Matlab Practice 05

Signal Reconstruction

Sangkeun Lee

Matlab Practice 5-1

For original signal

$$x(t) = \cos(2\pi f_0 t)$$

(a) Plot the original signal with f_0=100 Hz
(b) Plot the sampled signals with f_s=[75, 100, 125, 400]
(c) Plot the spectrum of the sampled signals

Set the parameters (it is up to you) to *ti*=-0.02, *tf*=0.02, and *dt*=0.00001

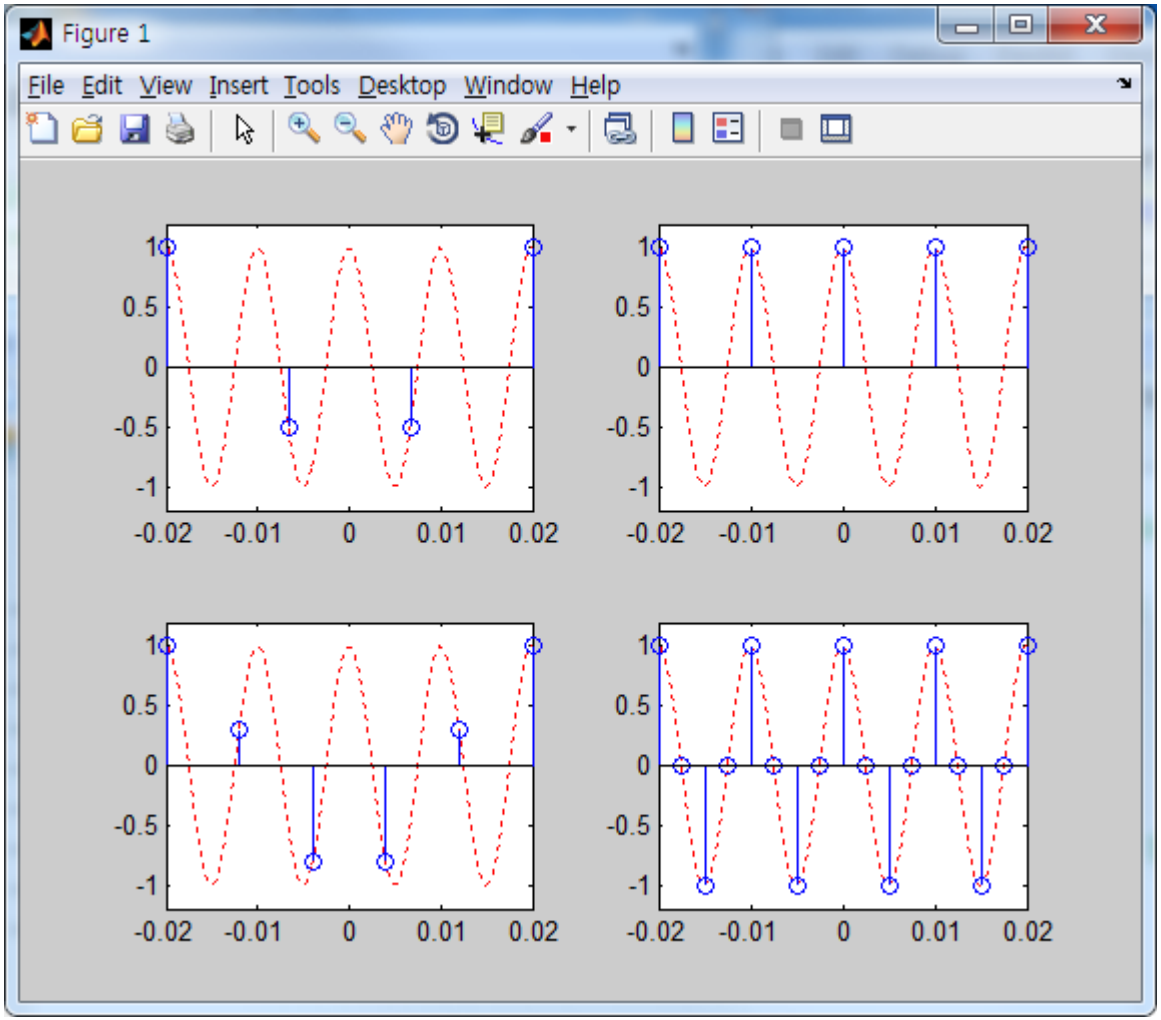


Figure 1
File Edit View Insert Tools Desktop Window Help
1
0.5
0
-0.5
-1
-0.02 -0.01 0 0.01 0.02

