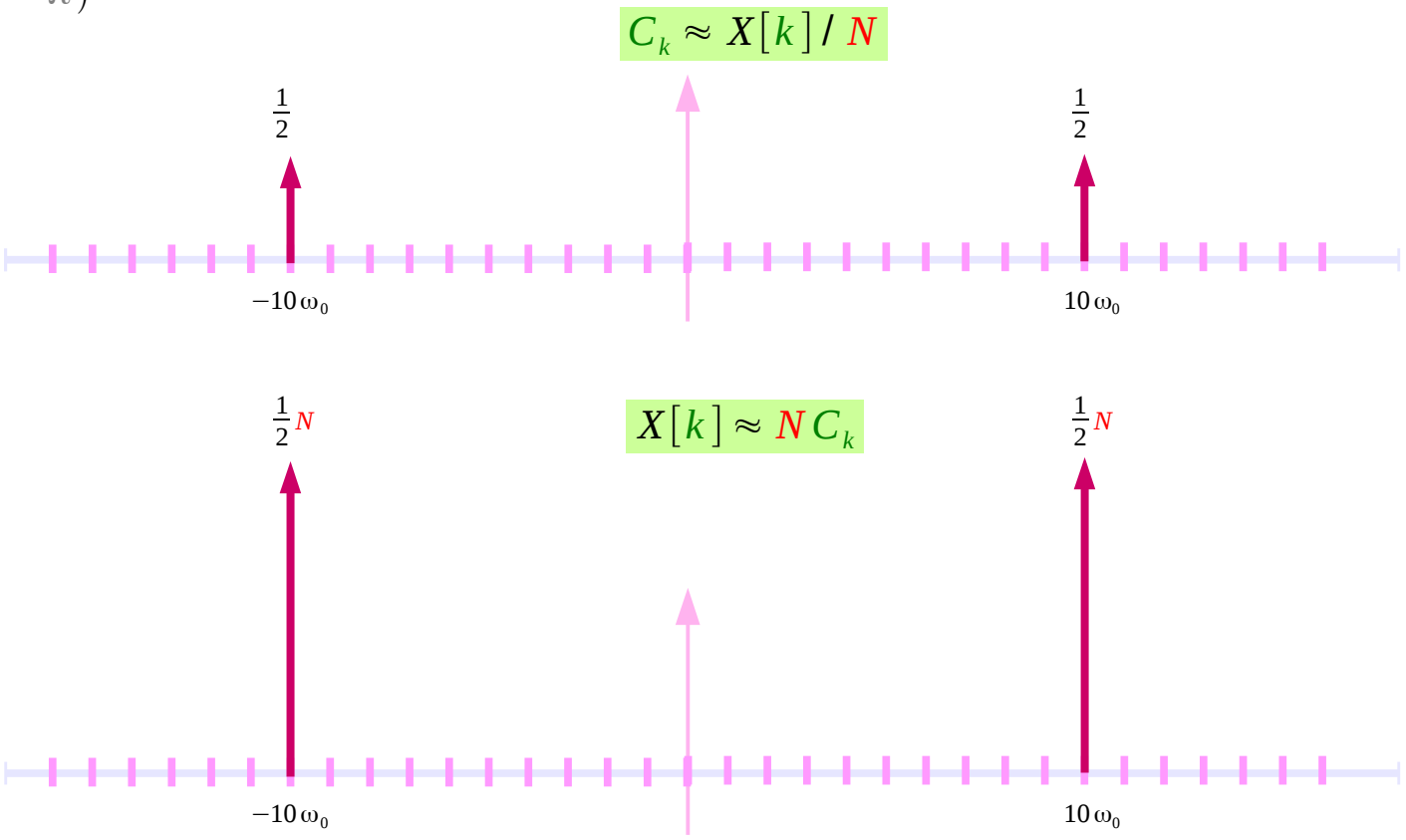
$$X[k] \approx N C_k$$

Scale up

X[10] and C_{10}

$$x(t) = \cos(\pi \cdot t) = \frac{1}{2}\left(e^{+j\pi t} + e^{-j\pi t}\right)$$

$$C_{10} = \frac{1}{2} \quad (\omega = 10\,\omega_0 = \pi)$$



Magnitude of a cosine spectrum

$$\cos(\pi \cdot t) = \frac{1}{2}\left(e^{+j\pi t} + e^{-j\pi t}\right)$$

$$\cos(5\pi/3 \cdot n T_s) = \cos(\omega n T_s)$$

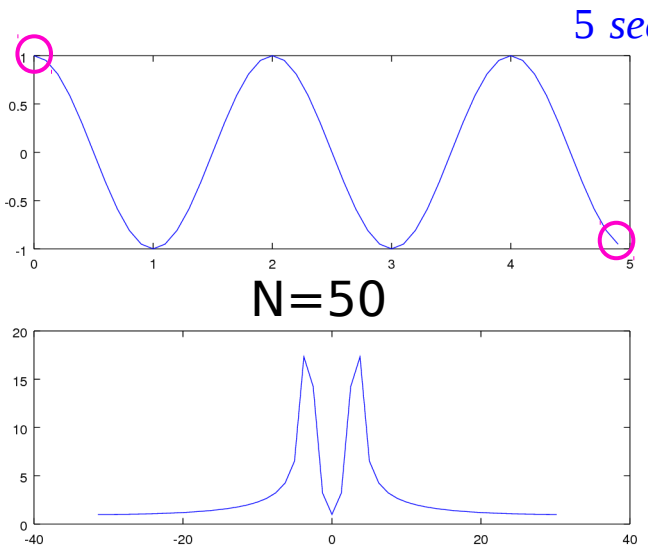
$$\cos(0.1667\pi n) = \cos(\hat{\omega} n)$$

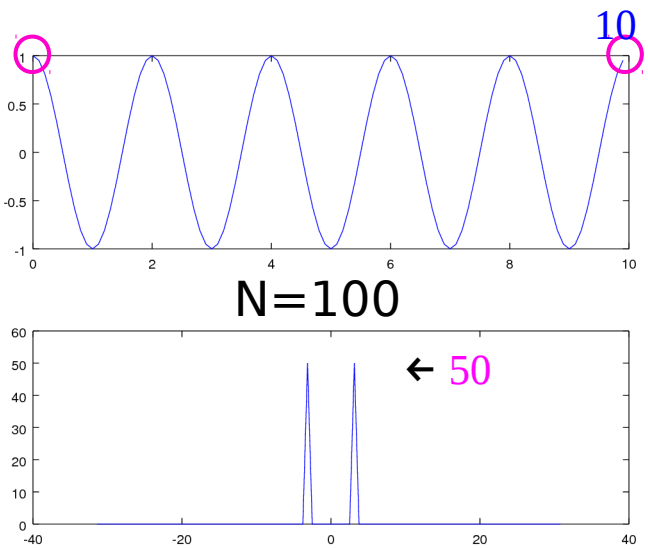
$$\hat{\omega} = 0.1667\pi = 0.5236$$

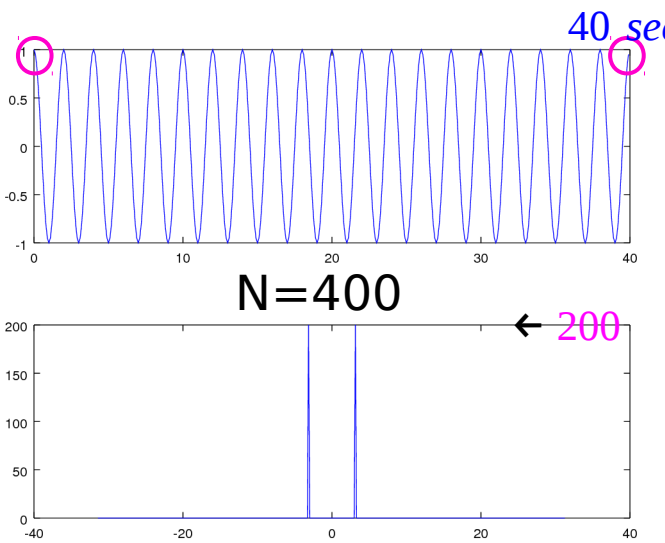
$$\omega = \hat{\omega} \cdot f_s = 0.1667\pi \cdot 10 = 5.236$$

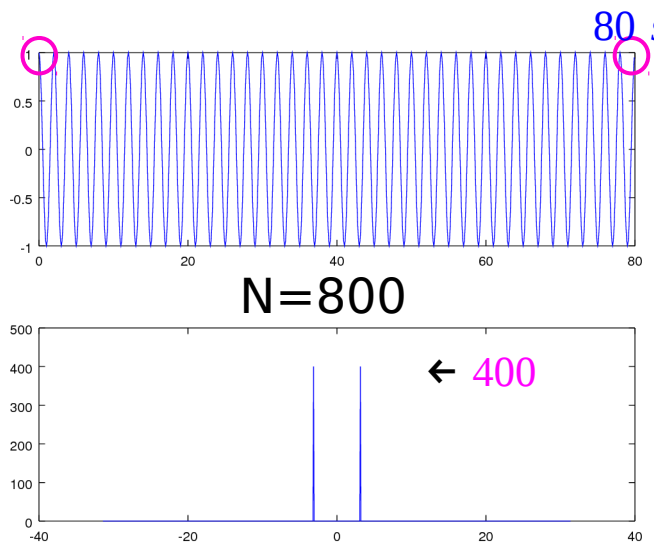
$$\hat{\omega} = \omega \cdot T_s = 1.667\pi \cdot 0.1 = 0.5236$$

$$N C_k \approx X[k]$$









Generating a cosine signal with added noise

```
pkg load communications

N = 400;

fs = 10;
t = (0:1:N-1)/fs;

x = 0.8*cos(2*pi*0.8*t);
xn = awgn(x, -2);

X = fft(x);
Xn = fft(xn);
```

$$\begin{cases} \omega = 4.9770 \\ amp = 160 \end{cases} \Rightarrow \begin{cases} f = 0.8 \\ A = 160 \end{cases}$$

awgn : additive white Gaussian noise

$$T_s = 0.1\ (sec) \qquad f_s = \frac{1}{T_s} = 10\ (Hz)$$

$$T_0 = N T_s = 400 \cdot 0.1 = 40\ (sec)$$

$$\omega_0 = \frac{2\pi}{T_0} = \frac{2\pi}{N T_s} = \frac{2\pi}{40} = 0.15708\ (rad/sec)$$

