CHAPTER 2

Discrete-Time Signals and Systems

Tutorial Problems

1. (a) MATLAB script:

```
% P0201a: Generate and plot unit sample
close all; clc
n = -20:40; % specifiy support of signal
deltan = zeros(1,length(n)); % define signal
deltan(n==0)=1;
% Plot:
hf = figconfg('P0201a','small');
stem(n,deltan,'fill')
axis([min(n)-1,max(n)+1,min(deltan)-0.2,max(deltan)+0.2])
xlabel('n','fontsize',LFS); ylabel('\delta[n]','fontsize',LFS);
title('Unit Sample \delta[n]','fontsize',TFS)
```

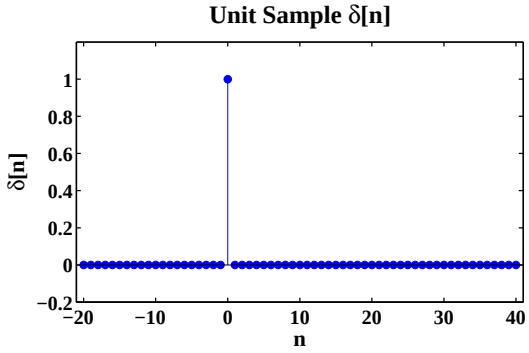


FIGURE 2.1: unit sample $\delta[n]$.

(b) MATLAB script:

```
% P0201b: Generate and plot unit step sequence
close all; clc
n = -20:40; % specifiy support of signal
un = zeros(1,length(n)); % define signal
un(n>=0)=1;
% Plot:
hf = figconfg('P0201b','small');
stem(n,un,'fill')
axis([min(n)-1,max(n)+1,min(un)-0.2,max(un)+0.2])
xlabel('n','fontsize',LFS); ylabel('u[n]','fontsize',LFS);
title('Unit Step u[n]','fontsize',TFS)
```

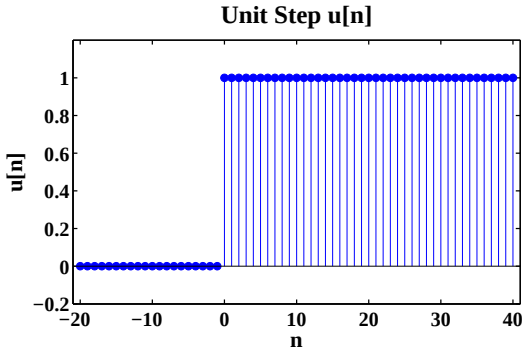


FIGURE 2.2: unit step $u[n]$.

(c) MATLAB script:

```
% P0201c: Generate and plot real exponential sequence
close all; clc
n = -20:40; % specifiy support of signal
x1n = 0.8.^n; % define signal
% Plot:
hf = figconfg('P0201c','small');
stem(n,x1n,'fill')
axis([min(n)-1,max(n)+1,min(x1n)-5,max(x1n)+5])
xlabel('n','fontsize',LFS); ylabel('x_1[n]','fontsize',LFS);
title('Real Exponential Sequence x_1[n]','fontsize',TFS)
```

(d) MATLAB script:

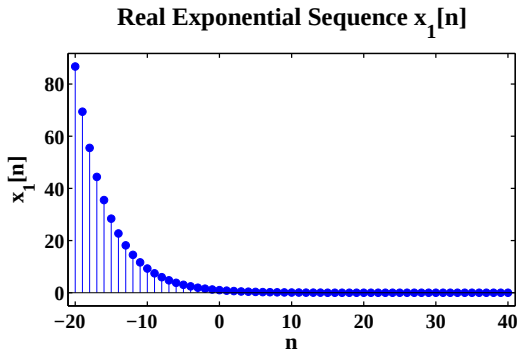


FIGURE 2.3: real exponential signal $x_1[n] = (0.80)^n$.

```
% P0201d: Generate and plot complex exponential sequence
close all; clc
n = -20:40; % specifiy support of signal
x2n = (0.9*exp(j*pi/10)).^n; % define signal
x2n_r = real(x2n); % real part
x2n_i = imag(x2n); % imaginary part
x2n_m = abs(x2n); % magnitude part
x2n_p = angle(x2n); % phase part
% Plot:
hf = figconfg('P0201d');
subplot(2,2,1)
stem(n,x2n_r,'fill')
axis([min(n)-1,max(n)+1,min(x2n_r)-1,max(x2n_r)+1])
xlabel('n','fontsize',LFS); ylabel('Re\{x_2[n]\}','fontsize',LFS);
title('Real Part of Sequence x_2[n]','fontsize',TFS)
subplot(2,2,2)
stem(n,x2n_i,'fill')
axis([min(n)-1,max(n)+1,min(x2n_i)-1,max(x2n_i)+1])
xlabel('n','fontsize',LFS); ylabel('Im\{x_2[n]\}','fontsize',LFS);
title('Imaginary Part of Sequence x_2[n]','fontsize',TFS)
subplot(2,2,3)
stem(n,x2n_m,'fill')
axis([min(n)-1,max(n)+1,min(x2n_m)-1,max(x2n_m)+1])
xlabel('n','fontsize',LFS); ylabel('|x_2[n]|','fontsize',LFS);
title('Magnitude of Sequence x_2[n]','fontsize',TFS)
subplot(2,2,4)
```

```
stem(n,x2n_p,'fill')
axis([min(n)-1,max(n)+1,min(x2n_p)-1,max(x2n_p)+1])
xlabel('n','fontsize',LFS); ylabel('\phi(x_2[n])','fontsize',LFS);
title('Phase of Sequence x_2[n]','fontsize',TFS)
```

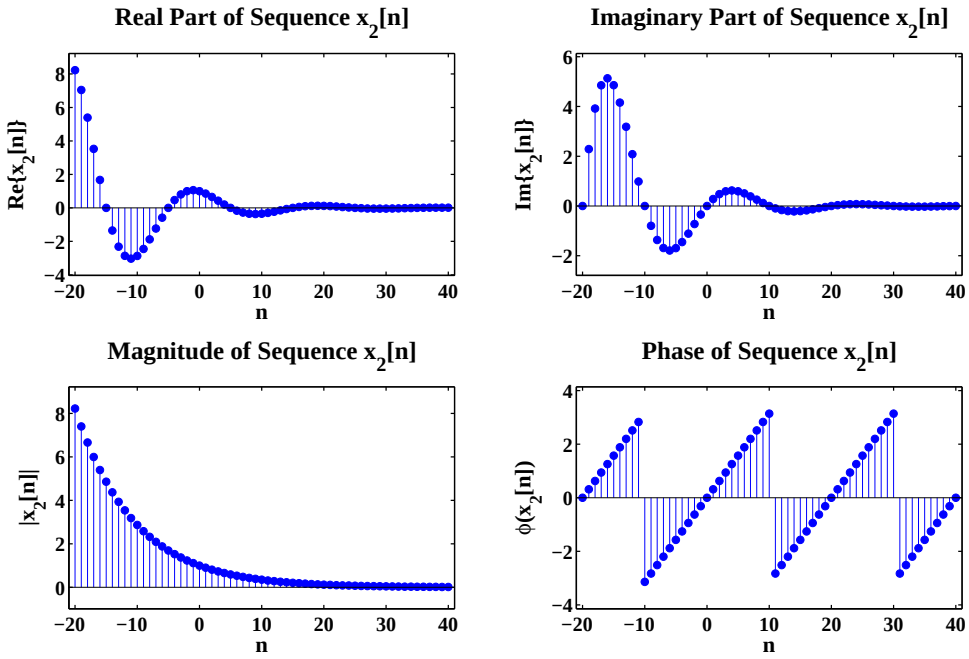


FIGURE 2.4: complex exponential signal $x_2[n] = (0.9\mathrm{e}^{\mathrm{j}\pi/10})^n$.

(e) MATLAB script:

```
% P0201e: Generate and plot real sinusoidal sequence
close all; clc
n = -20:40; % specifiy support of signal
x3n = 2*cos(2*pi*0.3*n+pi/3); % define signal
% Plot:
hf = figconfg('P0201e','small');
stem(n,x3n,'fill')
axis([min(n)-1,max(n)+1,min(x3n)-0.5,max(x3n)+0.5])
xlabel('n','fontsize',LFS); ylabel('x_3[n]','fontsize',LFS);
title('Real Sinusoidal Sequence x_3[n]','fontsize',TFS)
```

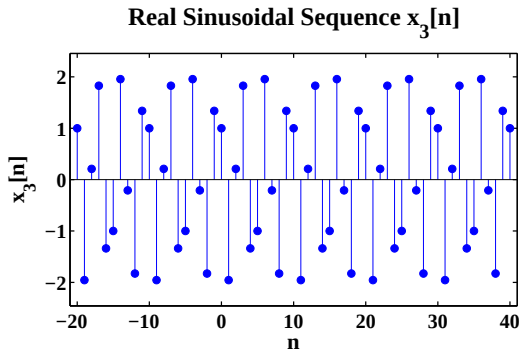


FIGURE 2.5: sinusoidal sequence $x_3[n] = 2\cos[2\pi(0.3)n + \pi/3]$.