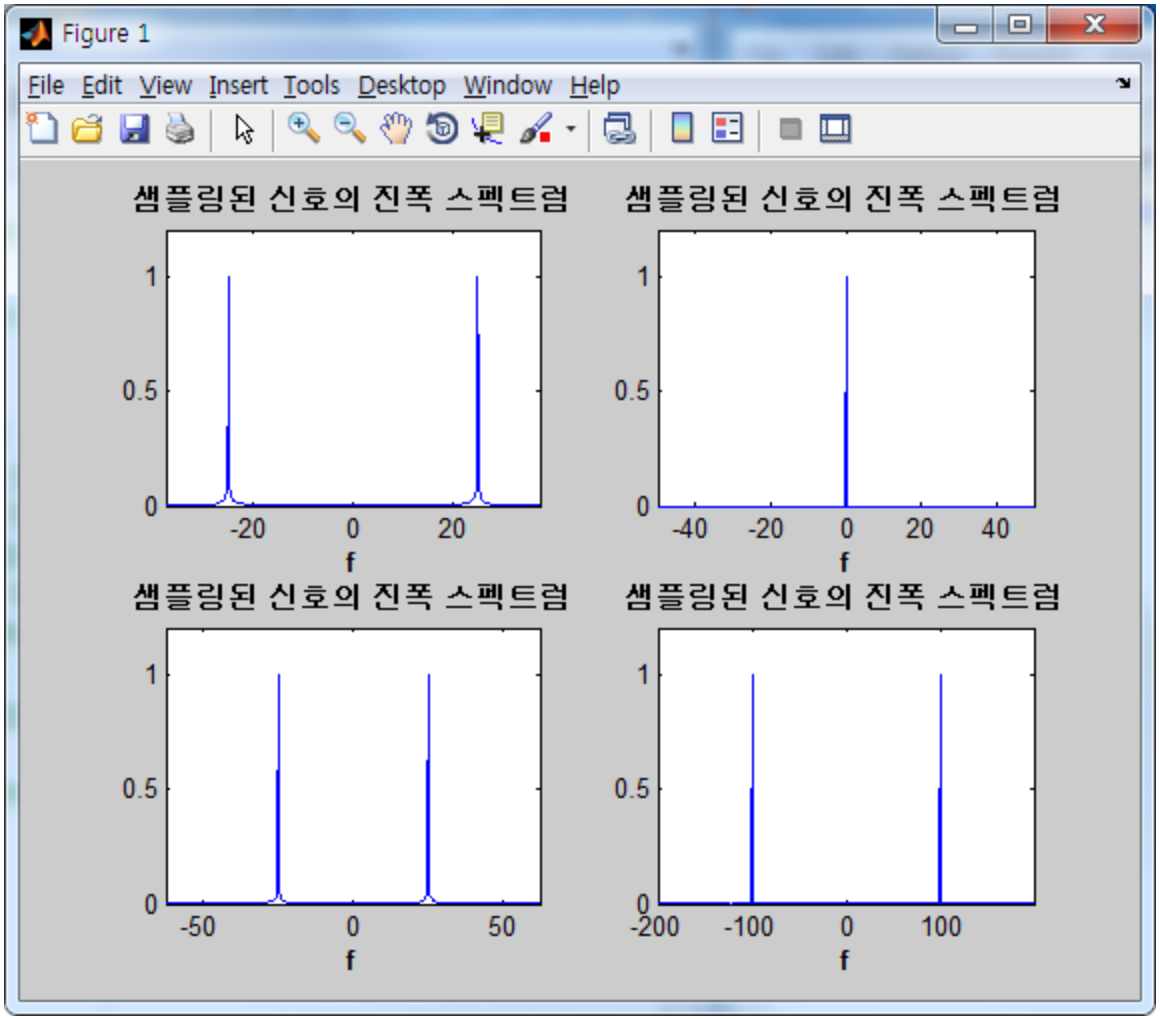
Signal Representation

```
close all;                                          % 모든 생성된 창을 닫음
clear all;                                          % 열려있는 작업영역(workspace)을 모두 비움
clc;                                                % 명령(command) 창을 비움

f0=100;                                             % 연속 정현파 x(t)의 고유 주파수 설정
ti=-0.02; tf=0.02; dt=0.00001;                      % 시간 축 설정 위한 파라미터 값 지정
fs=[75,100,125,400];                                % 샘플링 주파수 설정
sample_signal_1(ti,tf,dt,fs(1),1,f0,2,2,1);         % 샘플링 주파수 75[Hz]인 이산 정현파 그림
sample_signal_1(ti,tf,dt,fs(2),1,f0,2,2,2);         % 샘플링 주파수 100[Hz]인 이산 정현파 그림
sample_signal_1(ti,tf,dt,fs(3),1,f0,2,2,3);         % 샘플링 주파수 125[Hz]인 이산 정현파 그림
sample_signal_1(ti,tf,dt,fs(4),1,f0,2,2,4);         % 샘플링 주파수 400[Hz]인 이산 정현파 그림
```