$$\hat{\omega}_0 = \omega_0 T_s = \frac{2\pi}{N} = \frac{2\pi}{400} = 0.015708$$

$$\omega = 2\pi f = 4.9770 \rightarrow f = 0.79211 \approx 0.8$$

$$C_k = \frac{1}{N} X[k]$$

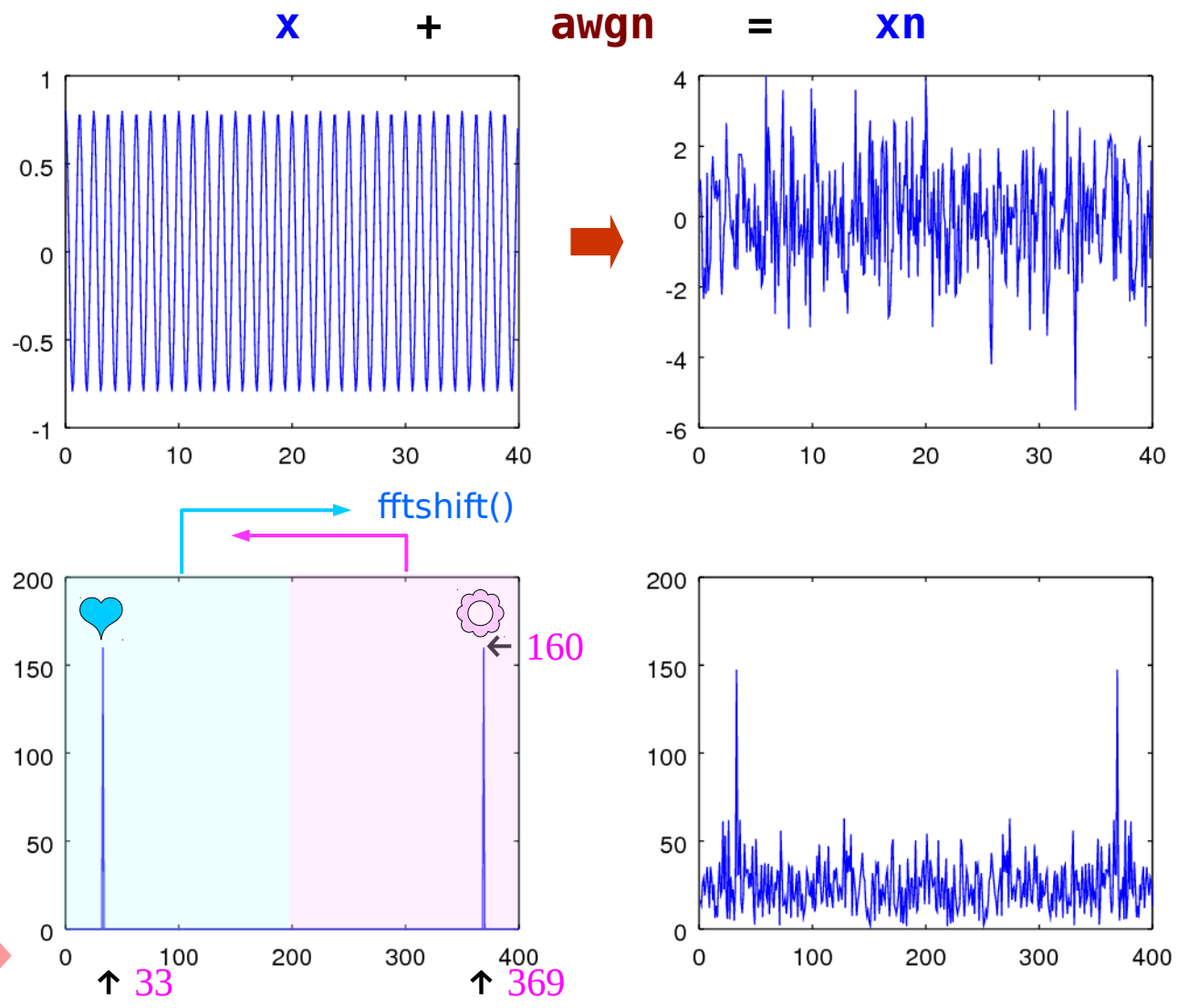
$$\frac{A}{2} = \frac{1}{400} amp = \frac{1}{400} |X[k]| = \frac{1}{400} 160$$

$$\rightarrow \quad A = \frac{160}{400} * 2 = 0.7953 \approx 0.8$$

B Ninness, Spectral Analysis using the FFT

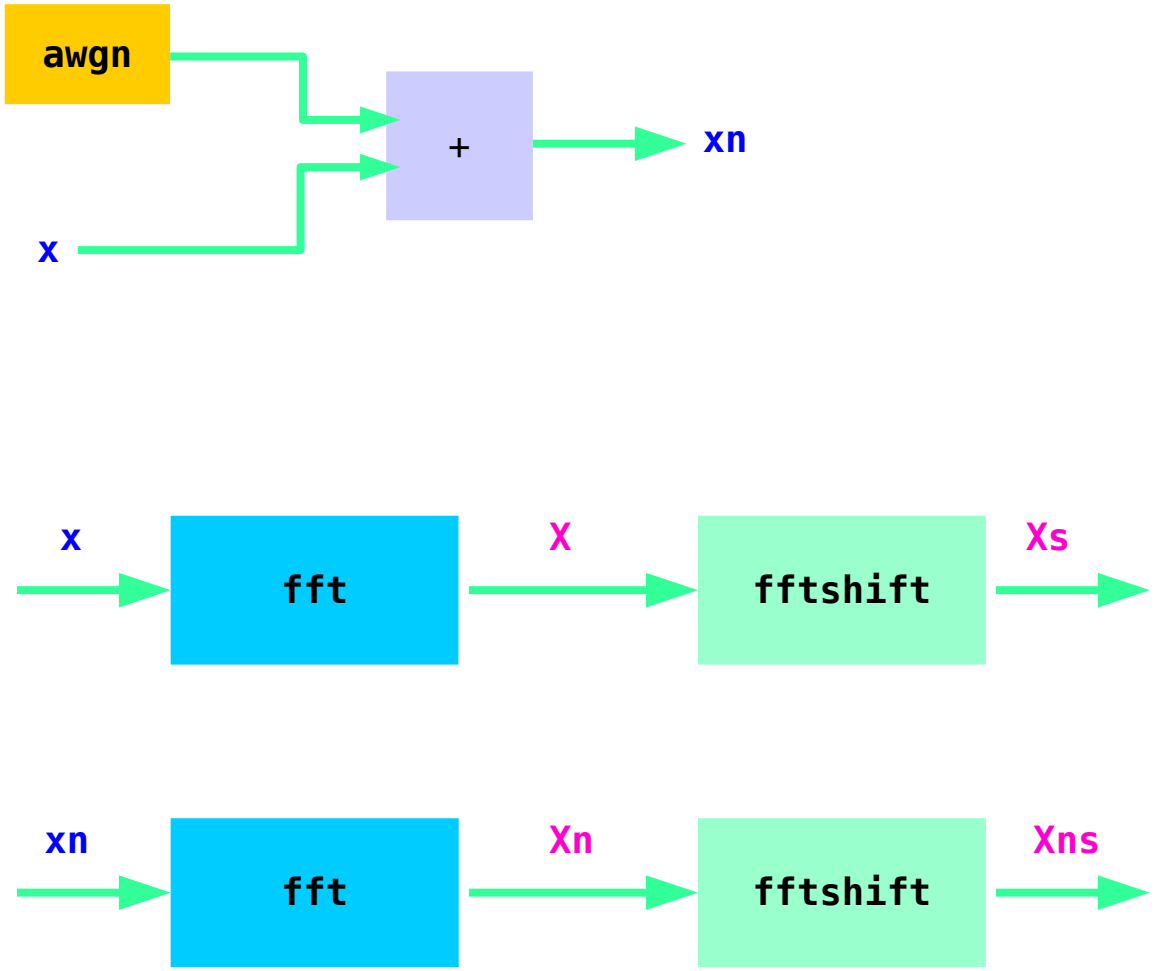
FFT of a noisy cosine

```
subplot(2, 2, 1);
plot(t, x);

subplot(2, 2, 2);
plot(t, xn);

subplot(2, 2, 3);
plot( abs(X) );

subplot(2, 2, 4);
plot( abs(Xn) );

pause
```



B Ninness, Spectral Analysis using the FFT

X, Xn, and Xns

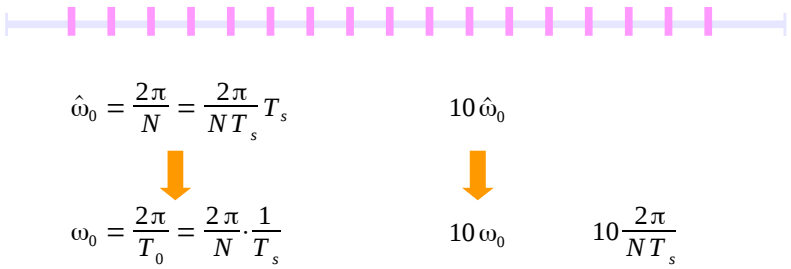


Computing frequency domain scales

```
ws = 2*pi/N;
wnorm = -pi:ws:+pi;
wnorm = wnorm(1:N);
w = wnorm * fs;
```

$$\hat{\omega} = \omega \cdot T_s$$

$$\hat{\omega} \cdot f_s = \omega$$



ws : $\hat{\omega}_0 = \frac{2\pi}{N}$

wnorm : $[\ -\pi,\ \cdots\ ,\ k\,\hat{\omega}_0,\ \cdots\ ,\ +\pi]$

w : $[-\pi f_s,\ \cdots\ ,\ k\,\omega_0,\ \cdots\ ,+\pi f_s]$

$$\omega_0 = \frac{2\pi}{T_0} = \frac{2\pi}{N T_s}$$

$$\hat{\omega}_0 = 2\pi\left(\frac{T_s}{T_0}\right) = \frac{2\pi}{N}$$

$$\omega_0 = \hat{\omega}_0 \cdot f_s$$

$$\hat{\omega}_0 = \omega_0 \cdot T_s$$

$$T_s = 0.1\ (sec) \qquad f_s = \frac{1}{T_s} = 10\ (Hz)$$

$$T_0 = N T_s = 400 \cdot 0.1 = 40\ (sec)$$

$$\omega_0 = \frac{2\pi}{T_0} = \frac{2\pi}{N T_s} = \frac{2\pi}{40} = 0.15708\ (rad/sec)$$

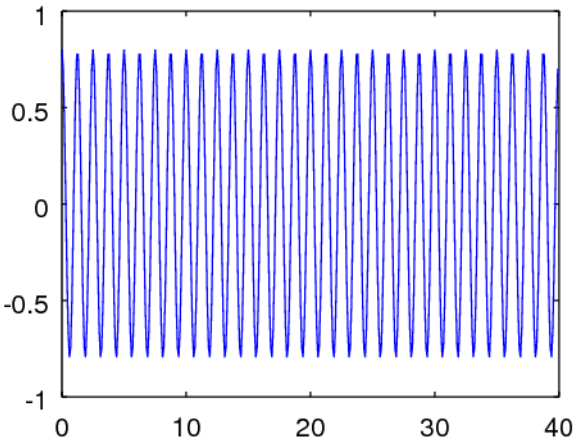
$$\hat{\omega}_0 = \omega_0 T_s = \frac{2\pi}{N} = \frac{2\pi}{400} = 0.015708$$