2. MATLAB script:

```
% P0202: Illustrate the noncommutativity of folding and shifting
close all; clc
nx = 0:4; % specify the support
x = 5:-1:1; % specify sequence
n0 = 2;
% (a) First folding, then shifting
[y1 ny1] = fold(x,nx);
[y1 ny1] = shift(y1,ny1,-n0);
% (b) First shifting, then folding
[y2 ny2] = shift(x,nx,-n0);
[y2 ny2] = fold(y2,ny2);
% Plot
hf = figconfg('P0202');
xylimit = [min([nx(1),ny1(1),ny2(1)])-1,max([nx(end),ny1(end)...
    ,ny2(end)])+1,min(x)-1,max(x)+1];
subplot(3,1,1)
stem(nx,x,'fill')
axis(xylimit)
ylabel('x[n]','fontsize',LFS); title('x[n]','fontsize',TFS);
set(gca,'Xtick',xylimit(1):xylimit(2))
subplot(3,1,2)
stem(ny1,y1,'fill')
axis(xylimit)
ylabel('y_1[n]','fontsize',LFS);
```

```
title('y_1[n]: Folding and Shifting','fontsize',TFS)
set(gca,'Xtick',xylimit(1):xylimit(2))
subplot(3,1,3)
stem(ny2,y2,'fill')
axis(xylimit)
xlabel('n','fontsize',LFS); ylabel('y_2[n]','fontsize',LFS);
title('y_2[n]: Shifting and Folding','fontsize',TFS)
set(gca,'Xtick',xylimit(1):xylimit(2))
```

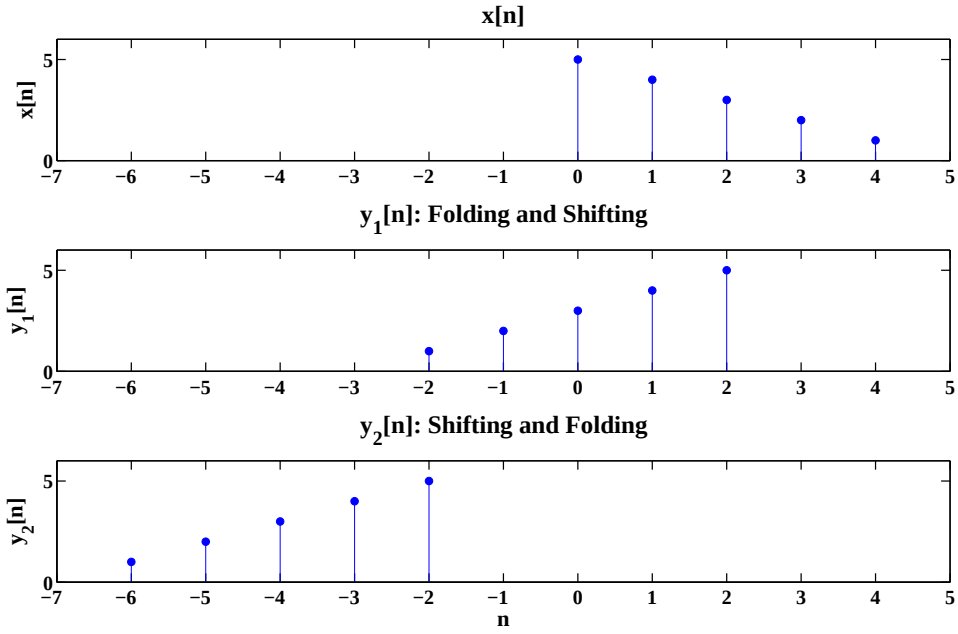


FIGURE 2.6: Illustrating noncommunativity of folding and shifting operations.

Comments:
From the plot, we can see $y_1[n]$ and $y_2[n]$ are different. Indeed, $y_1[n]$ represents the correct $x[2-n]$ signal while $y_2[n]$ represents signal $x[-n-2]$.

3. (a) $x[-n] = \{4, 4, 4, 4, \underset{\uparrow}{4}, 3, 2, 1, 0, -1\}$
$x[n-3] = \{-1, 0, \underset{\uparrow}{1}, 2, 3, 4, 4, 4, 4, 4\}$
$x[n+2] = \{-1, 0, 1, 2, 3, 4, 4, \underset{\uparrow}{4}, 4, 4\}$

(b) see part (c)

(c) MATLAB script:

```
% P0203bc: Illustrate the folding and shifting effect
close all; clc
nx = -5:4; % specify support
x = [-1:4,4*ones(1,4)]; % define sequence
[y1 ny1] = fold(x,nx); % folding
[y2 ny2] = shift(x,nx,-3); % right-shifting
[y3 ny3] = shift(x,nx,2); % left-shifting
% Plot
hf = figconfg('P0203');
xylimit = [min([nx(1),ny1(1),ny2(1),ny3(1)])-1,max([nx(end),...
    ny1(end),ny2(end),ny2(end)])+1,min(x)-1,max(x)+1];
subplot(4,1,1)
stem(nx,x,'fill'); axis(xylimit)
ylabel('x[n]','fontsize',LFS); title('x[n]','fontsize',TFS);
set(gca,'Xtick',xylimit(1):xylimit(2))
subplot(4,1,2)
stem(ny1,y1,'fill'); axis(xylimit)
ylabel('x[-n]','fontsize',LFS); title('x[-n]','fontsize',TFS)
set(gca,'Xtick',xylimit(1):xylimit(2))
subplot(4,1,3)
stem(ny2,y2,'fill'); axis(xylimit)
ylabel('x[n-3]','fontsize',LFS); title('x[n-3]','fontsize',TFS);
set(gca,'Xtick',xylimit(1):xylimit(2))
subplot(4,1,4)
stem(ny3,y3,'fill'); axis(xylimit)
xlabel('n','fontsize',LFS); ylabel('x[n+2]','fontsize',LFS);
title('x[n+2]','fontsize',TFS)
set(gca,'Xtick',xylimit(1):xylimit(2))
```

4. MATLAB script:

```
% P0204: Illustrate the using of repmat, persegen and pulstran
%        to generate periodic signal
close all; clc
n = 0:9; % specify support
x = [ones(1,4),zeros(1,6)]; % sequence 1
% x = cos(0.1*pi*n); % sequence 2
```

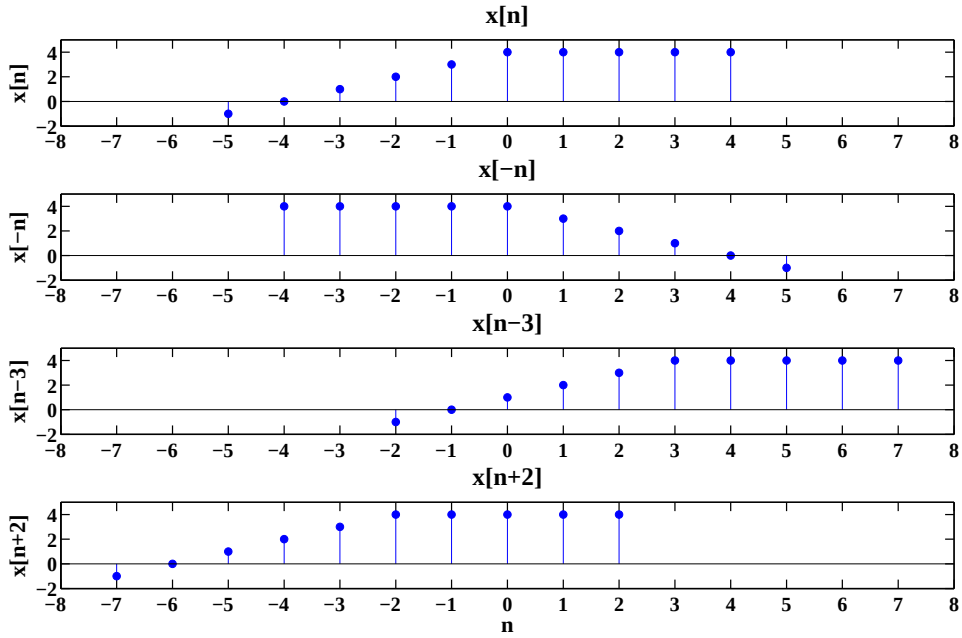


FIGURE 2.7: Illustrating folding and shifting operations.

```
% x = 0.8.^n; % sequence 3
Np = 5; % number of periods
xp1 = repmat(x,1,Np);
nxp1 = n(1):Np*length(x)-1;
[xp2 nxp2] = persegen(x,length(x),Np*length(x),n(1));
xp3 = pulstran(nxp1,(0:Np-1)'*length(x),x);
%Plot
hf = figconfg('P0204');
xylimit = [-1,nxp1(end)+1,min(x)-1,max(x)+1];
subplot(3,1,1)
stem(nxp1,xp1,'fill'); axis(xylimit)
ylabel('x_p[n]','fontsize',LFS);
title('Function ''repmat''','fontsize',TFS);
subplot(3,1,2)
stem(nxp2,xp2,'fill'); axis(xylimit)
ylabel('x_p[n]','fontsize',LFS);
```

```
title('Function ''persegen'''','fontsize',TFS)
subplot(3,1,3)
stem(nxp1,xp3,'fill'); axis(xylimit)
xlabel('n','fontsize',LFS); ylabel('x_p[n]','fontsize',LFS);
title('Function ''pulstran'''','fontsize',TFS)
```

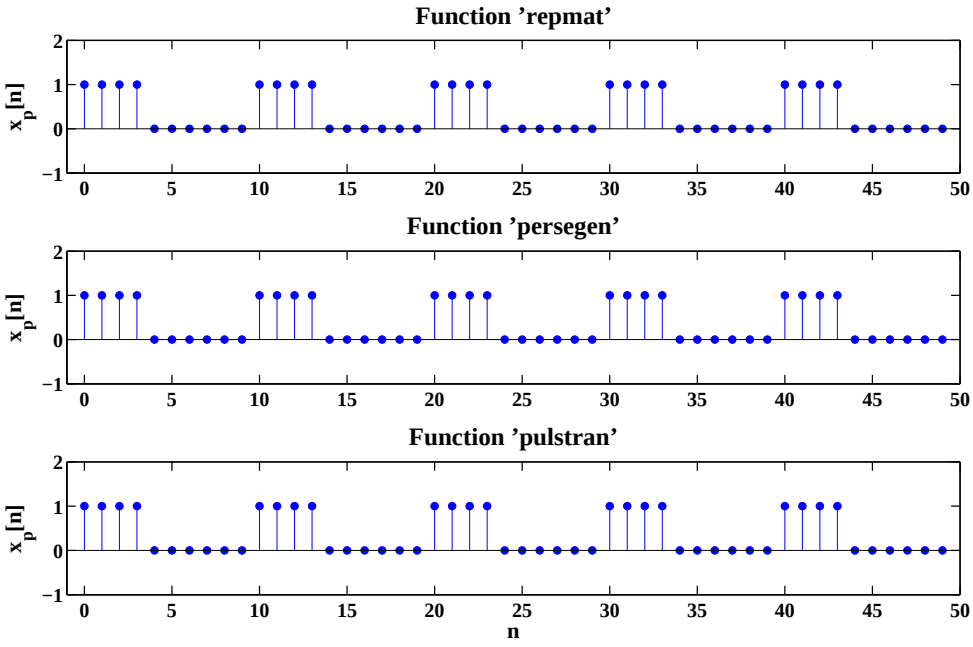


FIGURE 2.8: Periodically expanding sequence $\{1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 0\}$.