Function sample_signal_1

```
function sample_signal_1(ti,tf,dt,fs,A,f0,rs,cs,r)
                                        % 샘플링 주기에 따른 이산 정현파를 생성해서 그림


t=ti:dt:tf;                             % t를 ti부터 tf까지 dt 간격으로 증가시킴
xc=A*cos(2*pi*f0*t);                    % 연속 정현파 생성
n=ti:1/fs:tf;                           % t를 Ts 간격으로 샘플링하여 n을 생성
xd=A*cos(2*pi*f0*n);                    % 이산 정현파 생성
subplot(rs,cs,r);                       % rs행 cs열 분할 그림 창의 r번 창
plot(t,xc,':r');                        % 연속 정현파 포락선을 그림
axis([ti tf -(1.2*A) 1.2*A]);           % x축(ti~tf)과 y축(-1.2A~1.2A)의 그림 영역을 설정
hold on;                                % 같은 그림 창에 계속 그림
stem(n,xd);                             % 샘플링 주기가 Ts인 이산 정현파 그림
```

Spectrum Representation using Fourier Transform



Required sub-function for spectrum analysis

```
function plot_spectrum(fs,N,w)
                                                    % 샘플링된 신호의 진폭 스펙트럼 그림

f0=100;                                             % 연속 정현파 신호 x(t)의 주파수
Ts=1/fs;                                            % 샘플링 주기 계산
df=fs/N;                                            % 주파수 해상도 계산
n=0:Ts:(N-1)*Ts;                                    % 샘플 추출 시간 설정
x=cos(2*pi*f0*n);                                   % 샘플링된 신호 x[n] 생성
X=fft(x);                                           % 고속 푸리에 변환에 의한 스펙트럼 계산
X=fftshift(X);                                      % 스펙트럼을 좌우 대칭이 되게 정렬
f=(-N/2)*df:df:(N/2-1)*df;                          % 그림을 그리기 위한 주파수 축 설정
Xmag=abs(X)/max(abs(X));                            % (규준화된) 진폭 스펙트럼 계산

subplot(2,2,w);                                     % 2행 2열 분할 그림 창의 w번 창
plot(f,Xmag);                                       % 진폭 스펙트럼 그림
axis([f(1) f(length(f)) 0 1.2]);                    % x축 및 y축 그림 영역 설정
title('₩bf{샘플링된 신호의 진폭 스펙트럼}');          % 그림 제목
xlabel('₩bf{f}');                                   % x축 라벨 표시
```

Code for spectrum representation

```
close all;                          % 모든 생성된 창을 닫음
clear all;                          % 열려있는 작업영역(workspace)을 모두 비움
clc;                                % 명령(command) 창을 비움

fs=[75,100,125,400];                % 샘플링 주파수 설정
N=1024;                             % 데이터 개수 설정
plot_spectrum(fs(1),N,1);           % fs=75인 샘플링된 신호의 진폭 스펙트럼 그림
plot_spectrum(fs(2),N,2);           % fs=100인 샘플링된 신호의 진폭 스펙트럼 그림
plot_spectrum(fs(3),N,3);           % fs=125인 샘플링된 신호의 진폭 스펙트럼 그림
plot_spectrum(fs(4),N,4);           % fs=400인 샘플링된 신호의 진폭 스펙트럼 그림
```

