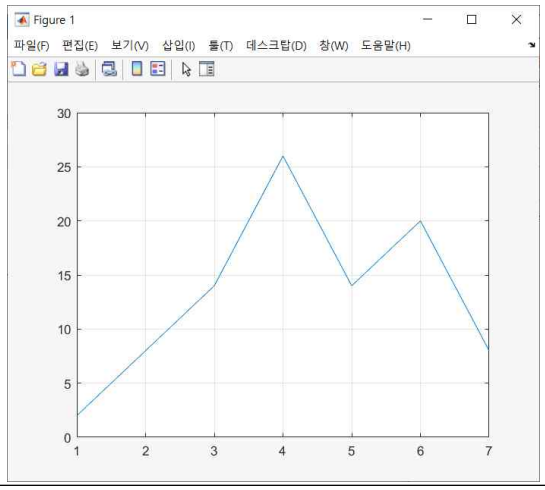
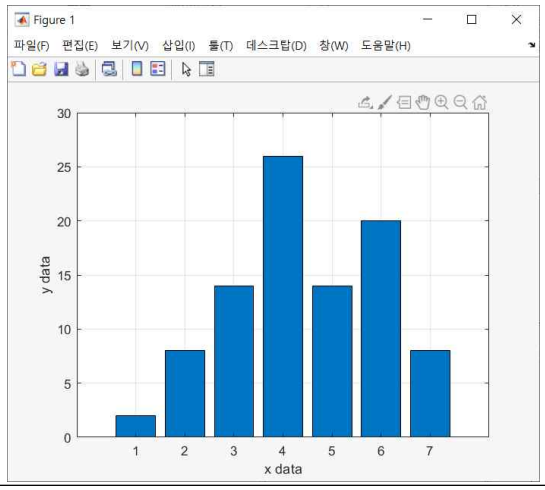
과목명	멀티미디어신호처리
담당교수	박규식 교수님
학과	소프트웨어학과
학번	32153180
이름	이상민
제출일자	2019.11.19

Simple vector addition

명령 창

```
>> a = [1 3 5 9 5 7 3];
>> b = a*3-1

b =

     2     8    14    26    14    20     8

>> plot(b)
>> grid on
>> bar(b)
>> xlabel('x data')
>> ylabel('y data')
>> grid on
>>
```



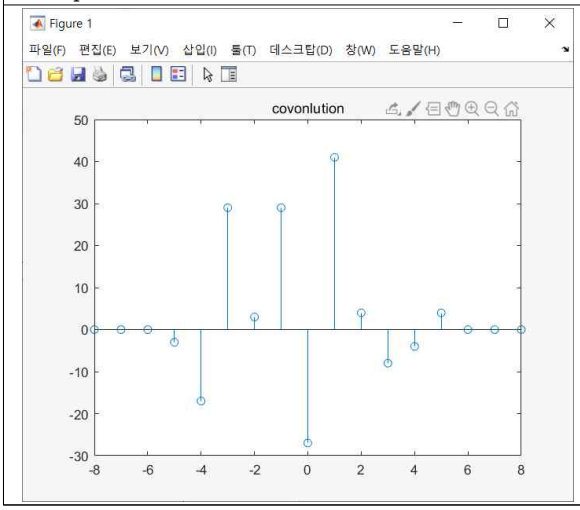


Simple matrix operation

명령 창

```
>> A = [1 3 3; 0 -5 7; 4 2 1]

A =

     1     3     3
     0    -5     7
     4     2     1

>> B = A'

B =

     1     0     4
     3    -5     2
     3     7     1

>> C = A*B

C =

    19     6    13
     6    74    -3
    13    -3    21

>> D = A.*B

D =

     1     0    12
     0    25    14
    12    14     1

>> x = inv(A)

x =

   -0.1520    0.0240    0.2880
    0.2240   -0.0880   -0.0560
    0.1600    0.0800   -0.0400

>> I = x*A

I =

    1.0000         0   -0.0000
         0    1.0000    0.0000
         0   -0.0000    1.0000
```

Sample convolution



sampconv.m

```
1 —    x = [-3, 4, 1, 5, -6, 2];
2 —    h = [1, 7, 0, 3, 4, 2];
3
4 —    y = conv(h, x);
5
6 —    z = [zeros(1, 3), y, zeros(1, 3)];
7 —    n = -8:8;
8 —    stem(n, z);
9 —    title('covonlution');
10
```