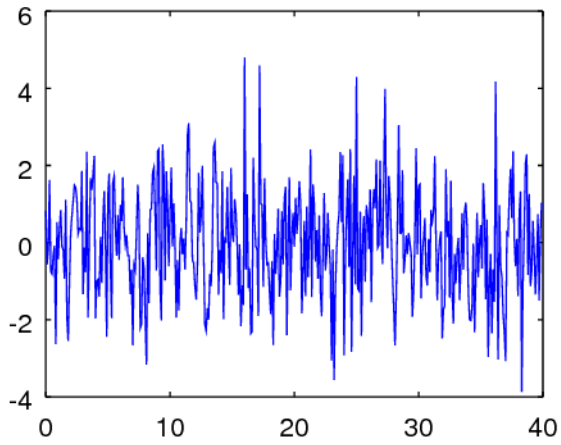
B Ninness, Spectral Analysis using the FFT

fftshift

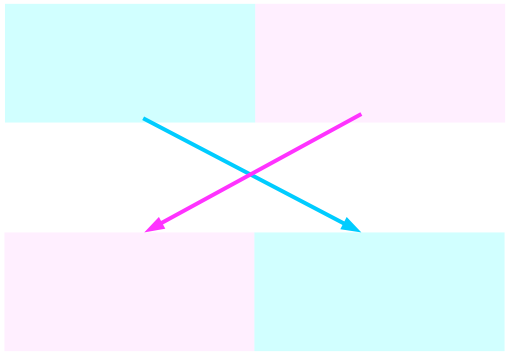
```
Xs = fftshift(X);
Xns = fftshift(Xn);

subplot(2,2,3);
plot(w, abs(Xs));
subplot(2,2,4);
plot(w, abs(Xns));
```

x + awgn = xn

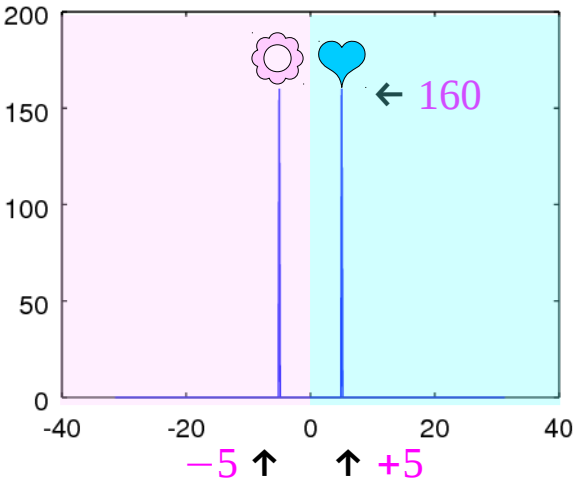




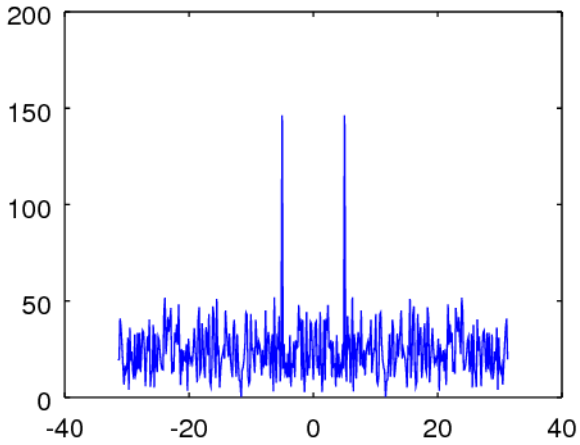
fftshift()



Xs = fftshift(X);



Xns = fftshift(Xn);

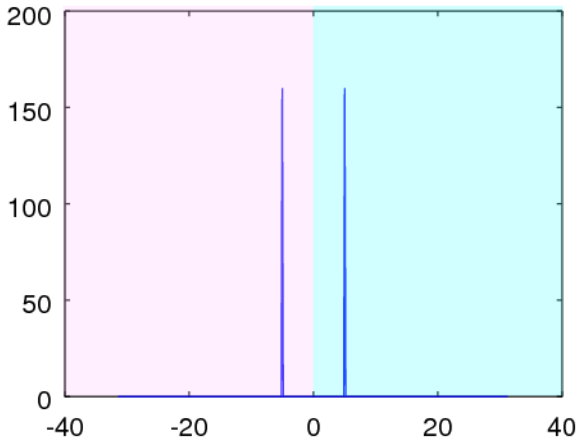


B Ninness, Spectral Analysis using the FFT

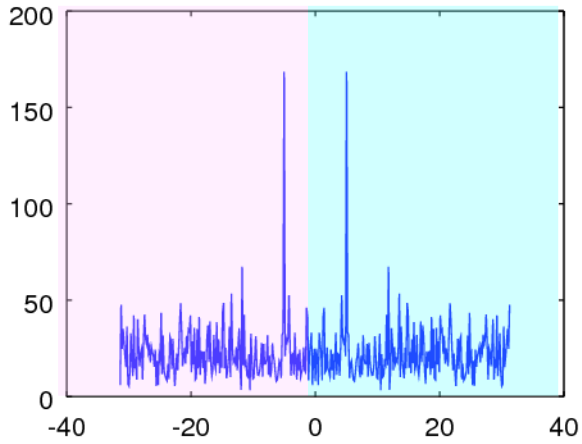
Self inverting function : fftshift()

```
Xs ← fftshift(X);
Xns ← fftshift(Xn);

X ← fftshift(Xs);
Xn ← fftshift(Xns);


X ← fftshift(fftshift(X));
Xn ← fftshift(fftshift(Xn));
```

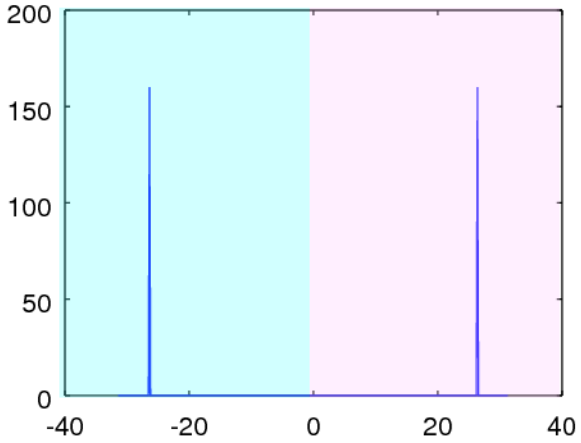
Xs = fftshift(X);



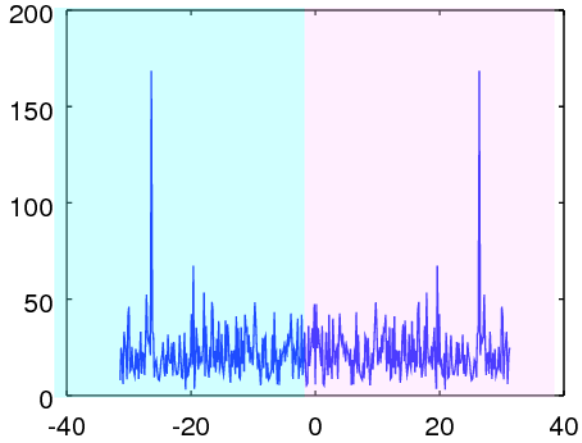
Xns = fftshift(Xn);



fftshift(Xs);



fftshift(Xns);



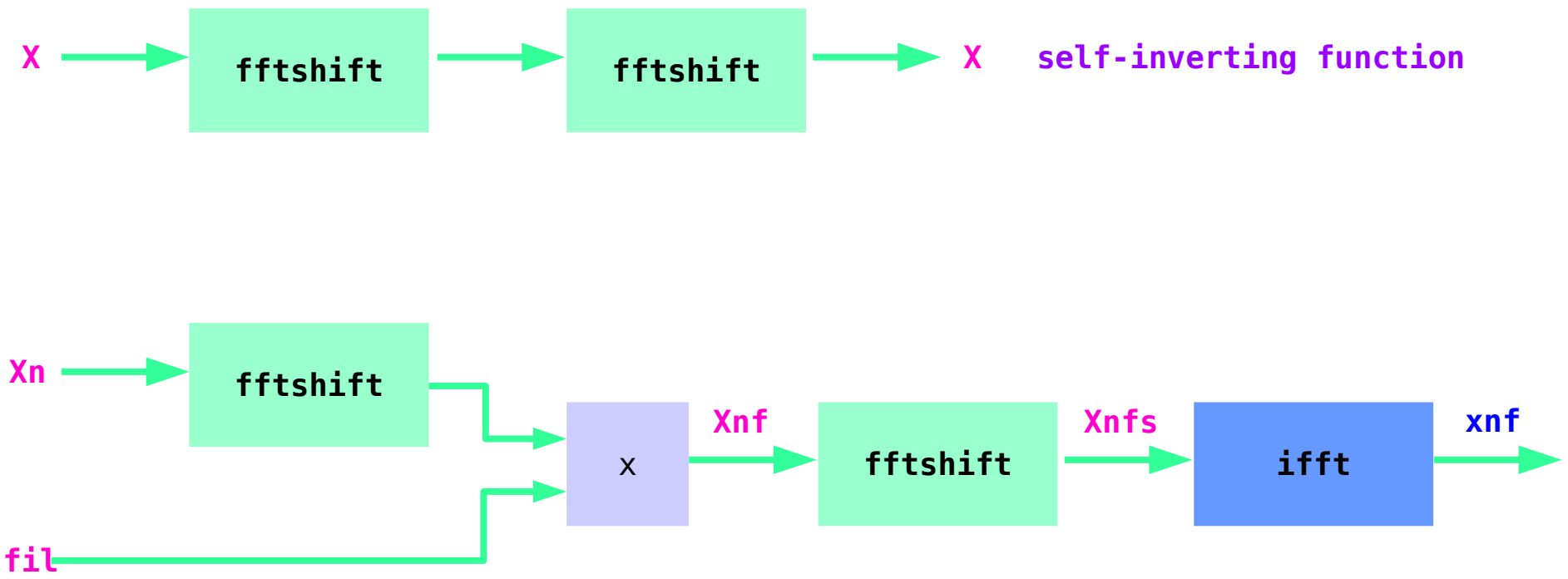
Octave ginput()

```
% close all figure windows
% must use this toolkit to use ginput()

close all;
graphics_toolkit(fltk);

plot(w, abs(fftshift(Xn)));

[freq, amp, buttons]= ginput()
```

```
freq =

  -5.3088
   5.0138
   4.8295
   4.8295
   4.8295
   4.6452
   4.6452

amp =

   142.448
   141.864
    42.536
    42.536
    42.536
    41.952
    41.952

buttons =

   1
   1
   1
   1
   1
   3
   3
```

```
>> available_graphics_toolkits
ans =
{
  [1,1] = fltk
  [1,2] = gnuplot
  [1,3] = qt
}
```

fftshift() before ifft()