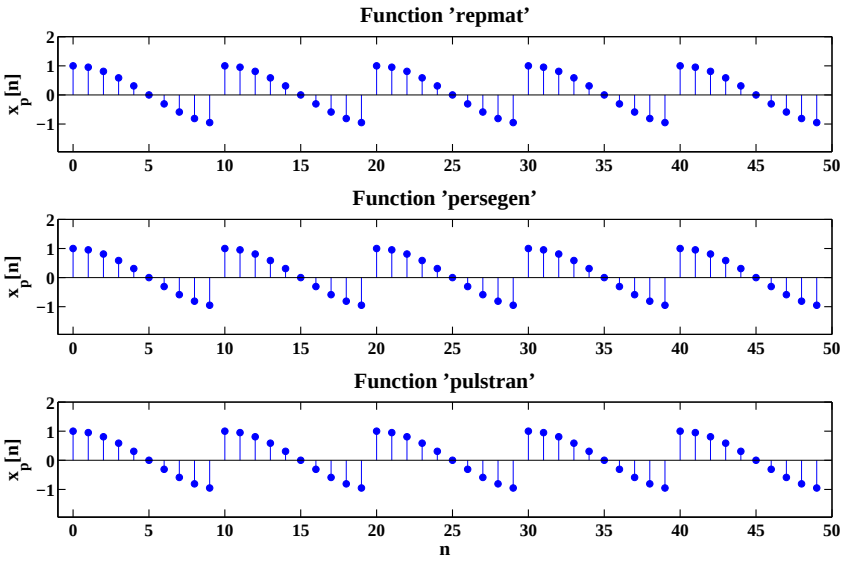


FIGURE 2.9: Periodically expanding sequence $\cos(0.1\pi n), 0 \leq n \leq 9$.

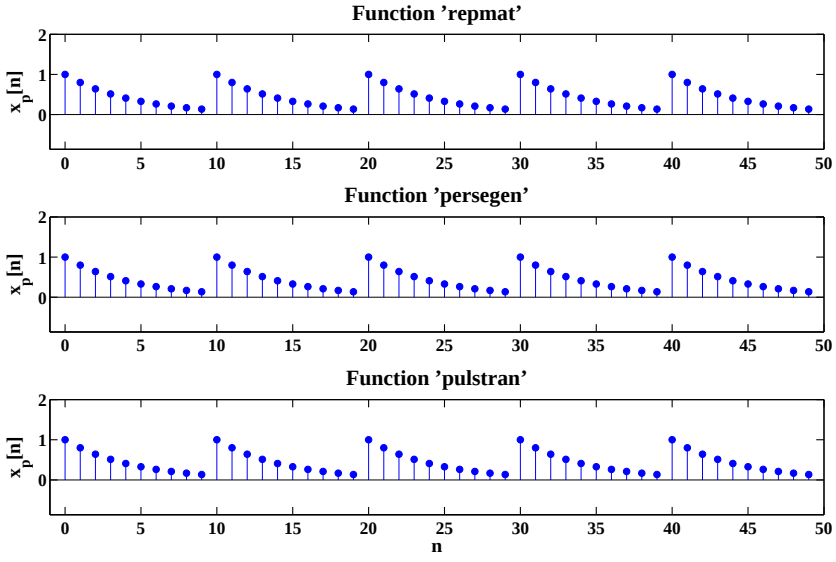


FIGURE 2.10: Periodically expanding sequence $0.8^n, 0 \leq n \leq 9$.

5. (a) Proof: If the sinusoidal signal $\cos(\omega_0 n + \theta_0)$ is periodic in n, we need to find a period N_p that satisfy $\cos(\omega_0 n + \theta_0) = \cos(\omega_0 n + \omega_0 N_p + \theta_0)$ for every n. Since $f_0 \triangleq \frac{\omega_0}{2\pi}$ is a rational number, we can substitute $\omega_0 = 2\pi f_0 = 2\pi \frac{M}{N}$ into the previous periodic condition to have $\cos(2\pi \frac{M}{N} n + \theta_0) = \cos(2\pi \frac{M}{N} n + 2\pi \frac{M}{N} N_p + \theta_0)$. No matter what integers M and N take, $N_p = N$ is a period of the sinusoidal signal.

   (b) The sequence is NOT periodic.

   (c) The sequence is periodic with fundamental period $N = 10$. N can be interpreted as period and M is the number of repetitions the corresponding continuous signal repeats itself.

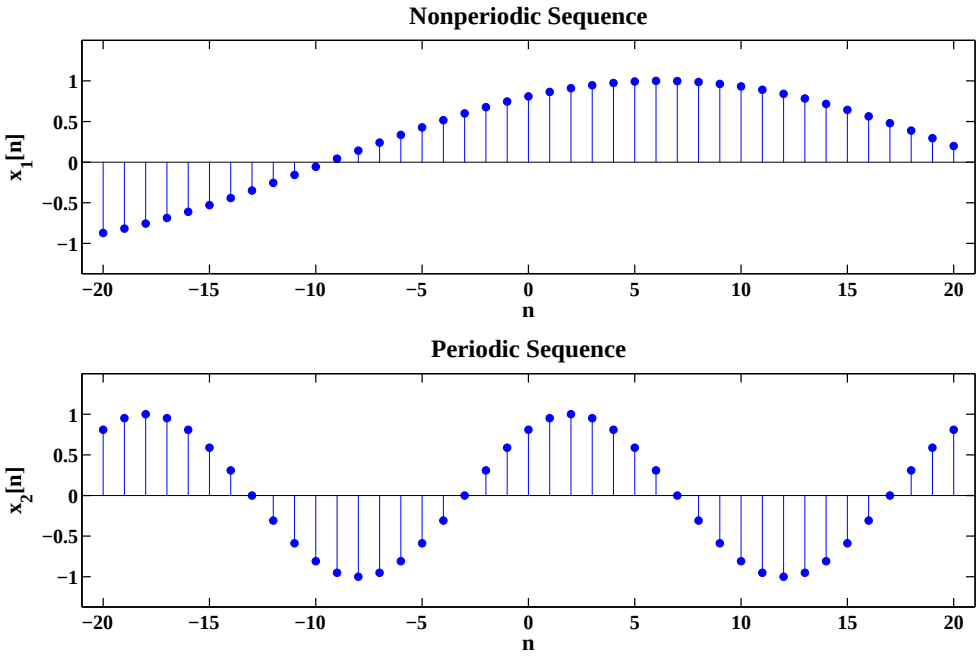


FIGURE 2.11: Illustrating the periodicity condition of sinusoidal signals.

MATLAB script:

```
% P0205: Illustrates the condition for periodicity of discrete
%               sinusoidal sequence
close all; clc
% Part (b): Nonperiodic
```

```
n = -20:20; % support
w1 = 0.1; % angular frequency
x1 = cos(w1*n-pi/5);
% Part (c): Periodic
w2 = 0.1*pi; % angular frequency
x2 = cos(w2*n-pi/5);
%Plot
hf = figconfg('P0205');
xylimit = [n(1)-1,n(end)+1,min(x1)-0.5,max(x1)+0.5];
subplot(2,1,1)
stem(n,x1,'fill'); axis(xylimit)
xlabel('n','fontsize',LFS); ylabel('x_1[n]','fontsize',LFS);
title('Nonperiodic Sequence','fontsize',TFS);
% set(gca,'Xtick',xylimit(1):xylimit(2))
subplot(2,1,2)
stem(n,x2,'fill'); axis(xylimit)
xlabel('n','fontsize',LFS); ylabel('x_2[n]','fontsize',LFS);
title('Periodic Sequence','fontsize',TFS)
```

6. MATLAB script:

```
% P0206: Investigates the effect of downsampling using
%        audio file 'handel'
close all; clc
load('handel.mat')
n = 1:length(y);
% Part (a): original sampling rate
sound(y,Fs); pause(1)
% Part (b): downsampling by a factor of two
y_ds2_ind = mod(n,2)==1;
sound(y(y_ds2_ind),Fs/2); pause(1)
% Part (c): downsampling by a factor of four
y_ds4_ind = mod(n,4)==1;
sound(y(y_ds4_ind),Fs/4)
% save the sound file
wavwrite(y(y_ds4_ind),Fs/4,'handel_ds4')
```

7. Comments: The first system is NOT time-invariant but the second system is time invariant.

   MATLAB script:

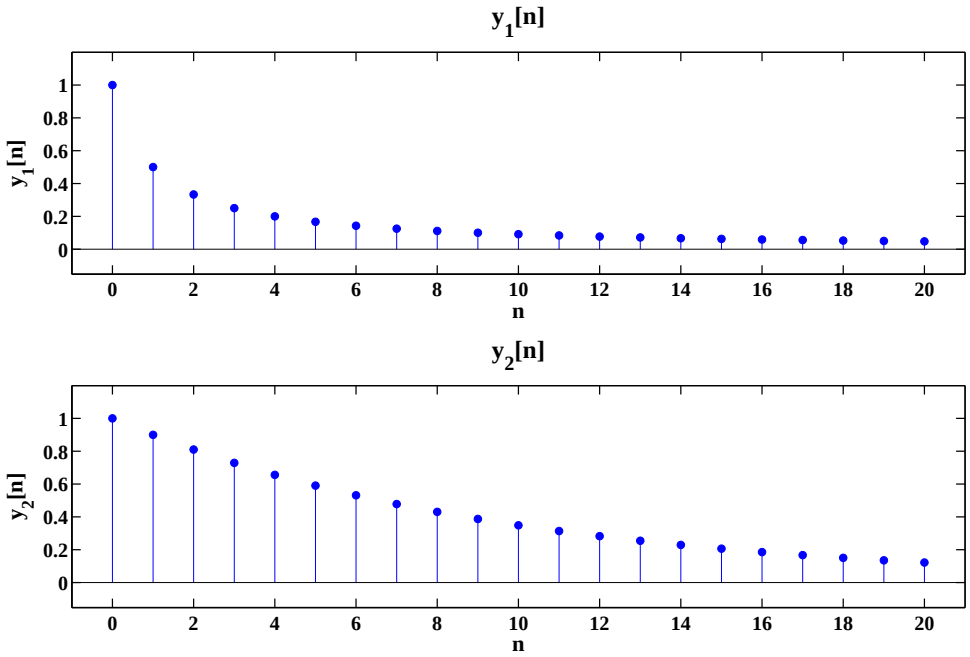


FIGURE 2.12: System responses with respect to input signal $x[n] = \delta[n]$.

```
% P0207: Compute and plot sequence defined by difference equations
close all; clc
n = 0:20; % define support
yi = 0; % zero initial condition
xn = delta(n(1),0,n(end))'; % input 1
% xn = delta(n(1),5,n(end))'; % input 2
% Compute sequence 1:
yn1 = zeros(1,length(n));
yn1(1) = n(1)/(n(1)+1)*yi+xn(1);
for ii = 2:length(n)
    yn1(ii) = n(ii)/(n(ii)+1)*yn1(ii-1)+xn(ii);
end
% Compute sequence 2:
yn2 = filter(1,[1,-0.9],xn);
%Plot
hf = figconfg('P0207');
```

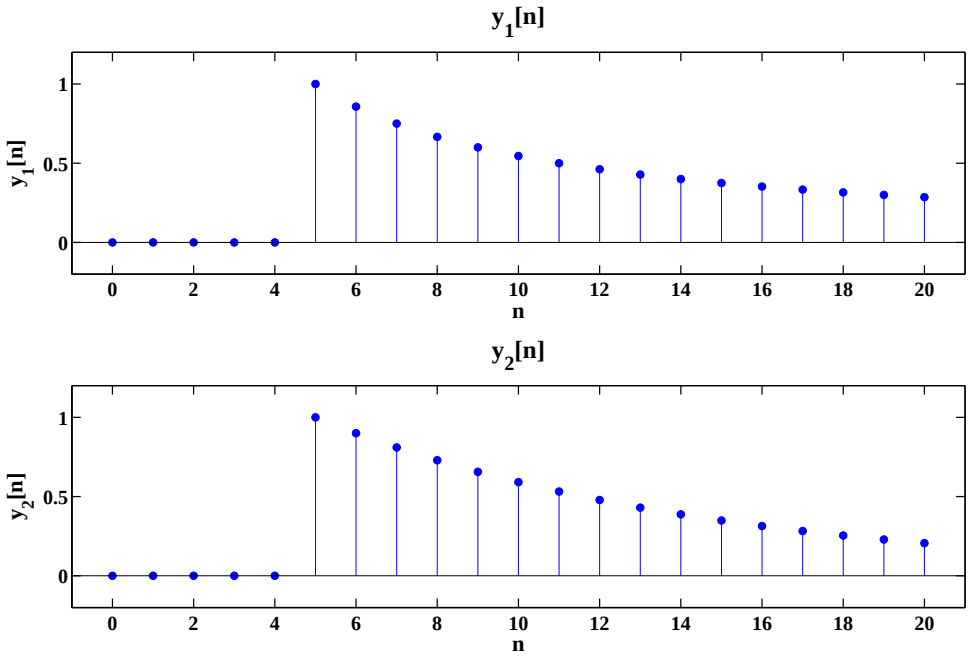


FIGURE 2.13: System responses with respect to input signal $x[n] = \delta[n-5]$.

```
xylimit = [n(1)-1,n(end)+1,min(yn1)-0.2,max(yn1)+0.2];
subplot(2,1,1)
stem(n,yn1,'fill'); axis(xylimit)
xlabel('n','fontsize',LFS); ylabel('y_1[n]','fontsize',LFS);
title('y_1[n]','fontsize',TFS);
subplot(2,1,2)
stem(n,yn2,'fill'); axis(xylimit)
xlabel('n','fontsize',LFS); ylabel('y_2[n]','fontsize',LFS);
title('y_2[n]','fontsize',TFS)
```

8. (a)

$$y[n] = \frac{1}{5}\left(x[n] + x[n-1] + x[n-2] + x[n-3] + x[n-4]\right)$$

(b)

$$h[n] = u[n] - u[n-5]$$

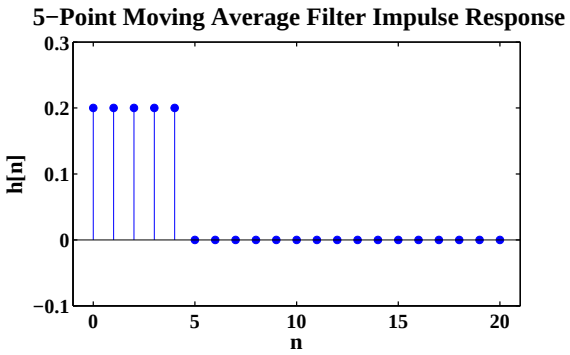


FIGURE 2.14: Impulse response of a 5-point moving average filter.

(c) Block diagram.

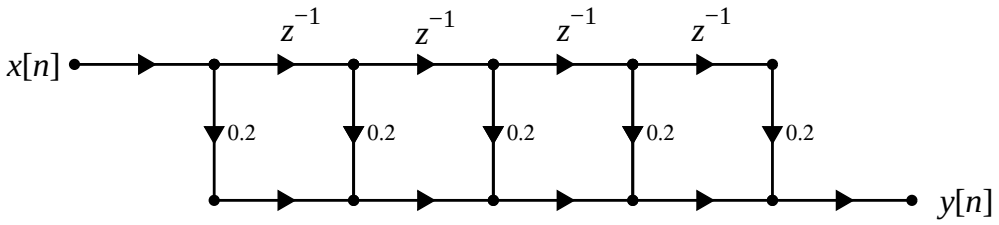


FIGURE 2.15: Block diagram of a 5-point moving average filter.

MATLAB script:

```
% P0208: Plot the 5-point moving average filter
%       y[n] = 1/5*(x[n]+x[n-1]+x[n-2]+x[n-3]+x[n-4]);
close all; clc
n = 0:20;
xn = delta(n(1),0,n(end))';
hn = filter(ones(1,5)/5,1,xn);
%Plot
hf = figconfg('P0208','small');
xylimit = [n(1)-1,n(end)+1,min(hn)-0.1,max(hn)+0.1];
stem(n,hn,'fill'); axis(xylimit)
```

```
xlabel('n','fontsize',LFS); ylabel('h[n]','fontsize',LFS);
title('5-Point Moving Average Filter Impulse Response',...
    'fontsize',TFS);
```

9. (a) Proof:

$$\sum_{n=0}^{\infty} a^n = 1 + a + a^2 + \cdots$$

$$a\sum_{n=0}^{\infty} a^n = a + a^2 + a^3 + \cdots$$

$$(1-a)\sum_{n=0}^{\infty} a^n = 1 + (a-a) + (a^2 - a^2) + \cdots + (a^\infty - a^\infty)$$

$$(1-a)\sum_{n=0}^{\infty} a^n = 1 + 0 + 0 + \cdots + 0$$

$$\sum_{n=0}^{\infty} a^n = \frac{1}{1-a}$$

(b) Proof:

$$\sum_{n=0}^{N-1} a^n = \sum_{n=0}^{\infty} a^n - \sum_{n=N}^{\infty} a^n = \sum_{n=0}^{\infty} a^n - a^N \sum_{n=0}^{\infty} a^n$$

Substituting the result in part (a), we have

$$\sum_{n=0}^{N-1} a^n = (1-a^N)\sum_{n=0}^{\infty} a^n = \frac{1-a^N}{1-a}$$

10. (a) Solution:

$$x[-m] = \{-1, 2, 3, \underset{\uparrow}{1}\}$$

$$x[3-m] = \{\underset{\uparrow}{-1}, 2, 3, , 1\}$$

$$h[m] = \{\underset{\uparrow}{2}, 2(0.8)^1, 2(0.8)^2, 2(0.8)^3, 2(0.8)^4, 2(0.8)^5, 2(0.8)^6\}$$

$$x[3-m] * h[m] = \{\underset{\uparrow}{-2}, 4(0.8)^1, 6(0.8)^2, 2(0.8)^3\}$$

$$y[3] = \sum_{m=0}^{3} x[3-m] * h[m] = 6.064$$

(b) MATLAB script:

```
% P0210: Graphically illustrate the convolution sum
close all; clc
nx = 0:3;
x = [1,3,2,-1]; % input sequence
nh = 0:6;
h = 2*(0.8).^nh; % impulse response
nxf = fliplr(-nx); xf = fliplr(x); %folding
nxfs = nxf+3; % left shifting
[y1 y2 n] = timealign(xf,nxfs,h,nh);
y = y1.*y2;
y3 = sum(y);
%Plot
hf = figconfg('P0210');
subplot(5,1,1)
stem(nx,x,'fill')
axis([-4 7 min(x)-1 max(x)+1])
ylabel('x[k]','fontsize',LFS);
subplot(5,1,2)
stem(nh,h,'fill')
axis([-4 7 min(h)-1 max(h)+1])
ylabel('h[k]','fontsize',LFS);
subplot(5,1,3)
stem(nxf,xf,'fill')
axis([-4 7 min(x)-1 max(x)+1])
ylabel('x[-k]','fontsize',LFS);
subplot(5,1,4)
stem(nxfs,xf,'fill')
axis([-4 7 min(x)-1 max(x)+1])
ylabel('x[-k+3]','fontsize',LFS);
subplot(5,1,5)
stem(n,y,'fill')
axis([-4 7 min(y)-1 max(y)+1])
xlabel('k','fontsize',LFS);
ylabel('h[k]*x[-k+3]','fontsize',LFS);
```