User created function

stat.m

```
1      function [mean, stdev, cal] = stat(x)
2
3      % Calculate mean and standard deviation of vector x
4
5 —    [m, n] = size(x);
6
7 —    mean = sum(x)/n;
8
9 —    stdev = sqrt(sum(x.^2)/m - mean.^2);
10
11 —   cal = sum(x.^2);
```

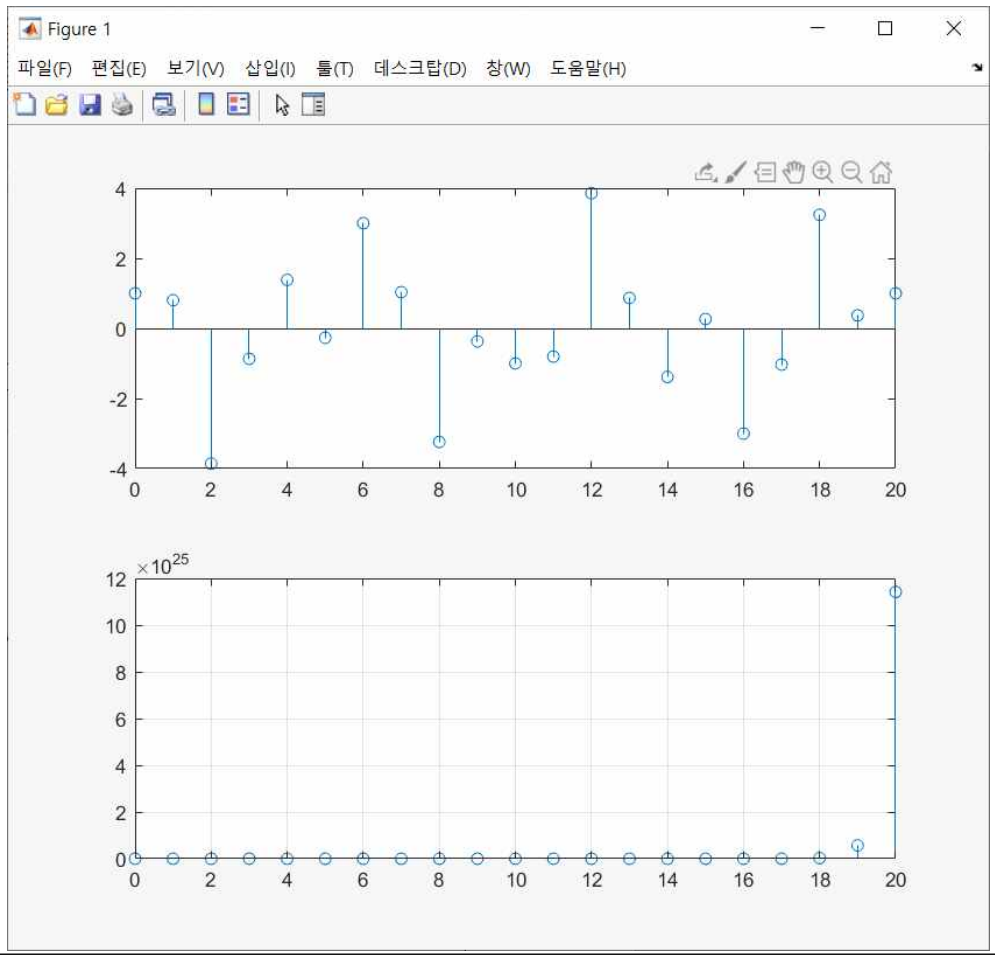
명령 창

```
>> a = [8, 1, 2, 3, 5, 6, 2];
>> [m, n, o] = stat(a)

m =

    3.8571


n =

   11.3191


o =

   143
```