Signal Reconstruction with zero-hold, and first order hold

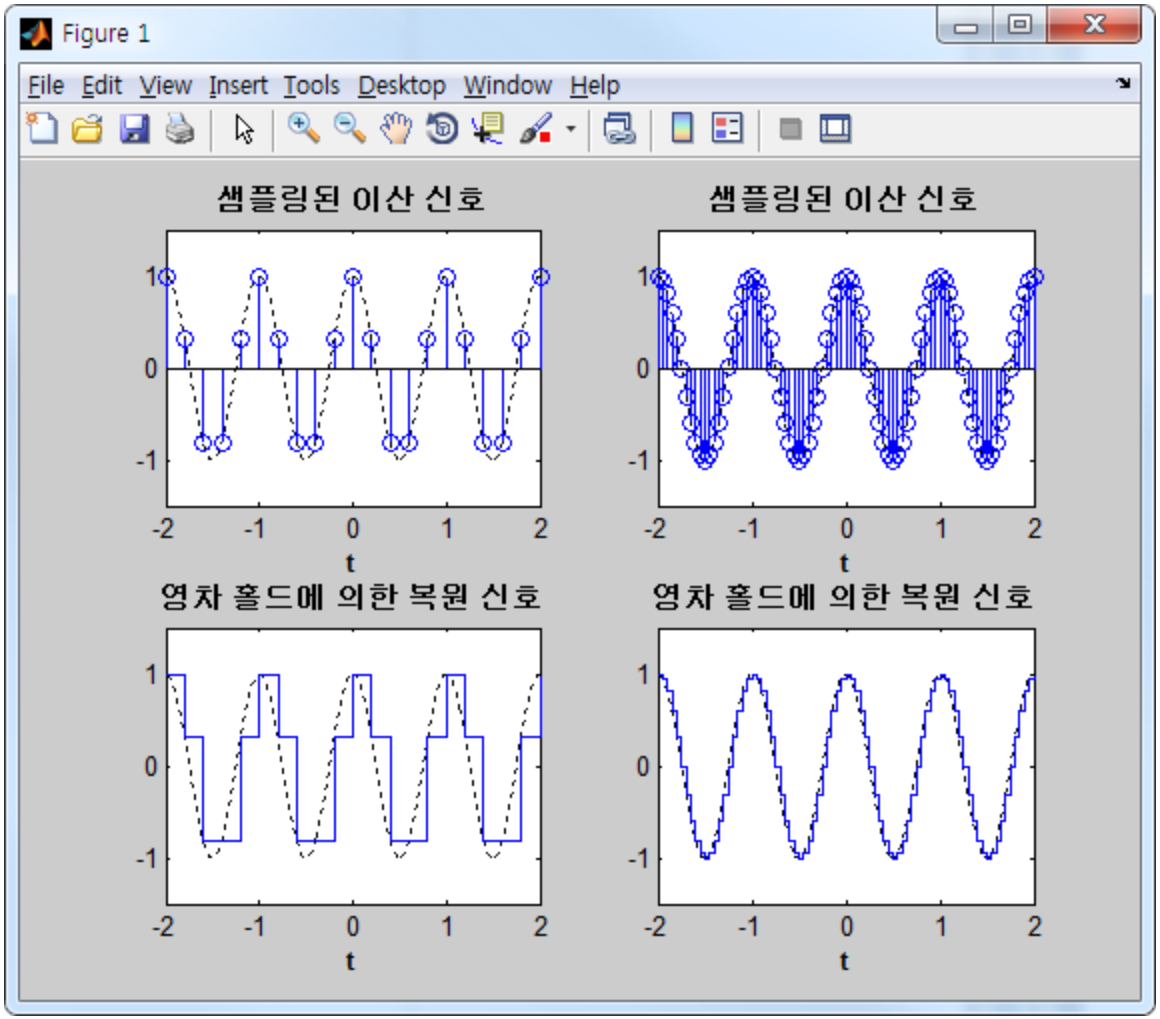
- **Matlab Practice 5-2**

For original signal

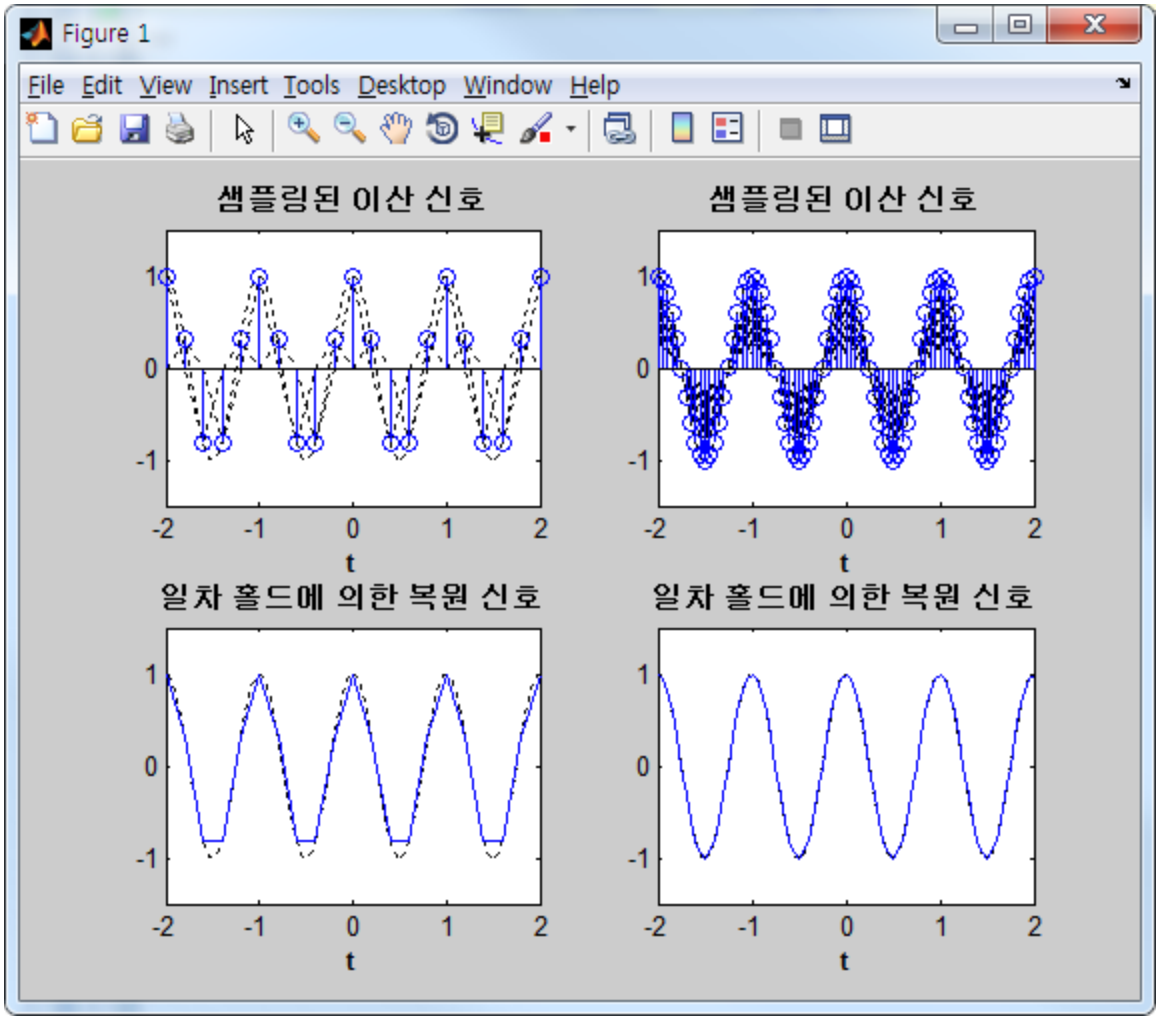
$$x(t) = \cos(2\pi f_0 t)$$

(a) Plot the original signal with f_0=1 Hz
(b) Plot the sampled signals with f_s=[5, 10]
(c) Reconstruct the sampled signals using zero- and first order hold approaches.

Display the results from zero-hold reconstruction



Display the results from first order hold reconstruction



Code for zero-hold reconsruction

```
close all;                                  % 모든 생성된 창을 닫음
clear all;                                  % 열려있는 작업영역(workspace)을 모두 비움
clc;                                        % 명령(command) 창을 비움

fs=[5,20];                                  % 샘플링 주파수 설정
ti=-2; tf=2; dt=0.001;                      % 시간 파라미터 설정
t=ti:dt:tf;                                 % 시간축 설정
xt=cos(2*pi*t);                             % 원래의 연속 신호 x(t) 생성
zero_hold(fs(1),ti,tf,t,xt,[1 3])           % fs=5일 때의 x[n] 및 xr(t) 그림
zero_hold(fs(2),ti,tf,t,xt,[2 4]);          % fs=20일 때의 x[n] 및 xr(t) 그림
```