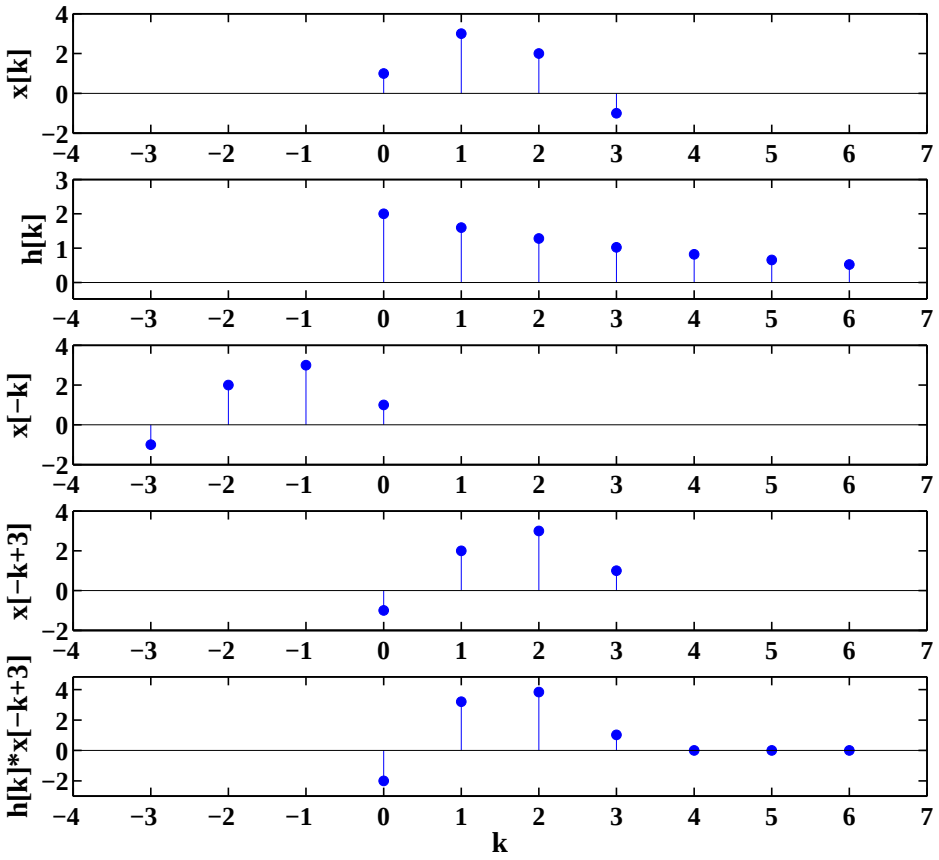


FIGURE 2.16: Graphically illustration of convolution as a superposition of scaled and scaled replicas.

11. Comments: The step responses of the two equivalent system representations are equal.

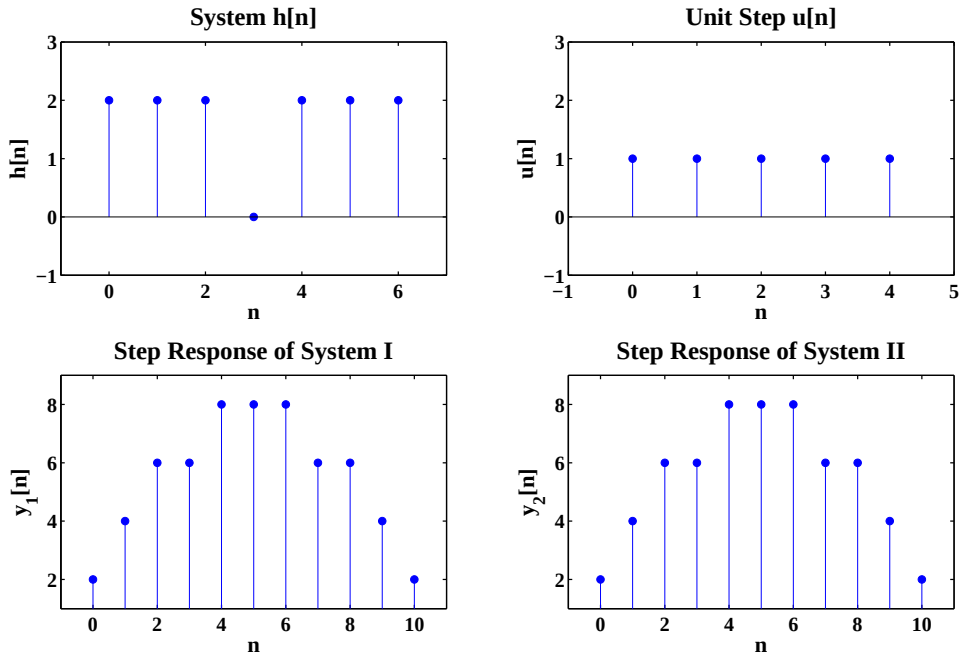


FIGURE 2.17: Illustrating equivalent system representation.

MATLAB script:

```
% P0211: Illustrating the combination of parallel and
%        series systems
close all; clc
n1 = 0:4;
h1 = ones(1,5);
h2 = [1 -1 -1 -1 1];
n2 = 0:2;
h3 = ones(1,3);
[h n] = conv0(h1+h2,n1,h3,n2);
un = unitstep(n1(1),0,n1(end));
[ytemp1 nyt] = conv0(h1,n1,un,n1);
ytemp2 = conv(h2,un);
```

```
[y1 ny1] = conv0(h3,n2,ytemp1+ytemp2,nyt);
[y2 ny2] = conv0(h,n,un,n1);
%Plot
hf = figconfg('P0211');
subplot(2,2,1)
stem(n,h,'fill')
axis([n(1)-1 n(end)+1 min(h)-1 max(h)+1])
xlabel('n','fontsize',LFS);
ylabel('h[n]','fontsize',LFS);
title('System h[n]','fontsize',TFS);
subplot(2,2,2)
stem(n1,un,'fill')
axis([n1(1)-1 n1(end)+1 min(h)-1 max(h)+1])
xlabel('n','fontsize',LFS);
ylabel('u[n]','fontsize',LFS);
title('Unit Step u[n]','fontsize',TFS);
subplot(2,2,3)
stem(ny1,y1,'fill')
axis([ny1(1)-1 ny1(end)+1 min(y1)-1 max(y1)+1])
xlabel('n','fontsize',LFS);
ylabel('y_1[n]','fontsize',LFS);
title('Step Response of System I','fontsize',TFS);
subplot(2,2,4)
stem(ny2,y2,'fill')
axis([ny1(1)-1 ny1(end)+1 min(y2)-1 max(y2)+1])
xlabel('n','fontsize',LFS); ylabel('y_2[n]','fontsize',LFS);
title('Step Response of System II','fontsize',TFS);
```

12. MATLAB script:

```
% P0212: Illustrating the usage of function 'convmtx'
close all; clc
nx = 0:5; nh = 0:3;
x = ones(1,6); h = 0.^(0:3);

A = convmtx(x,length(h));
y = h*A; % compute convolution
ny = (nx(1)+nh(1)):(nx(end)+nh(end)); % compute support
% [y2 ny2] = conv0(x,nx,h,nh);
%Plot
```

```
hf = figconfg('P0212','small');
stem(ny,y,'fill')
axis([ny(1)-1 ny(end)+1 min(y)-1 max(y)+1])
xlabel('n','fontsize',LFS); ylabel('y[n]','fontsize',LFS);
title('Convolution y[n]','fontsize',TFS);
set(gca,'XTick',ny(1):ny(end))
```

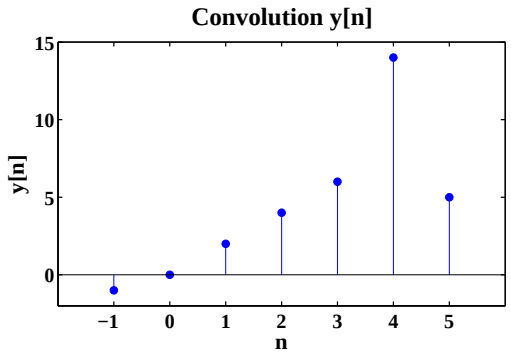


FIGURE 2.18: Compute the convolution of the finite length sequences in (2.38) using `convmtx`.

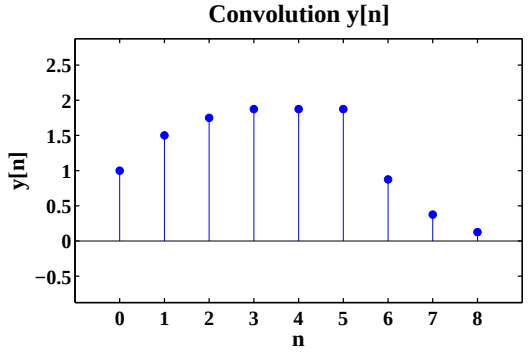


FIGURE 2.19: Compute the convolution of the finite length sequences in (2.39) using `convmtx`.