Discrete signal

signal.m

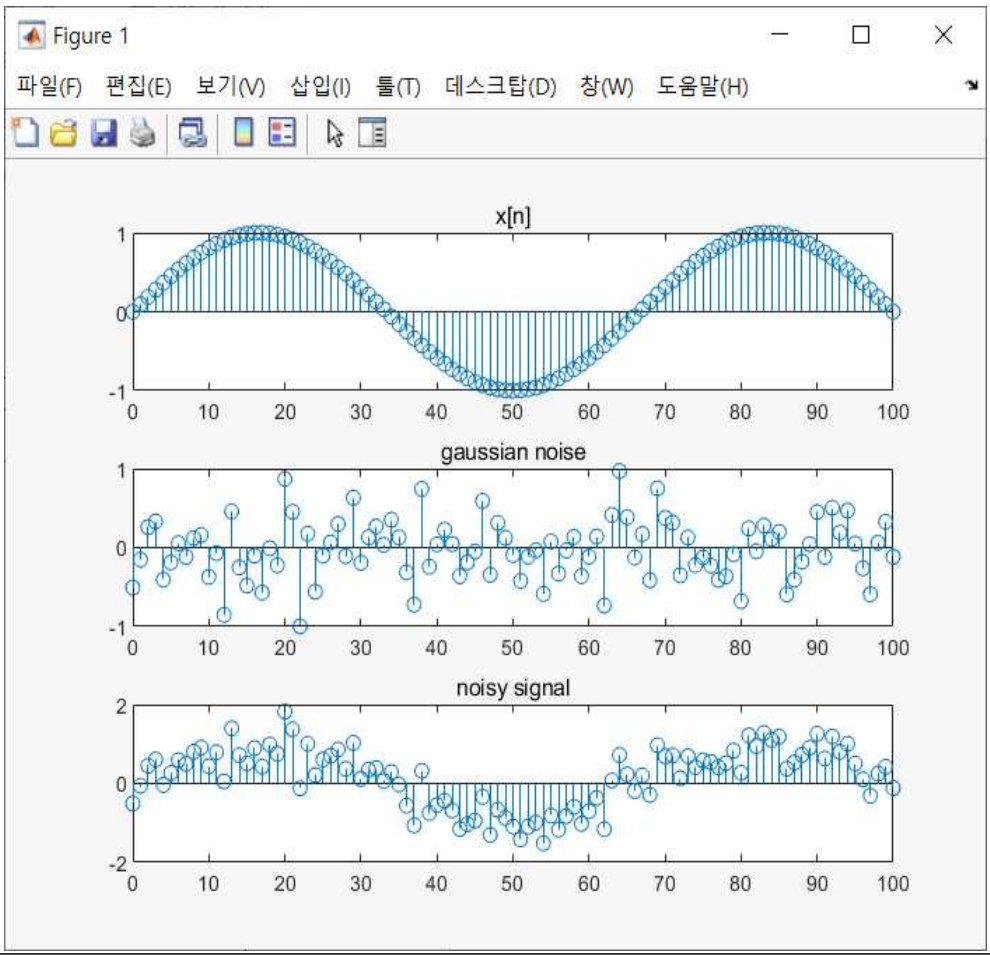
```
% sinusoidal signal
n = 0:20;
x = 2*cos(0.3*pi*n + pi/3) + 2*sin(0.7*pi*n);

% complex exponential
y = exp((3 + 2*j) * n);

figure(1)
subplot(211)
stem(n, x)
subplot(212)
stem(n, abs(y))
grid on