Required Function zero_hold

```
function zero_hold(fs, ti, tf, t, xt, w)
                                        % 영차 홀드를 이용한 샘플링된 신호의 복원

n=ti:1/fs:tf;                           % 샘플링 주파수에 따른 샘플 개수 설정
xn=cos(2*pi*n);                         % 샘플링된 신호 x[n] 계산

subplot(2,2,w(1));                      % 2행 2열 분할 그림창의 w(1)번 창 지정
plot(t,xt,'k:')                         % 원래의 연속 신호 x(t) 그림
axis([ti tf -1.5 1.5])                  % x축과 y축의 그림 영역 설정
hold on                                 % 현재 창 유지해 연속해서 그림
stem(n,xn)                              % 샘플링된 이산 신호 x[n] 그림
title('₩bf{샘플링된 이산 신호}');       % 그림 제목
xlabel('₩bf{t}');                       % x축 라벨 표시
subplot(2,2,w(2));                      % 2행 2열 분할 그림 창의 w(2)번 창 지정
plot(t,xt,'k:')                         % 원래의 연속 신호 x(t) 그림
axis([ti tf -1.5 1.5])                  % x축과 y축의 그림 영역 설정
hold on                                 % 현재 창 유지해 연속해서 그림
stairs(n,xn)                            % 영차 홀드에 의한 복원 신호 xr(t) 그림
title('₩bf{영차 홀드에 의한 복원 신호}');  % 그림 제목
xlabel('₩bf{t}');                       % x축 라벨 표시
```

Code for first order hold reconstruction

```
close all;                                  % 모든 생성된 창을 닫음
clear all;                                  % 열려있는 작업영역(workspace)을 모두 비움
clc;                                        % 명령(command) 창을 비움

fs=[5,20];                                  % 샘플링 주파수 설정
ti=-2; tf=2; dt=0.001;                      % 시간 파라미터 설정
t=ti:dt:tf;                                 % 시간축 설정
xt=cos(2*pi*t);                             % 원래의 연속 신호  생성
first_hold(fs(1),ti,tf,t,xt,[1 3])          % fs=5일 때의 x[n] 및 xr(t) 그림
first_hold(fs(2),ti,tf,t,xt,[2 4]);         % fs=20일 때의 x[n] 및 xr(t) 그림
```

Required function: first_hold

```
function first_hold(fs, ti, tf, t, xt, w)
                                        % 영차 홀드를 이용한 샘플링된 신호의 복원

n=ti:1/fs:tf;                           % 샘플링 주파수에 따른 샘플 개수 설정
xn=cos(2*pi*n);                         % 샘플링된 신호 x[n] 계산
nf=length(xn);                          % 배열 데이터의 길이 파악
x1=zeros(1,nf); x2=zeros(1,nf);         % 배열 데이터 초기화(0으로 둠)
for m=1:2:nf
    x1(m)=xn(m);                        % 각 샘플의 일차 홀드 파형 표시용 데이터
end
for m=2:2:nf
   x2(m)=xn(m);                         % 각 샘플의 일차 홀드 파형 표시용 데이터
end

subplot(2,2,w(1));                      % 2행 2열 분할 그림창의 w(1)번 창 지정
plot(t,xt,'k:')                         % 원래의 연속 신호 x(t) 그림
axis([ti tf -1.5 1.5])                  % x축과 y축의 그림 영역 설정
hold on                                 % 현재 창 유지해 연속해서 그림
plot(n,x1,'k:')                         % 각 샘플에서의 일차 홀드 파형 표시(왼쪽)
plot(n,x2,'k:')                         % 각 샘플에서의 일차 홀드 파형 표시(오른쪽)
stem(n,xn)                              % 샘플링된 이산 신호 x[n] 그림
title('₩bf{샘플링된 이산 신호}');       % 그림 제목
xlabel('₩bf{t}');                       % x축 라벨 표시
subplot(2,2,w(2));                      % 2행 2열 분할 그림창의 w(2)번 창 지정
plot(t,xt,'k:')                         % 원래의 연속 신호 x(t) 그림
axis([ti tf -1.5 1.5])                  % x축과 y축의 그림 영역 지정
hold on                                 % 현재 창 유지해 연속해서 그림
plot(n,xn)                              % 일차 홀드에 의한 복원 신호 xr(t) 그림
title('₩bf{일차 홀드에 의한 복원 신호}');  % 그림 제목
xlabel('₩bf{t}');                       % x축 라벨 표시
```