Filtering

```
% fil = [ zeros(1,160), ones(1,15), zeros(1,50),
          ones(1,15), zeros(1,160)];
fil = [ zeros(1,166), ones(1,5), zeros(1,59),
        ones(1,5), zeros(1,165)];
```

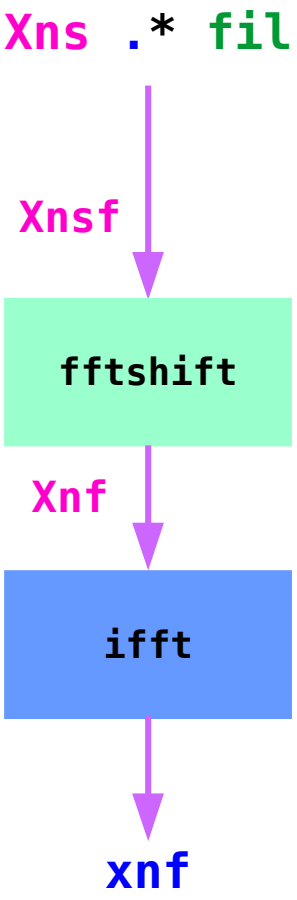
```
Xnf= fil .* Xns;
xnf= ifft(fftshift(Xnf));

subplot(2, 2, 3);
plot(w, abs(fftshift(Xn)),'b',  w, 160*fil,'-.r');

subplot(2, 2, 4);
plot(t, real(xnf));
```

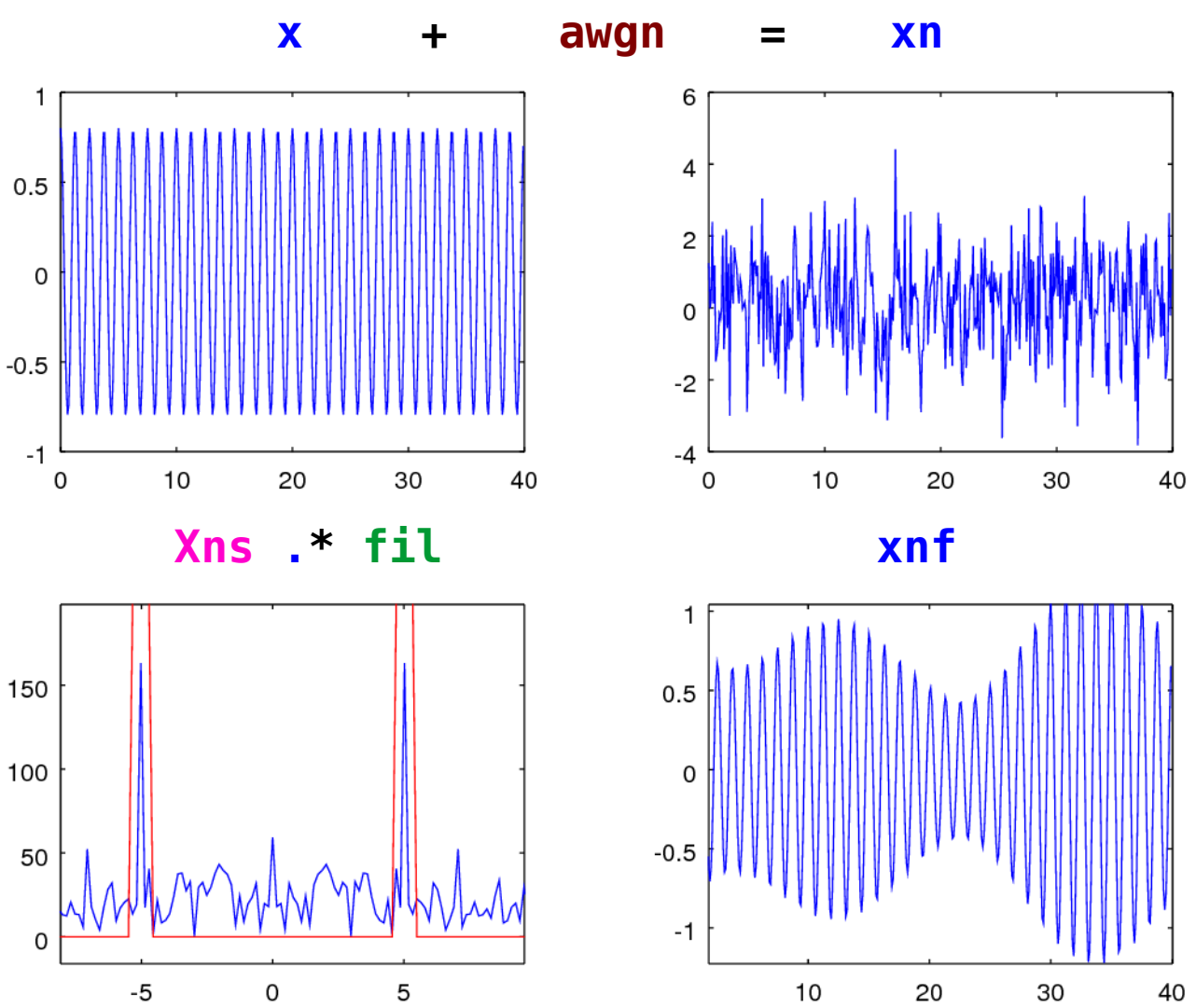
black

red

B Ninness, Spectral Analysis using the FFT

Filtering results





B Ninness, Spectral Analysis using the FFT

Window Function

```
win = hamming(N);

Hamming Window function returns a column vector

win(:) → Column access of the column vector
xn(:) →  Column access of the row vector

(:) does not transpose the original input matrix


To get the correct element-wise multiplication

xw = win(:) .* xn(:);

xn : a row vector
xw : a column vector
```

B Ninness, Spectral Analysis using the FFT

Colon Syntax (1)

As a special case, when a colon is used as a single index, the output is a column vector containing all the elements of the vector or matrix. For example:

```
a(:)      # result is a column vector
a(:)'     # result is a row vector
```

The above two code idioms are often used in place of reshape when a simple vector, rather than an arbitrarily sized array, is needed.

Given the matrix

```
a = [1, 2; 3, 4]
```

all of the following expressions are equivalent and select the first row of the matrix.

```
a(1, [1, 2])    # row 1, columns 1 and 2
a(1, 1:2)       # row 1, columns in range 1-2
a(1, :)         # row 1, all columns
```

https://www.gnu.org/software/octave/doc/interpreter/Index-Expressions.html

Colon Syntax (2)

```
>> A = [1, 2; 3, 4]
A =

   1   2
   3   4
```

```
>> A(:)
ans =

   1
   3
   2
   4
```

```
>> >> B = [1, 2, 3; 4, 5, 6; 7, 8, 9]
B =

   1   2   3
   4   5   6
   7   8   9
```

```
>> B(:)
ans =

   1
   4
   7
   2
   5
   8
   3
   6
   9
```

```
>> B(1, :)
ans =

   1   2   3

>> B(2, :)
ans =

   4   5   6

>> B(:, 1)
ans =

   1
   4
   7

>> B(:, 2)
ans =

   2
   5
   8
```

B Ninness, Spectral Analysis using the FFT

Column-wise Access (1)

```
>> a = [1 2 3 ]
a =

   1   2   3

>> b = [2 4 6]
b =

   2   4   6

>> a .* b
ans =

    2    8   18

>> a * b
error: operator *:
nonconformant
arguments (op1 is
1x3, op2 is 1x3)
```

```
>> a(:)
ans =

   1
   2
   3

>> b(:)
ans =

   2
   4
   6

>> a(:) .* b(:)
ans =

    2
    8
   18

>> a(:) * b(:)
error: operator *:
nonconformant
arguments (op1 is
3x1, op2 is 3x1)
```

```
>> a(:)
ans =

   1
   2
   3

>> b
b =

   2   4   6

>> a(:) .* b
ans =

    2    4    6
    4    8   12
    6   12   18

>> a(:) * b
ans =

    2    4    6
    4    8   12
    6   12   18

>>
```

```
>> a
a =

   1   2   3

>> b(:)
ans =

   2
   4
   6

>> a .* b(:)
ans =

    2    4    6
    4    8   12
    6   12   18

>> a * b(:)
ans =  28
>>
```

B Ninness, Spectral Analysis using the FFT

Column-wise Access (2)

```
>> a = [1; 2; 3]
a =

   1
   2
   3

>> b = [2; 4; 6]
b =

   2
   4
   6

>> a .* b
ans =

    2
    8
   18

>> a * b
error: operator *:
nonconformant
arguments (op1 is
3x1, op2 is 3x1)
```

```
>> a(:)
ans =

   1
   2
   3

>> b(:)
ans =

   2
   4
   6

>> a(:) .* b(:)
ans =

    2
    8
   18

>> a(:) * b(:)
error: operator *:
nonconformant
arguments (op1 is
3x1, op2 is 3x1)
```

```
>> a(:)
ans =

   1
   2
   3

>> b
b =

   2
   4
   6

>> a(:) .* b
ans =

    2
    8
   18

>> a(:) * b
error: operator *:
nonconformant
arguments (op1 is
3x1, op2 is 3x1)
```

```
>> a
a =

   1
   2
   3

>> b(:)
ans =

   2
   4
   6

>> a .* b(:)
ans =

    2
    8
   18

>> a * b(:)
error: operator *:
nonconformant
arguments (op1 is
3x1, op2 is 3x1)
```

B Ninness, Spectral Analysis using the FFT

Window Function

```
win = hamming(N);
xw = win(:) .* xn(:);

Xw = fft(xw);
Xwf= fil(:) .* fftshift(Xw);
Xwf= ifft(fftshift(Xnf));
```