```



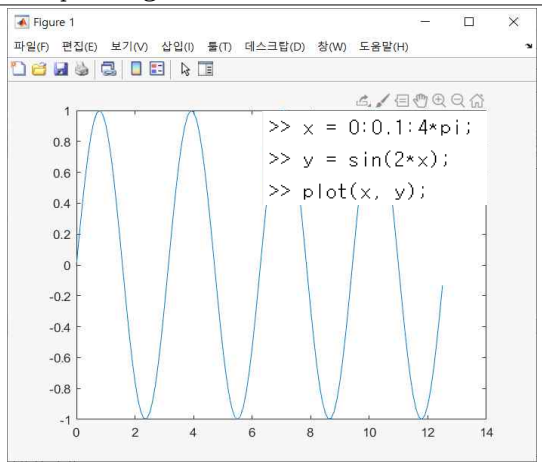
Gaussian signal

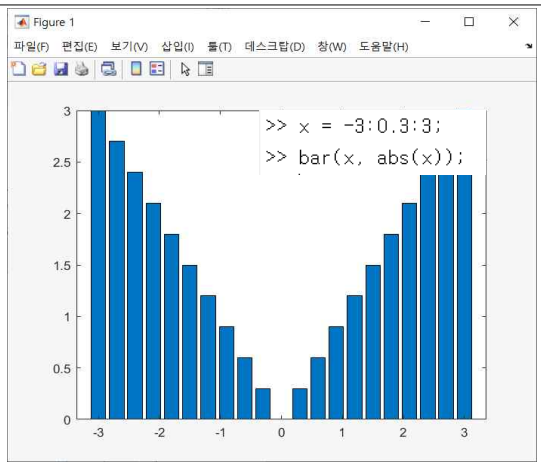
gauss_noise.m

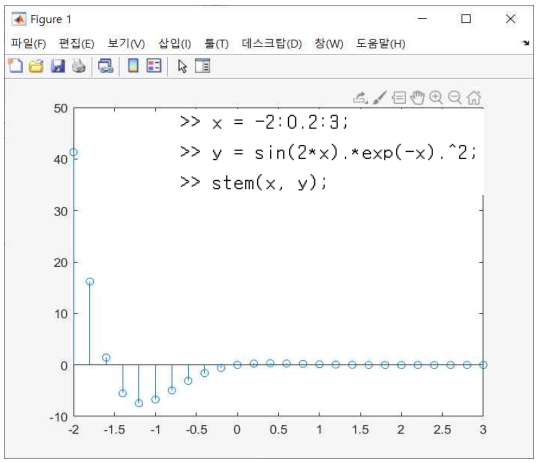
```
% signal with gaussian noise

n = 0:100;
x = sin(0.03*pi*n);
w = 0.4*randn(size(n));
y = x + w;

figure(1)
subplot(311)
stem(n, x); title('x[n]');
subplot(312)
stem(n, w); title('gaussian noise');
subplot(313)
stem(n, y); title('noisy signal');
```

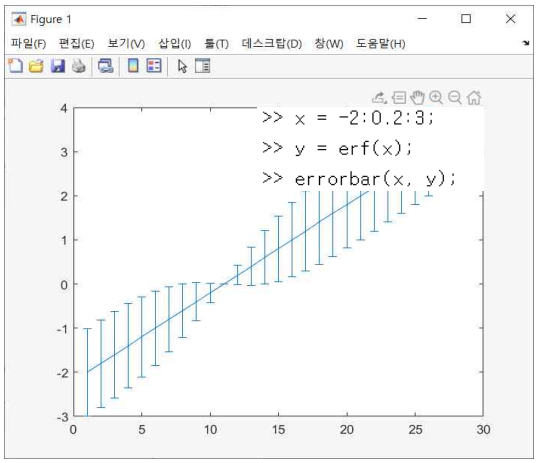


2-D plotting









Tic & Toc

sampconv.m

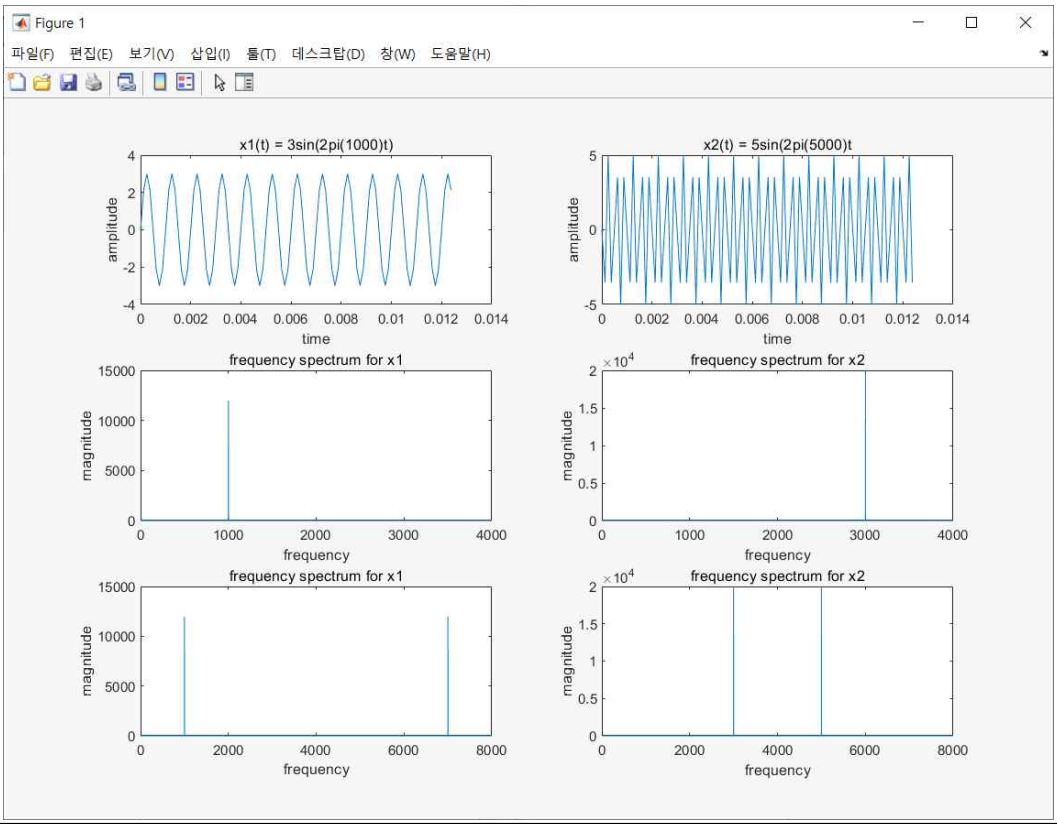
```
1 —     x = [-3, 4, 1, 5, -6, 2];
2 —     h = [1, 7, 0, 3, 4, 2];
3
4 —     tic
5 —     y = conv(h, x);
6 —     toc
7
8 —     z = [zeros(1, 3), y, zeros(1, 3)];
9 —     n = -8:8;
10 —    stem(n, z);
11 —    title('covonlution');
12
```

```
>> sampconv
경과 시간은 0.001714초입니다.
```