13. Proof:
Since the linear time-invariant system is stable, we have

$$\sum_{n=-\infty}^{\infty} |h[n]| < \infty$$

$$\sum_{n=-\infty}^{\infty} |h[n]| = \lim_{N\to\infty} \left(\sum_{n=-\infty}^{N} |h[n]| + \sum_{n=N+1}^{\infty} |h[n]| \right)$$

$$\lim_{N\to\infty} \left(\sum_{n=N+1}^{\infty} |h[n]| \right) = 0$$

$$y[n] = x[n] * h[n] = \sum_{m=-\infty}^{\infty} h[m]x[n-m] = \sum_{m=n-n_0}^{\infty} h[m]x[n-m]$$

$$\lim_{n\to\infty} |y[n]| = \lim_{n\to\infty} | \sum_{m=n-n_0}^{\infty} h[m]x[n-m]| \leq \lim_{n\to\infty} \sum_{m=n-n_0}^{\infty} |h[m]||x[n-m]| = 0$$

Hence, we proved

$$\lim_{n\to\infty} y[n] = 0$$

14. MATLAB script:

```
% P0214: Use function 'conv(h,x)' to compute noncausal
%        h convolves causal x
close all; clc
nh = -4:4;
nx = 0:5;
h = ones(1,9);
x = 1:6;
y1 = conv(h,x); % compute convolution
ny1 = (nh(1)+nx(1)):(nh(end)+nx(end)); % define support
[y2 ny2] = conv0(h,nh,x,nx); % verification
```

15. Comments: The image is blurred by both filters and the larger the filter is the more blurred the image is.

    MATLAB script:

```
% P0215: Filtering 2D image lena.jpg using 2D filter
close all; clc
x = imread('lena.jpg');
% Part (a): image show
hfs = figconfg('P0215a','small');
imshow(x,[])
% Part (b):
hmn = ones(3,3)/9;
y1 = filter2(hmn,x);
% hmn is symmetric and no change if rotated by 180 degrees
% we can use 2d correlation instead of 2d convolution
hfs1 = figconfg('P0215b','small');
imshow(y1,[])
% Part (c):
hmn2 = ones(5,5)/25;
y2 = filter2(hmn2,x);
hfs2 = figconfg('P0215c','small');
imshow(y2,[])
```

(a)

(b)

(c)

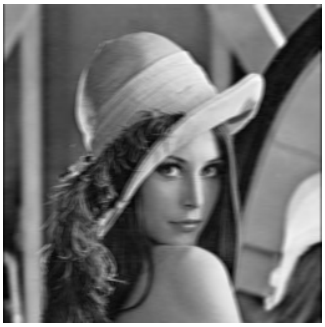
FIGURE 2.20: (a) Original image. (b) Output image processed by 3×3 impulse response $h[m, n]$ given in (2.75). (c) Output image processed by 5×5 impulse response $h[m, n]$ defined in part (c).

16. (a) See plots.
    (b) Comments: The resulting image is horizontally blurred.
    (c) Comments: The resulting image is vertically blurred.
    (d) Comments: The resulting image is blurred the same way as the one in part (c) in Problem 16.

MATLAB script:

```
% P0216: Filtering 2D image lena.jpg using 1D filter
x = imread('lena.jpg');
[nx ny] = size(x);
% Part (a): image show
hfs = figconfg('0216a','small');
imshow(x,[])
n = -2:2;
h = ones(1,5)/5;
% Part (b): horizontal filtering
yh = zeros(nx,ny);
for ii = 1:ny
    temp = conv(h,double(x(ii,:)));
    yh(ii,:) = temp(3:end-2);
end
hfs1 = figconfg('0216b','small');
imshow(yh,[])
% Part (c): vertical filtering
yv = zeros(nx,ny);
for ii = 1:nx
    temp = conv(h,double(x(:,ii)));
    yv(:,ii) = temp(3:end-2);
end
hfs2 = figconfg('0216c','small');
imshow(yv,[])
% Part (d): horizontal and vertical filtering
yhv = zeros(nx,ny);
for ii = 1:nx
    temp = conv(h,yh(:,ii));
    yhv(:,ii) = temp(3:end-2);
end
hfs3 = figconfg('0216d','small');
imshow(yhv,[])
```

(a)



(b)

(c)

(d)

FIGURE 2.21: (a) Original image. (b) Output image obtained by row processing. (c) Output image obtained by column processing. (d) Output image obtained by row and column processing.

17. (a) Impulse response.

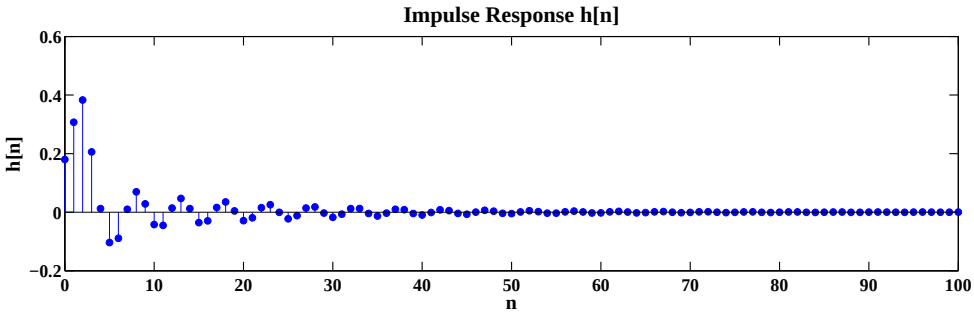


FIGURE 2.22: Impulse response $h[n]$.

(b) Output using `y=filter(b,a,x)`.

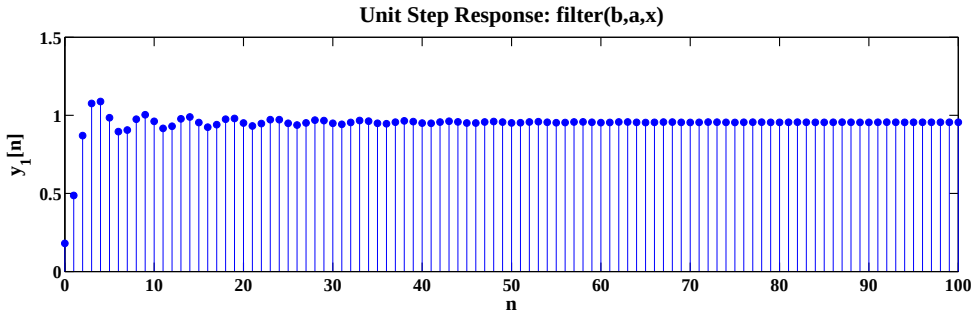


FIGURE 2.23: System step output $y[n]$ computed using the function `y=filter(b,a,x)`.

(c) Output using `y=conv(h,x)`.

(d) Output using `y=filter(h,1,x)`.

MATLAB script:

```
% P0217: Illustrating the usage of functions 'impz','filter,'conv'
close all; clc
n = 0:100;
b = [0.18 0.1 0.3 0.1 0.18];
a = [1 -1.15 1.5 -0.7 0.25];
% Part (a):
```

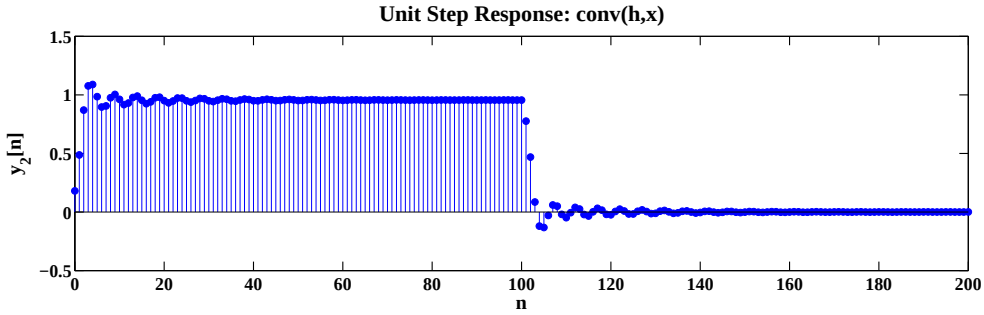


FIGURE 2.24: System step output $y[n]$ computed using the function `y=conv(h,x)`.

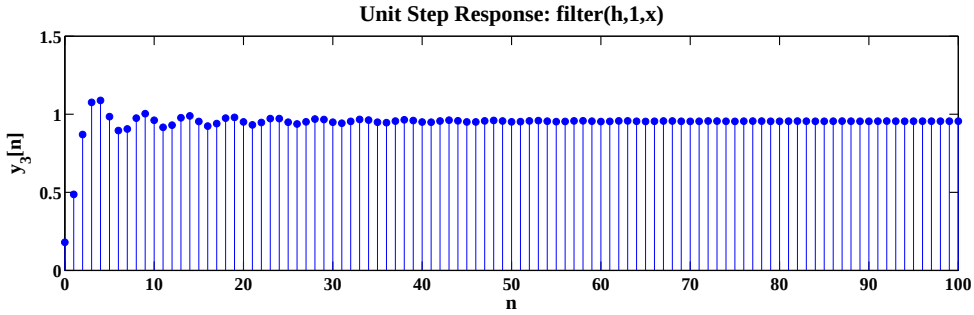


FIGURE 2.25: System step output $y[n]$ computed using the function `y=filter(h,1,x)`.

```
h = impz(b,a,length(n));
% Part (b):
u = unitstep(n(1),0,n(end));
y1 = filter(b,a,u);
% Part (c):
y2 = conv(h,u);
% Part (d):
y3 = filter(h,1,u);
%Plot
hf = figconfg('P0217a','long');
stem(n,h,'fill')
xlabel('n','fontsize',LFS); ylabel('h[n]','fontsize',LFS);
title('Impulse Response h[n]','fontsize',TFS);
```

```
hf2 = figconfg('P0217b','long');
stem(n,y1,'fill')
xlabel('n','fontsize',LFS); ylabel('y_1[n]','fontsize',LFS);
title('Unit Step Response: filter(b,a,x)','fontsize',TFS);
hf3 = figconfg('P0217c','long');
stem(0:2*n(end),y2,'fill')
xlabel('n','fontsize',LFS); ylabel('y_2[n]','fontsize',LFS);
title('Unit Step Response: conv(h,x)','fontsize',TFS);
hf4 = figconfg('P0217d','long');
stem(n,y3,'fill')
xlabel('n','fontsize',LFS); ylabel('y_3[n]','fontsize',LFS);
title('Unit Step Response: filter(h,1,x)','fontsize',TFS);
```

18. (a) Block diagrams.