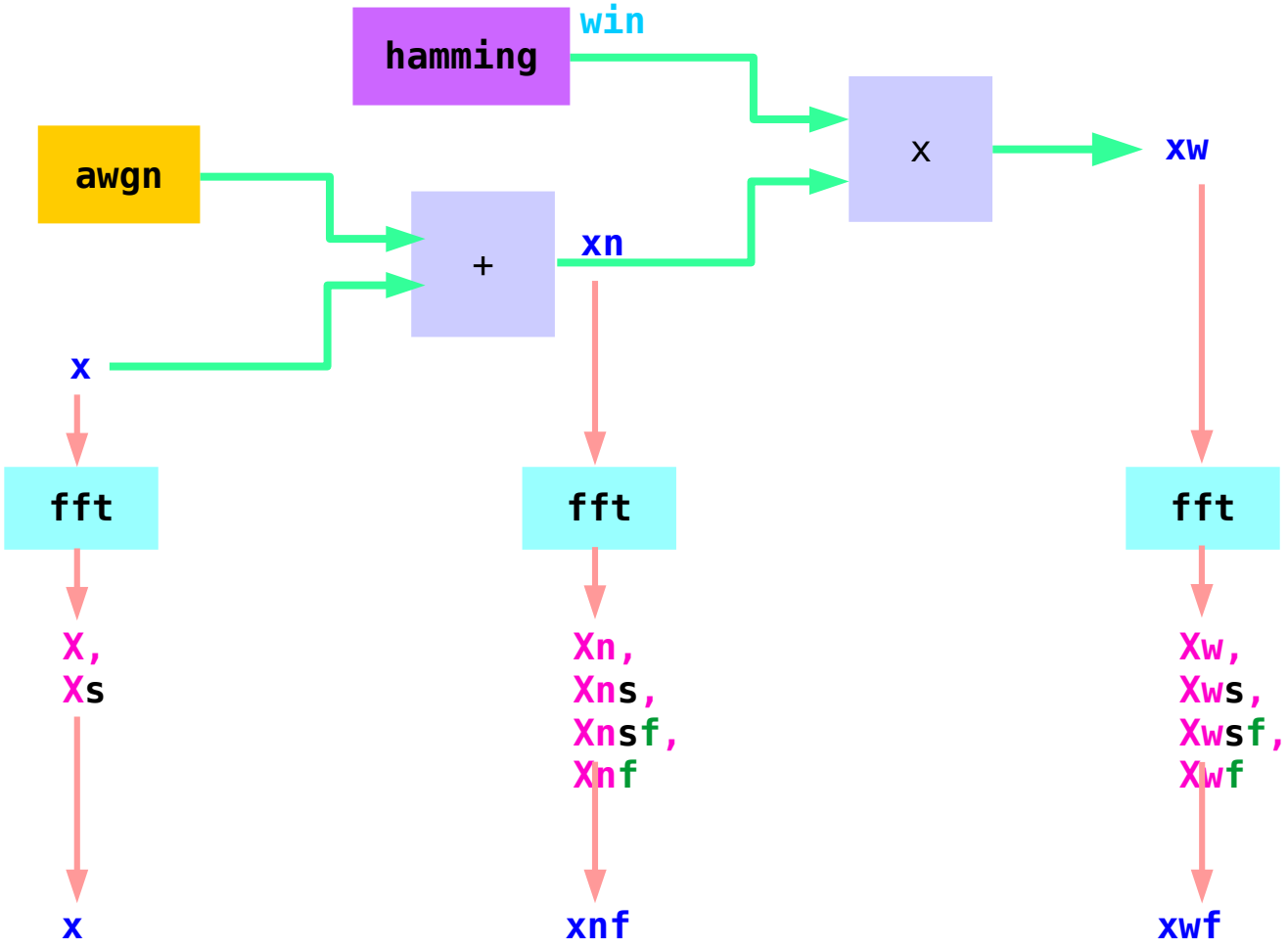
```
win: a column vector
xn : a row vector
xw : a column vector
Xw : a column vector
fil: a row vector
Xwf: a column vector
```

```
win_col = hamming(N);
win = win_col';
xw = win .* xn;

Xw = fft(xw);
Xws= fftshift(Xw);
Xwf= fil .* fftshift(Xw);
xwf= ifft(fftshift(Xwf));
```

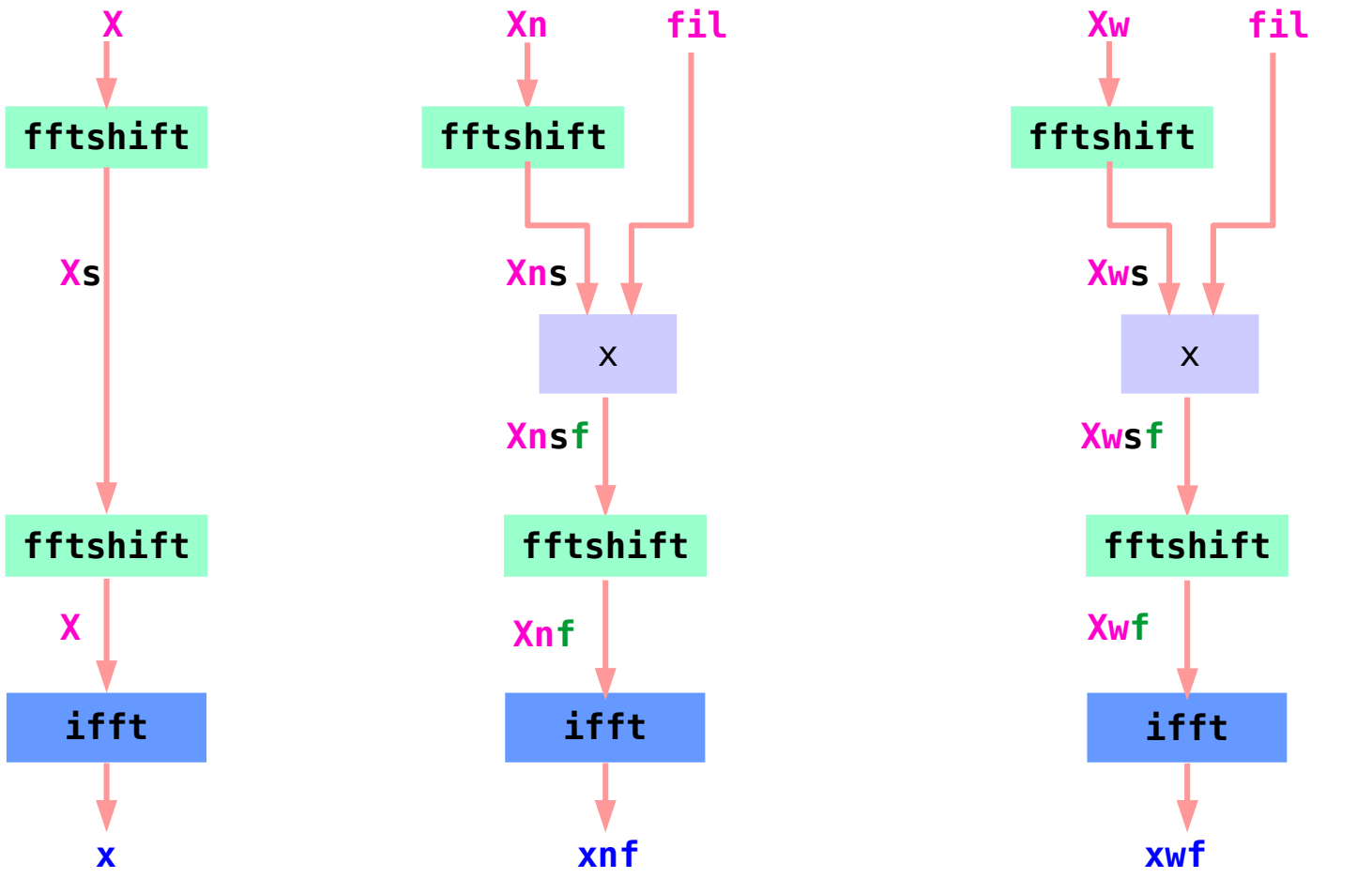
```
win: a row vector
xn : a row vector
xw : a row vector
Xw : a row vector
fil: a row vector
Xwf: a row vector
```

B Ninness, Spectral Analysis using the FFT

Tap Signals – time domain



Tap Signals – frequency domain



n : noise

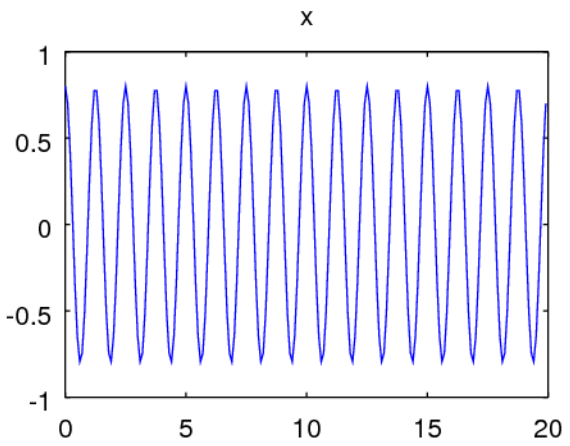
w : window

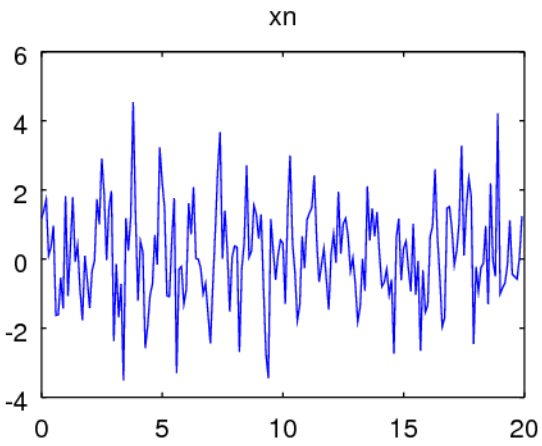
Window Result Plots

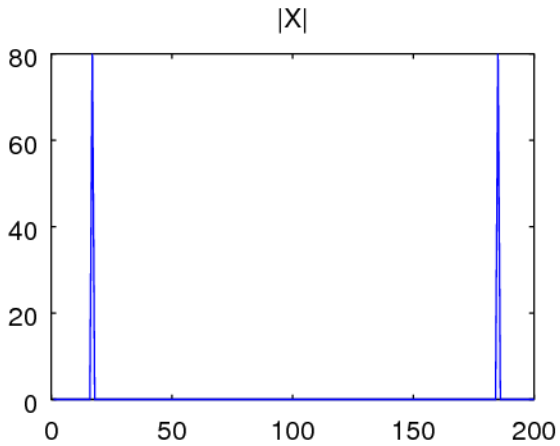
```
subplot(2, 2, 1);
plot(t, win);

subplot(2, 2, 2);
plot(t, xw);

subplot(2, 2, 3);
plot(w, abs(fftshift(Xw)),'b',   w, 200*fil,'-r');

subplot(2, 2, 4);
plot(t, real(xnf) ./ win(:) );


% plot(w, abs(fftshift(Xnf)),'b', w, 200*fil,'-r');
```