Spectrum example

spectrum_ex.m

```
1     % Sampling frequnecy Fs = 8000Hz
2     % Generate two sinusoid with f 1000Hz, and 5000Hz
3
4     % digital conversion with sampling frequency
5 —   fs = 8000;
6 —   t = 0:1/fs:1;
7 —   x1 = 3*sin(2*pi*1000*t); x2 = 5*sin(2*pi*5000*t);
8
9     % Frequency spectrum
10 —  y1 = abs(fft(x1, fs)); y2 = abs(fft(x2, fs));
11
12    % time-domain plot
13 —  figure(1)
14 —  subplot(321);
15 —  plot(t(1:100), x1(1:100));
16 —  xlabel('time'); ylabel('amplitude');
17 —  title('x1(t) = 3sin(2pi(1000)t)');
18
19 —  subplot(322);
20 —  plot(t(1:100), x2(1:100));
21 —  xlabel('time'); ylabel('amplitude');
22 —  title('x2(t) = 5sin(2pi(5000)t');
23
24    % frequency spectrum plot
25 —  fi = 1:1:fs/2;
26 —  subplot(323);
27 —  plot(fi, y1(1:fs/2));
28 —  xlabel('frequency'); ylabel('magnitude');
29 —  title('frequency spectrum for x1');
30
31 —  subplot(324);
32 —  plot(fi, y2(1:fs/2));
33 —  xlabel('frequency'); ylabel('magnitude');
34 —  title('frequency spectrum for x2');
35
36    % if fi = 1:1:fs
37 —  fi = 1:1:fs;
38 —  subplot(325);
39 —  plot(fi, y1(1:fs));
40 —  xlabel('frequency'); ylabel('magnitude');
41 —  title('frequency spectrum for x1');
42
43 —  subplot(326);
44 —  plot(fi, y2(1:fs));
45 —  xlabel('frequency'); ylabel('magnitude');
46 —  title('frequency spectrum for x2');
```



Soundsync

soundsyn.m

```
% Sound synthesis for 1 octave

fs = 8000;
t = 0:1/fs:0.2;
freq = [264 297 330 352 396 352 330 297 264];

% initialize the output vector
x = zeros(1, 8*fs+length(freq));

n1 = 1;
for k = 1:length(freq)
    n2 = n1 + length(t) - 1;
    x(n1:n2) = x(n1:n2) + cos(2*pi*freq(k)*t);
    n1 = n2 + 1;
end

soundsc(x, fs)      % play sound
```

Soundsync

RGBimage.m

```
% R, G, B separation
myimage = imread('C:₩Users₩sangmin₩Desktop₩ex.jpg');

% read each R, G, B as an array
redchannel = myimage(:, :, 1);
bluechannel = myimage(:, :, 2);
greenchannel = myimage(:, :, 3);

% Display
figure(1)
subplot(221);
imshow(myimage); title('original image');
subplot(222);
imshow(redchannel); title('red component');
subplot(223);
imshow(bluechannel); title('blue component');
subplot(224);
imshow(greenchannel); title('green component');
```