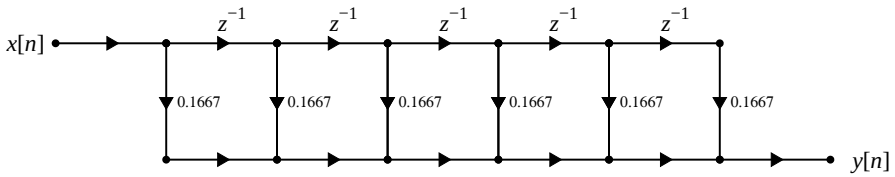


FIGURE 2.26: Block diagram representations of the nonrecursive implementation of $M = 5$ moving average filter.

(b) MATLAB script:

```
% P0218: Implement nonrecursive and recursive implementations
%        of moving average filter
close all; clc
M = 5;
n = 0:M;
un = unitstep(n(1),0,n(end));
% Nonrecursive implementation:
y_nr = filter(ones(1,M+1)/(M+1),1,un);
% Recursive implementation:
y_re = filter([1 zeros(1,M) -1]/(M+1),[1 -1],un);
hf = figconfg('P0218');
subplot(2,1,1)
```

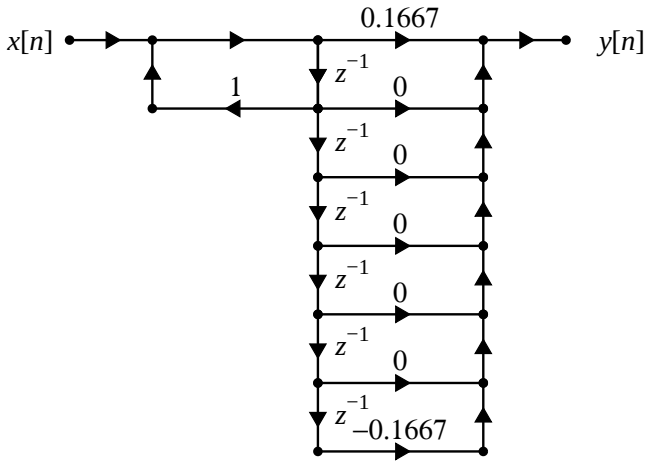


FIGURE 2.27: Block diagram representations of the recursive implementation of $M = 5$ moving average filter.

```
stem(n,y_nr,'fill')
axis([n(1)-1 n(end)+1 min(y_nr)-0.5 max(y_nr)+0.5])
xlabel('n','fontsize',LFS); ylabel('y_1[n]','fontsize',LFS);
title('Nonrecursive Implementation','fontsize',TFS);
subplot(2,1,2)
stem(n,y_re,'fill')
axis([n(1)-1 n(end)+1 min(y_re)-0.5 max(y_re)+0.5])
xlabel('n','fontsize',LFS); ylabel('y_2[n]','fontsize',LFS);
title('Recursive Implementation','fontsize',TFS);
```

19. MATLAB script:

```
% P0219: Generate digital reverberation using audio file 'handel'
close all; clc
load('handel.mat')
n = 1:length(y);
a = 0.7; % specify attenuation factor
tau = 50e-3; % Part (a)
```

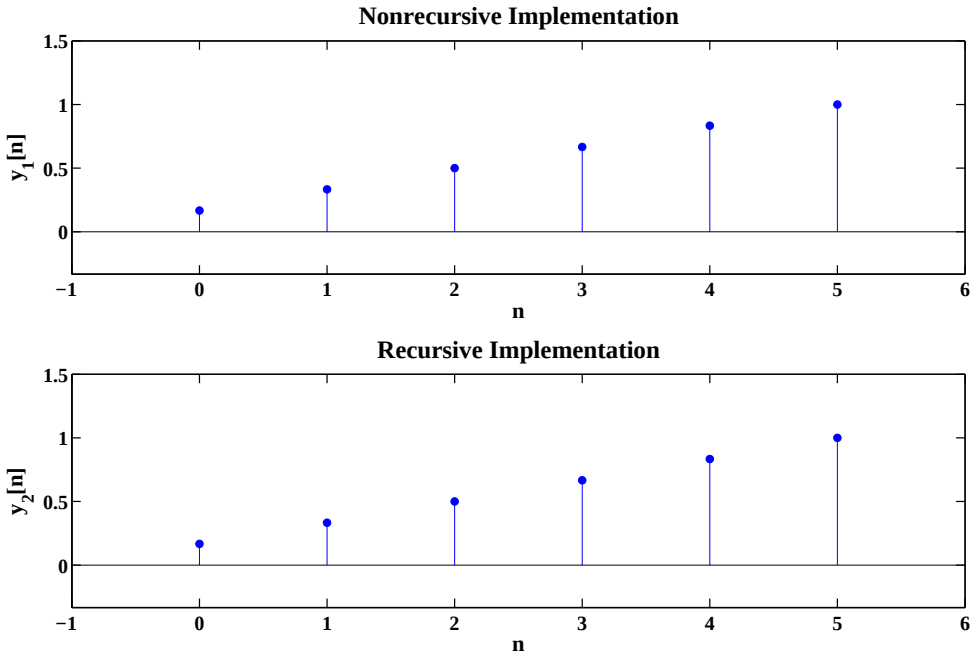


FIGURE 2.28: Step response computed by nonrecursive and recursive implementations.

```
% tau = 100e-3; % Part (b)
% tau = 500e-3; % Part (c)
D = floor(tau*Fs);  % compute delay
yd = filter(1,[1 zeros(1,length(D)-1),-a],y);
sound(yd,Fs)
```

20. (a) Solution:

$$y_1(t) = x_1(t) * h(t) = \int_{-\infty}^{\infty} h(\tau)x_1(t-\tau)d\tau = \int_{-\infty}^{\infty} \mathrm{e}^{-\tau/2}u(\tau)u(t-\tau)d\tau$$
$$= u(t)\int_0^t \mathrm{e}^{-\tau/2}d\tau = u(t)(-2)\mathrm{e}^{-\tau/2}|_0^t = 2(1-\mathrm{e}^{-t/2})u(t)$$

$$
\begin{aligned}
y_2(t) = x_2(t) * h(t) &= \int_{-\infty}^{\infty} h(t-\tau)x_2(\tau)d\tau = 2\int_0^3 \mathrm{e}^{-(t-\tau)/2}u(t-\tau)d\tau \\
&= (u(t)-u(t-3))2\int_0^t \mathrm{e}^{-(t-\tau)/2}d\tau + u(t-3)2\int_0^3 \mathrm{e}^{-(t-\tau)/2}d\tau \\
&= (u(t)-u(t-3))4\mathrm{e}^{-(t-\tau)/2}|_0^t + u(t-3)4\mathrm{e}^{-(t-\tau)/2}|_0^3 \\
&= 4(1-\mathrm{e}^{-t/2})u(t) - 4(1-\mathrm{e}^{-(t-3)/2})u(t-3)
\end{aligned}
$$

(b) Proof:

$$
\begin{aligned}
x_2(t) &= 2x_1(t) - 2x_1(t-3) \\
y_2(t) &= 2y_1(t) - 2y_1(t-3) \\
&= 4(1-\mathrm{e}^{-t/2})u(t) - 4(1-\mathrm{e}^{-(t-3)/2})u(t-3)
\end{aligned}
$$

Basic Problems

21. See book companion toolbox for the function.

22. (a) $x[n]$ versus n.

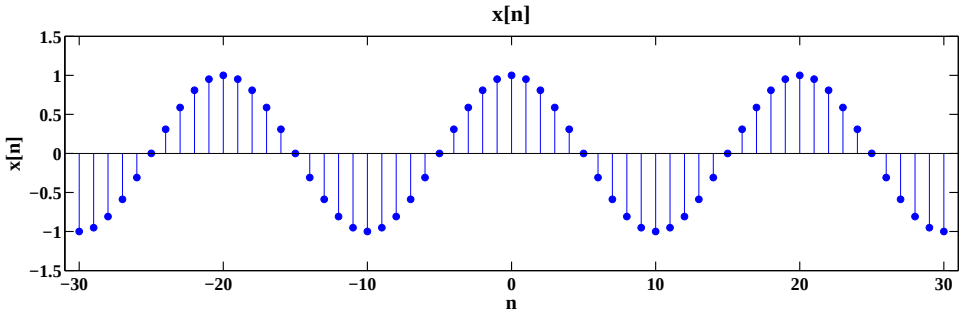


FIGURE 2.29: $x[n]$ versus n.

(b) A down sampled signal $y[n]$ for $M = 5$.

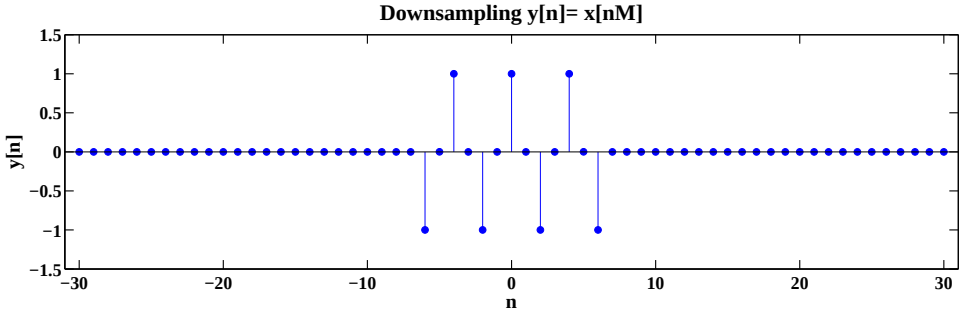


FIGURE 2.30: A down sampled signal $y[n]$ for $M = 5$.

(c) A down sampled signal $y[n]$ for $M = 20$.

(d) Comments: The downsampled signal is compressed.