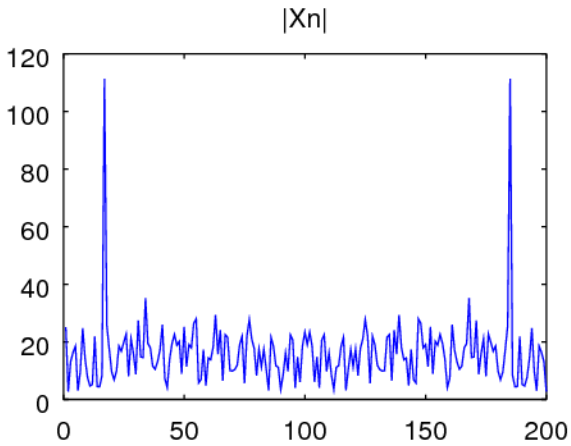
B Ninness, Spectral Analysis using the FFT

Window Result Plots (1)

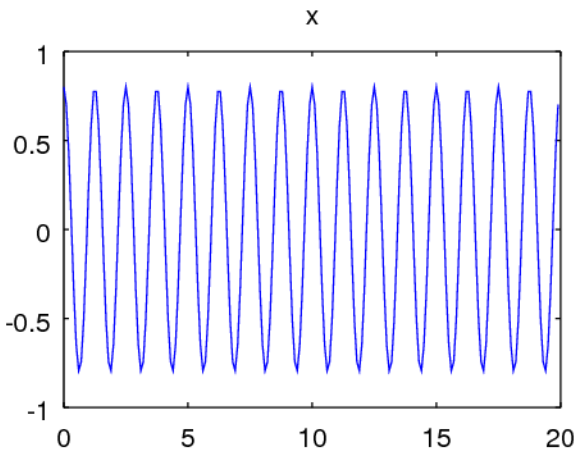


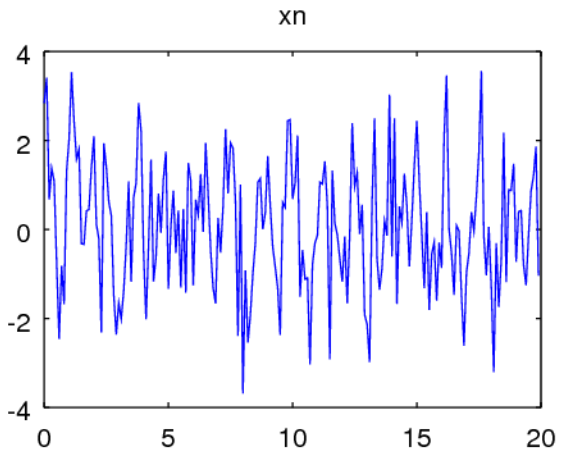


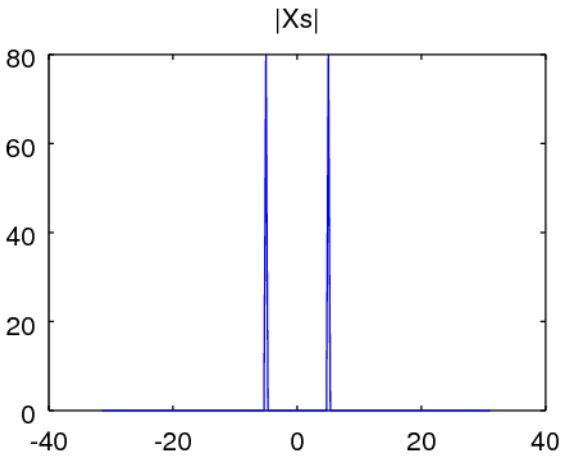


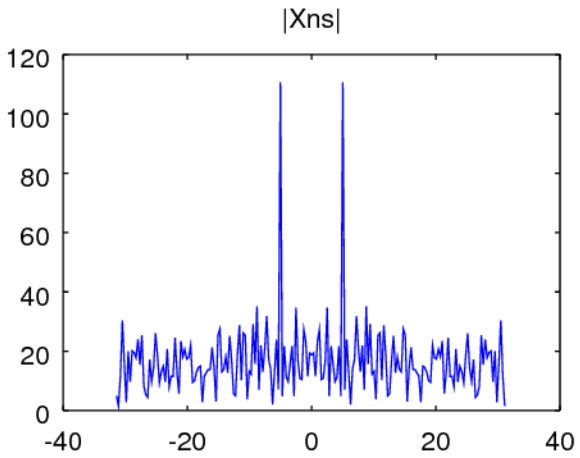


Window Result Plots (2)



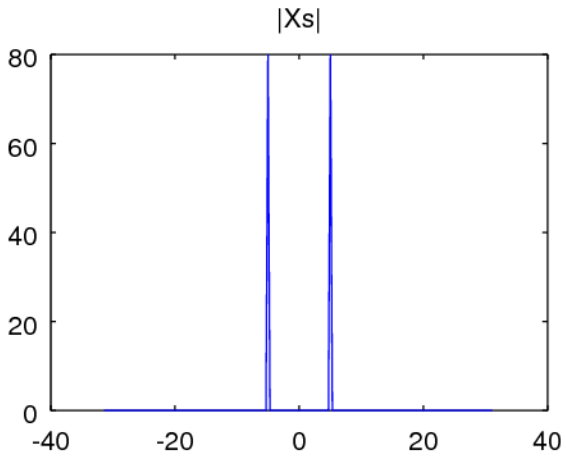


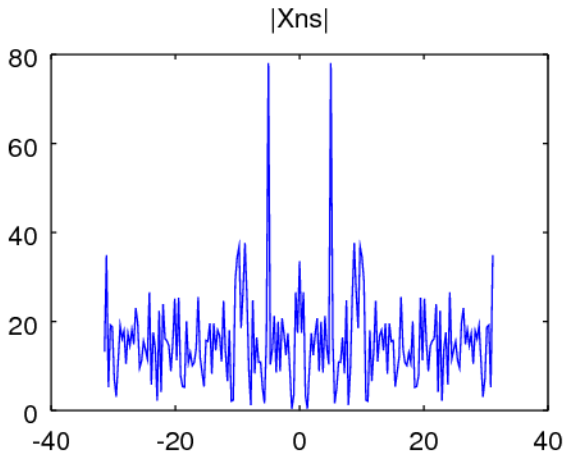


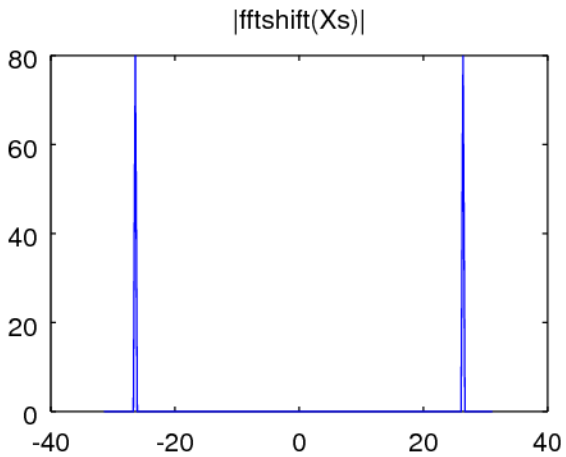


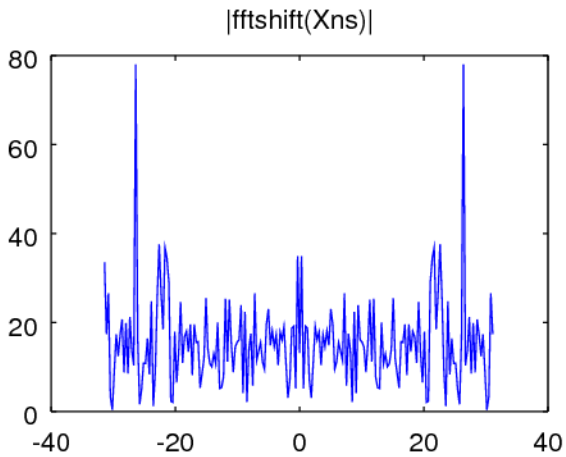
B Ninness, Spectral Analysis using the FFT

Window Result Plots (3)



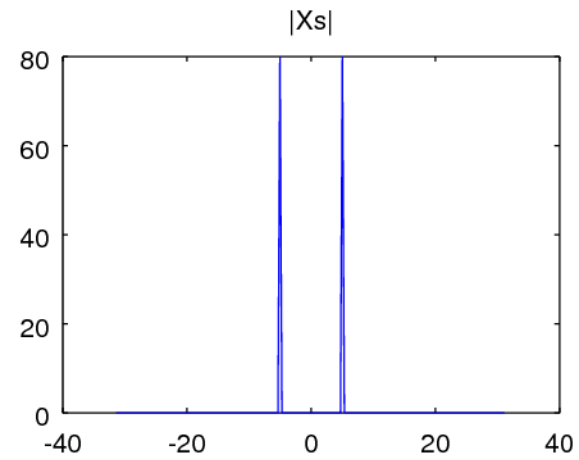


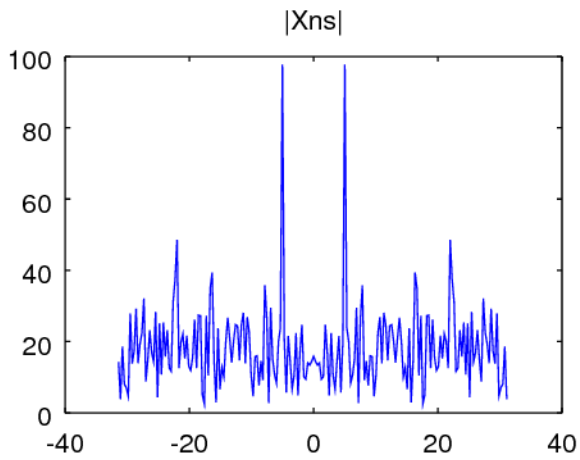


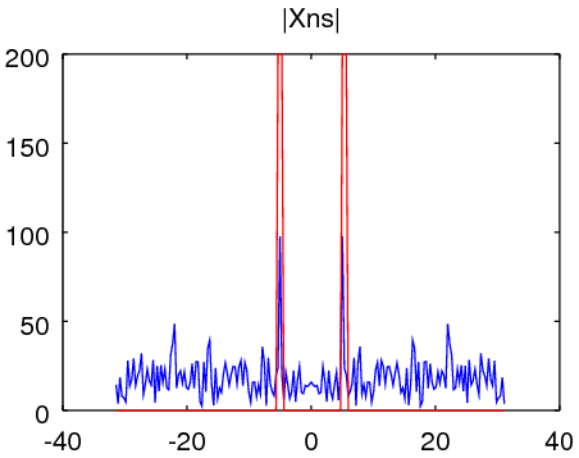


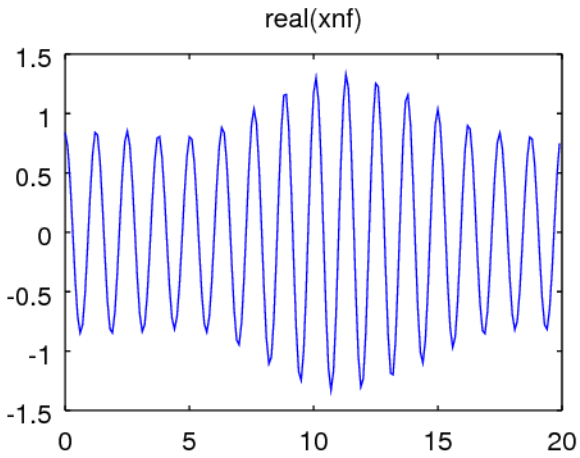
B Ninness, Spectral Analysis using the FFT

Window Result Plots (4)



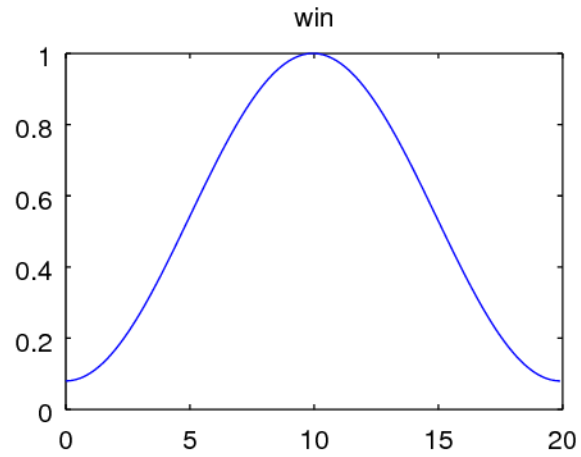


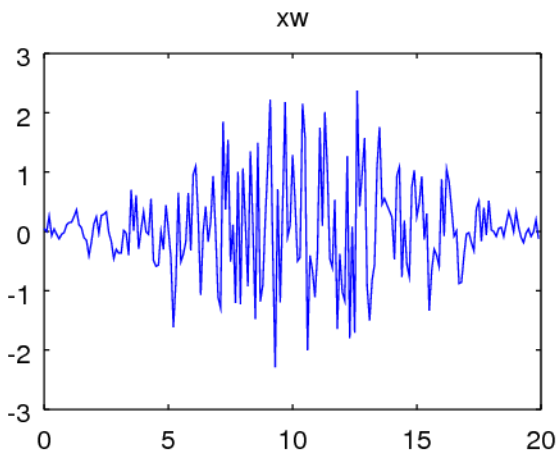


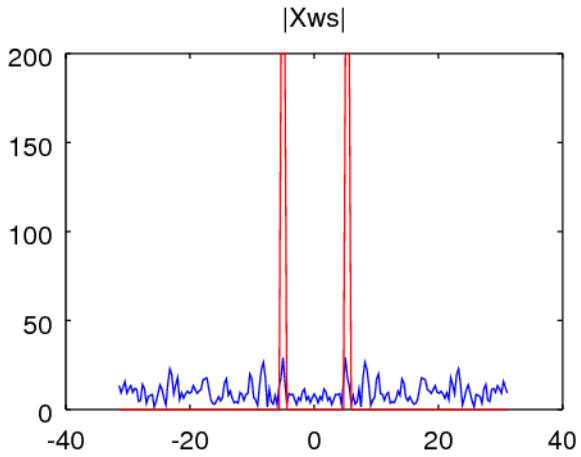


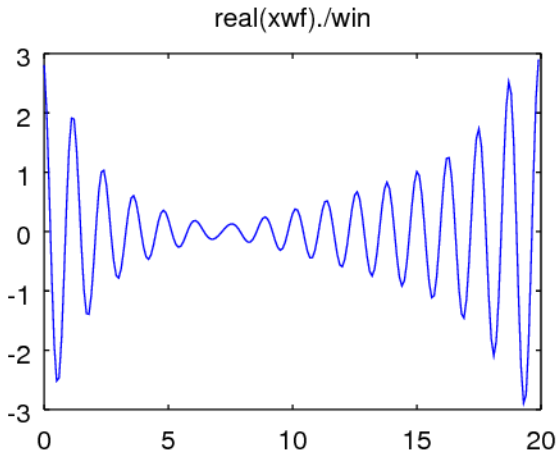
B Nimness, Spectral Analysis using the FFT

Window Result Plots (5)

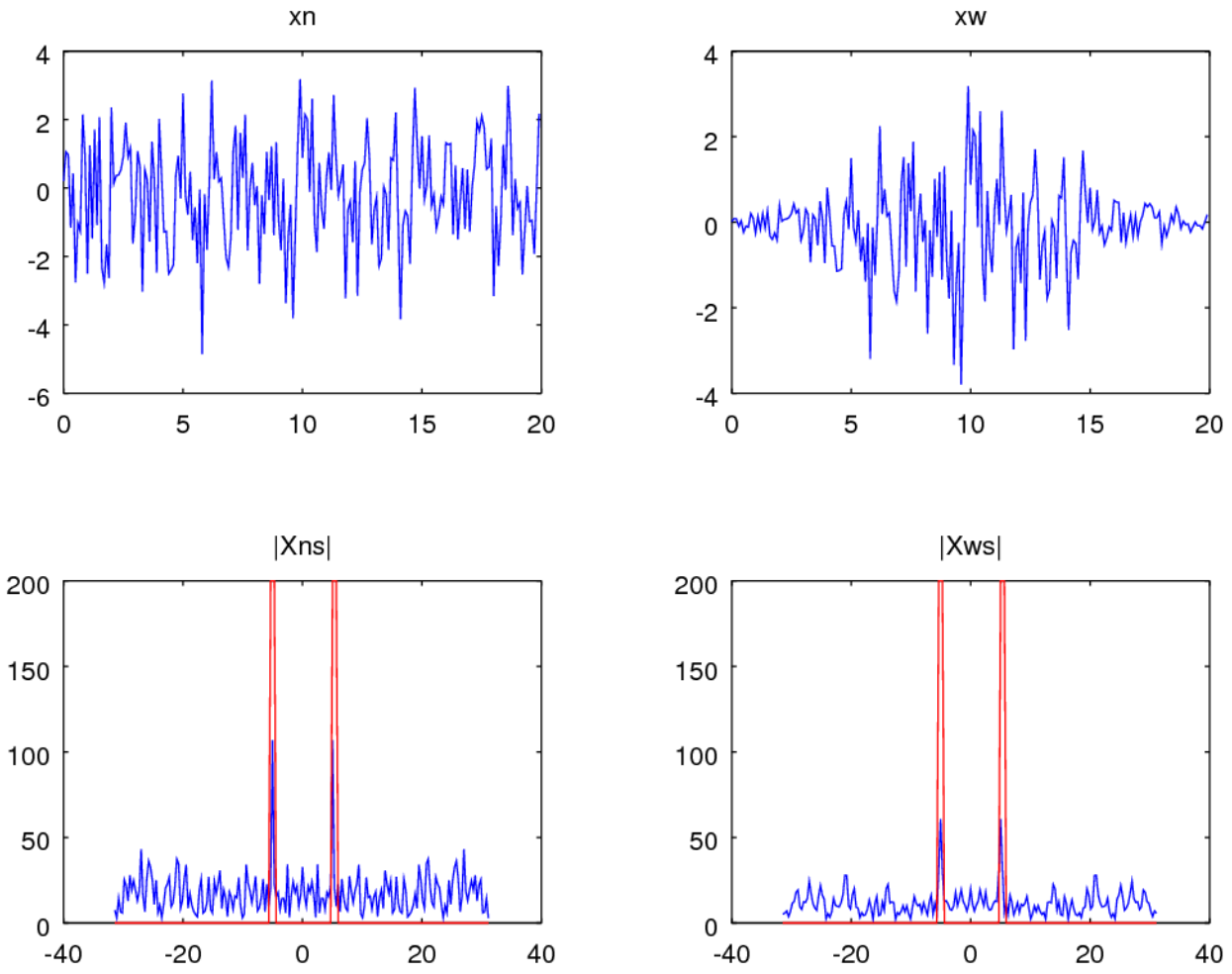






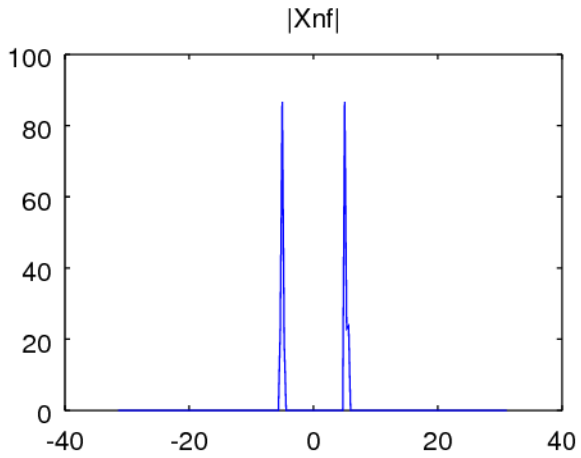


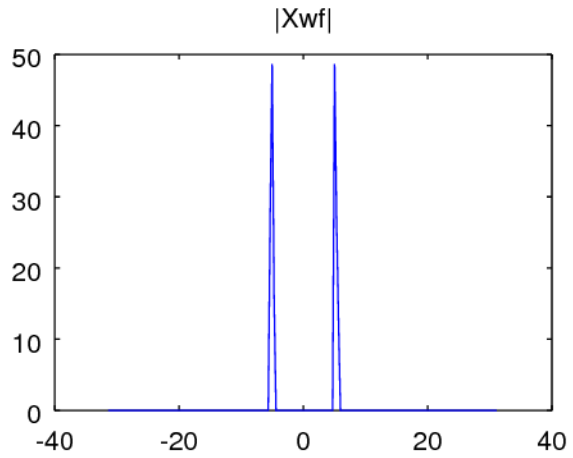
Window Result Plots (6)

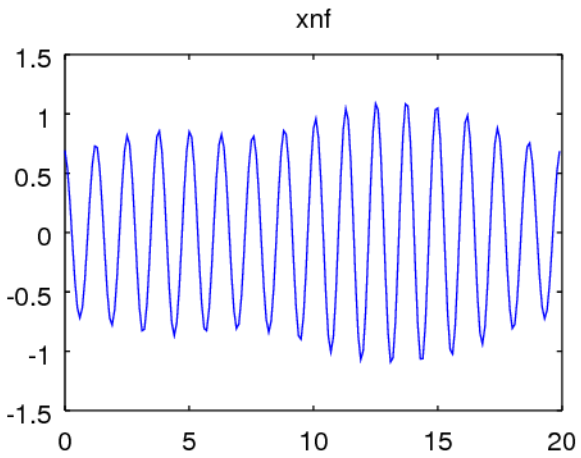


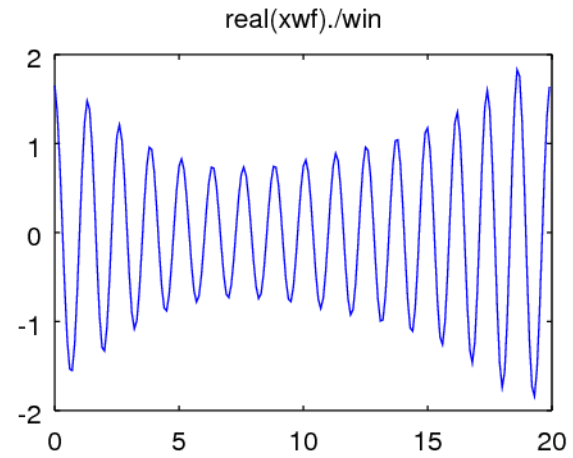
B Ninness, Spectral Analysis using the FFT

Window Result Plots (7)









B Ninness, Spectral Analysis using the FFT

Normalized ω_s and ω_0

```
N = length(x);
ws = 2*pi/N;
wnorm = -pi:ws:pi;
wnorm = wnorm(1:length(x));
w = wnorm*fs;

X= fft(x);

plot(w,abs(fftshift(X)));
axis([-30,30,0,160]);
```

B Ninness, Spectral Analysis using the FFT

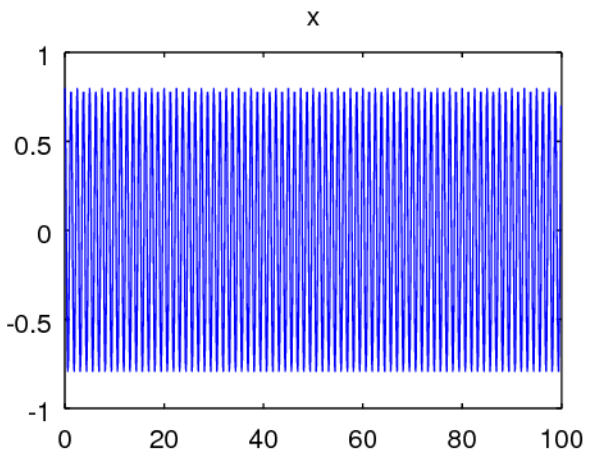
Normalized ω_s and ω_0

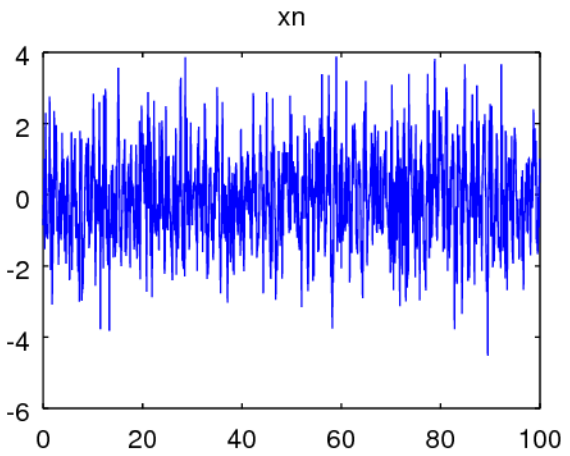
```
win = hamming(N);

xw = win(:) .* x(:);

Xw = fft(xw);

plot(w, abs(fftshift(Xw)));
axis([-10, 10, 0, 80]);
```

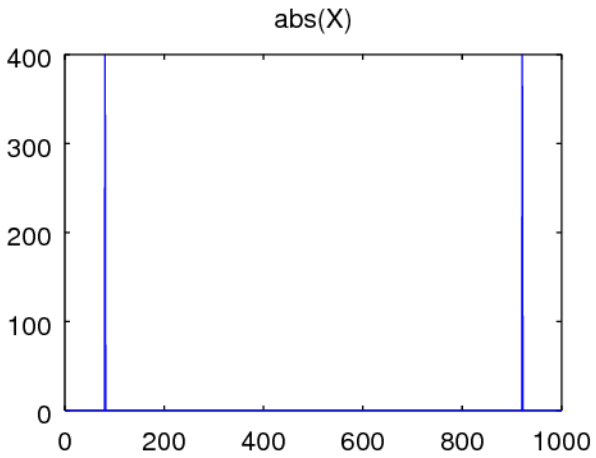
B Ninness, Spectral Analysis using the FFT

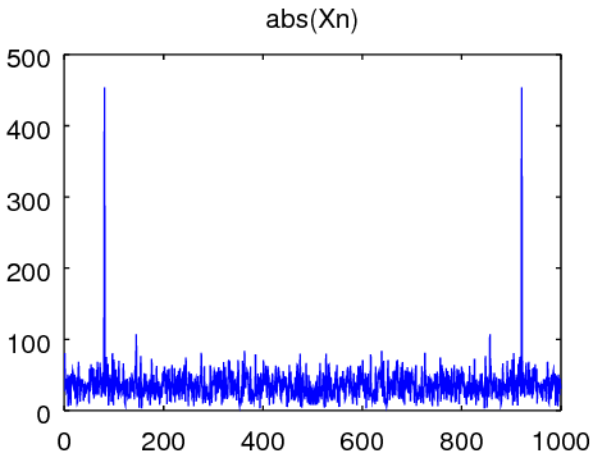
Normalized ω_s and ω_0

N=1000









B Ninness, Spectral Analysis using the FFT

FFT Example Code (1)

```
pkg load communications

N = 2000;

fs = 10;
t = (0:1:N-1)/fs;

x = 0.8*cos(2*pi*0.8*t);
xn = awgn(x, -2);

X  = fft(x);
Xn = fft(xn);

display('[1] x, xn, |X|, |Xn| : freq index');
figure(1, 'name', 'Fig.1 x, xn, |X|, |Xn| : freq index');
subplot(2, 2, 1); plot(t, x);    title("x");
subplot(2, 2, 2); plot(t, xn);   title("xn");
subplot(2, 2, 3); plot(abs(X));  title("|X|");
subplot(2, 2, 4); plot(abs(Xn)); title("|Xn|");

pause

ws = 2*pi/N;
wnorm = -pi:ws:+pi;
wnorm = wnorm(1:N);
w = wnorm * fs;

Xs  = fftshift(X);
Xns = fftshift(Xn);
```

B Ninness, Spectral Analysis using the FFT

FFT Example Code (2)

```
display('[2] |Xs|, |Xn| : freq scales');
figure(1,'name','[2] |Xs|, |Xns| : freq scales');
subplot(2,2,3); plot(w, abs(Xs));   title("|Xs|");
subplot(2,2,4); plot(w, abs(Xns));  title("|Xns|");

pause

display('[3] |Xs|, |Xns|, |Xss|, |Xnss| : self-inverting');
figure(1,'name','[3] |Xs|, |Xns|, |Xss|, |Xnss| : self-inverting');
subplot(2,2,1); plot(w, abs(Xs));            title("|Xs|");
subplot(2,2,2); plot(w, abs(Xns));           title("|Xns|");
subplot(2,2,3); plot(w, abs(fftshift(Xs)));  title("|fftshift(Xs)|");
subplot(2,2,4); plot(w, abs(fftshift(Xns))); title("|fftshift(Xns)|");

pause

% close all figure windows
% must use this toolkit to use ginput()
% close all;
% graphics_toolkit fltk
%
% plot(w, abs(fftshift(Xn)));
%
% [freq,amp, buttons]= ginput()


b1 = round(  5 / 400 * N);
a1 = round(166 / 400 * N);
a2 = round( 59 / 400 * N);
a3 = N -a1 -a2 -b1 -b1;
```

B Ninness, Spectral Analysis using the FFT

FFT Example Code (3)

```
display(a1);
display(a2);
display(a3);
display(b1);
fil = [ zeros(1,a1), ones(1,b1), zeros(1,a2), ones(1,b1), zeros(1,a3)];
% fil = [ zeros(1,166), ones(1,5), zeros(1,59), ones(1,5), zeros(1,165)];
% fil = [ zeros(1,160), ones(1,15), zeros(1,50), ones(1,15), zeros(1,160)];


Xnf= fil .* fftshift(Xn);
xnf= ifft(fftshift(Xnf));

display('[4] |Xns|+fil, xnf');
figure(1, 'name', '[4] |Xns|+fil, xnf');
subplot(2, 2, 3); plot(w, abs(Xns),'b', w, 200*fil,'-r'); title("|Xns|");
subplot(2, 2, 4); plot(t, real(xnf));                 title("real(xnf)");

pause

win_col = hamming(N);
win = win_col';
disp( length(win) );

xw = win .* xn;

Xw = fft(xw);
Xws= fftshift(Xw);
Xwf= fil .* fftshift(Xw);
xwf= ifft(fftshift(Xwf));
```

B Ninness, Spectral Analysis using the FFT

FFT Example Code (4)

```
display('[5] win, xw, |Xws|+fil, xwf/win');
figure(1,'name','[5] win, xw, |Xws|+fil, xwf/win');
subplot(2, 2, 1); plot(t, win);                              title("win");
subplot(2, 2, 2); plot(t, xw);                               title("xw");
subplot(2, 2, 3); plot(w, abs(Xws),'b', w, 200*fil,'-r'); title("|Xws|");
subplot(2, 2, 4); plot(t, real(xwf) ./ win );    title("real(xwf)./win");

pause

display('[6] xn, xw, |Xns|+fil, |Xws|+fil');
figure(1,'name','[6] xn, xw, |Xns|+fil, |Xws|+fil');
subplot(2, 2, 1); plot(t, xn);                               title("xn");
subplot(2, 2, 2); plot(t, xw);                               title("xw");
subplot(2, 2, 3); plot(w, abs(Xns),'b', w, 200*fil,'-r'); title("|Xns|");
subplot(2, 2, 4); plot(w, abs(Xws),'b', w, 200*fil,'-r'); title("|Xws|");

pause

display('[7] |Xnf|, |Xwf|, xnf, xwf');
figure(1,'name','[7], |Xnf|, |Xwf|, xnf, xwf');
subplot(2, 2, 1); plot(w, abs(Xnf));                         title("|Xnf|");
subplot(2, 2, 2); plot(w, abs(Xwf));                         title("|Xwf|");
subplot(2, 2, 3); plot(t, xnf);                              title("xnf");
subplot(2, 2, 4); plot(t, real(xwf)./win);      title("real(xwf)./win");

pause
```

B Ninness, Spectral Analysis using the FFT

References

[1] http://en.wikipedia.org/
[2] J.H. McClellan, et al., Signal Processing First, Pearson Prentice Hall, 2003
[3] M.J. Roberts, Fundamentals of Signals and Systems
[4] S.J. Orfanidis, Introduction to Signal Processing
[5] K. Shin, et al., Fundamentals of Signal Processing for Sound and Vibration Engineerings

[6] A “graphical interpretation” of the DFT and FFT, by Steve Mann