MATLAB script:

```
% P0222: Illustrate downsampling: y[n] = x[nM]
close all; clc
nx = -30:30;
x = cos(0.1*pi*nx);
```

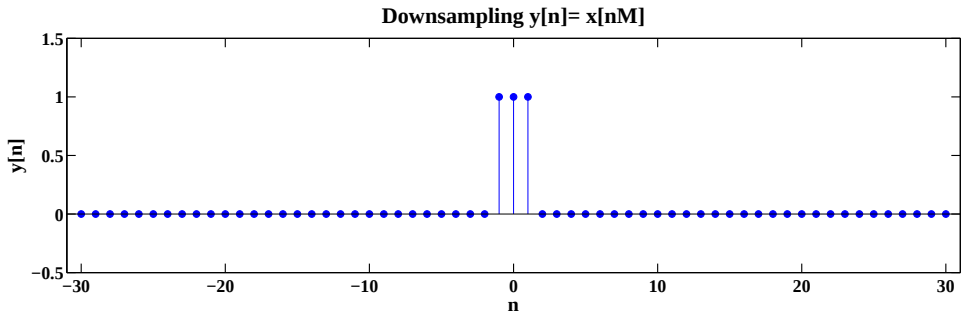


FIGURE 2.31: A down sampled signal $y[n]$ for $M = 20$.

```
% M = 5; % Part (b)
M = 20; % Part (c)
yind = mod(nx,M)==0;
y = x(yind);
ny = nx(yind)/M;
[x y n] = timealign(x,nx,y,ny);
hf = figconfg('P0222a','long');
stem(n,x,'fill')
axis([n(1)-1 n(end)+1 min(x)-0.5 max(x)+0.5])
xlabel('n','fontsize',LFS); ylabel('x[n]','fontsize',LFS);
title('x[n]','fontsize',TFS);
hf2 = figconfg('P0222b','long');
stem(n,y,'fill')
axis([n(1)-1 n(end)+1 min(y)-0.5 max(y)+0.5])
xlabel('n','fontsize',LFS); ylabel('y[n]','fontsize',LFS);
title('Downsampling y[n]= x[nM]','fontsize',TFS);
```

23. (a) $y[n] = x[-n]$ (Time-flip)
linear, time-variant, noncausal, and stable

(b) $y[n] = \log(|x[n]|)$ (Log-magnitude)
nonlinear, time-invariant, causal, and unstable

(c) $y[n] = x[n] - x[n-1]$ (First-difference)
linear, time-invariant, causal, and stable

(d) $y[n] = \text{round}\{x[n]\}$ (Quantizer)
nonlinear, time-invariant, causal, and stable

24. Comments: The filtered data are smoother and $y_1[n]$ is 25 samples delayed than $y_2[n]$.

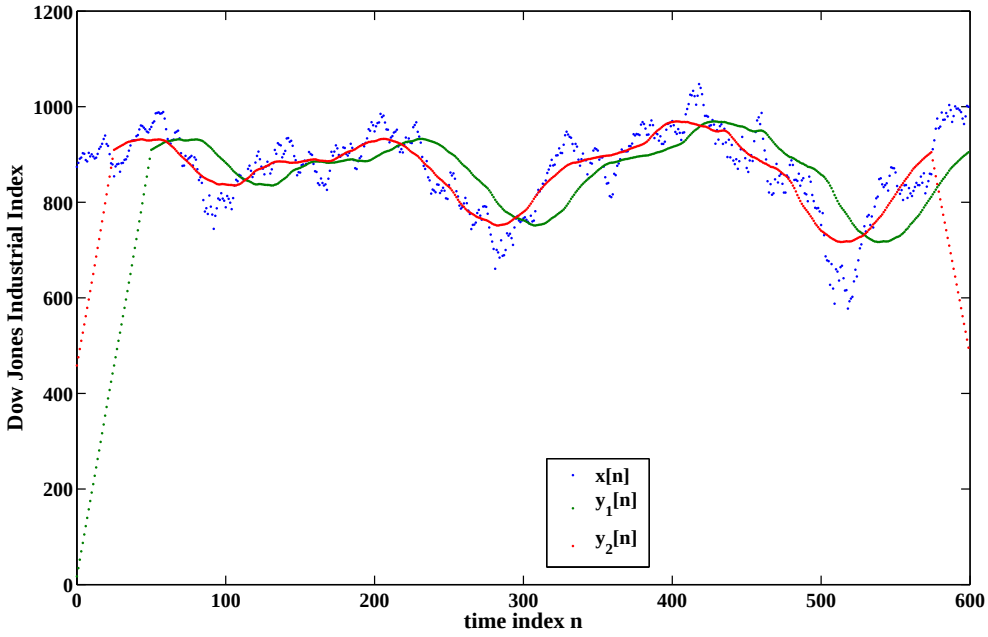


FIGURE 2.32: Dow Jones Industrial Average weekly opening value $x[n]$ and its moving averages.

MATLAB script:

```
% P0224: Write MATLAB script to compute moving averages
close all; clc
x = load('djw6576.txt');
N = length(x);
nx = 0:N-1;
xepd1 = [zeros(50,1);x];
y1 = zeros(N,1);
for ii = 1:N
    y1(ii) = sum(xepd1(ii:ii+50))/51;
end
xepd2 = [zeros(25,1);x;zeros(25,1)];
```

```
y2 = zeros(N,1);
for ii = 1:N
    y2(ii) = sum(xepd2(ii:ii+50))/51;
end
% Plot:
hf = figconfg('P0224');
plot(nx,x,'.',nx,y1,'.',nx,y2,'.')
xlabel('time index n','fontsize',LFS)
ylabel('Dow Jones Industrial Index','fontsize',LFS)
legend('x[n]','y_1[n]','y_2[n]','fontsize',LFS,'location','best')
```

25. (a) Solution:

$$y[n] = h[n] * x[n] = \sum_{m=-\infty}^{\infty} h[m]x[n-m]$$

$$= \sum_{m=-\infty}^{\infty} m(u[m] - u[m-M])(u[n-m] - u[n-M-N])$$

if $n \in [0, M-1]$

$$y[n] = \sum_{m=0}^{n} m = \frac{n(n+1)}{2}$$

if $n \in [M-1, N-1]$

$$y[n] = \sum_{m=0}^{M-1} m = \frac{M(M-1)}{2}$$

if $n \in [N-1, M+N-3]$

$$y[n] = \sum_{m=n-(N-1)}^{M-1} m = \sum_{m=0}^{M-1} m - \sum_{m=0}^{n-N} m = \frac{M(M-1)}{2} - \frac{(n-N+1)(n-N)}{2}$$

(b) Comments: The analytical solution can be verified.

MATLAB script:

```
% P0225: Verify the analytical expression
close all; clc
N = 10; M = 5;
n = 0:N-1;
```

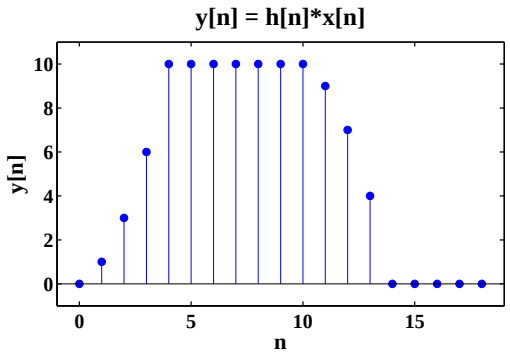


FIGURE 2.33: MATLAB verification of analytical expression for the sequence $y[n] = h[n] * x[n]$.

```
x = unitpulse(0,0,N-1,N-1)';
h = n.*unitpulse(0,0,M-1,N-1)';
[y ny] = conv0(h,n,x,n);
% Plot:
hf = figconfg('P0225','small');
stem(ny,y,'fill')
axis([ny(1)-1,ny(end)+1,min(y)-1,max(y)+1])
xlabel('n','fontsize',LFS); ylabel('y[n]','fontsize',LFS);
title('y[n] = h[n]*x[n]','fontsize',TFS)
```

26. Solution:

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[n-k]h[k]$$

if $n \in [0, N-1]$

$$y[n] = \sum_{k=0}^{n} a^k = \frac{1-a^{n+1}}{1-a}$$

if $n \in [N-1, M-1]$

$$y[n] = \sum_{k=n-N+1}^{n} a^k = \sum_{k=0}^{n} a^k - \sum_{k=0}^{n-N} a^k = \frac{a^{n+1}(a^{-N}-1)}{1-a}$$

if $n \in [M-1, M+N-2]$

$$y[n] = \sum_{k=n-N+1}^{M-1} a^k = \sum_{k=0}^{M-1} a^k - \sum_{k=0}^{n-N} a^k = \frac{a^{n-N+1} - a^M}{1-a}$$

$y[n] = 0, \quad$ otherwise

27. Solution:

$$\begin{aligned} y[n] &= h[n] * x[n] = a^n u[n] * b^n u[n] \\ &= \sum_{m=-\infty}^{\infty} a^m u[m] b^{n-m} u[n-m] = u[n] \sum_{m=0}^{n} a^m b^{n-m} \\ &= u[n] b^n \sum_{m=0}^{n} a^m b^{-m} = \frac{b^{n+1} - a^{n+1}}{b-a} u[n] \end{aligned}$$

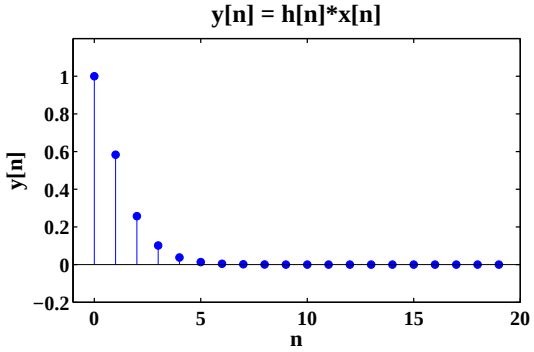


FIGURE 2.34: MATLAB verification of analytical expression for the sequence $y[n] = h[n] * x[n]$.

MATLAB script:

```
% P0227: Verify the analytical expression
close all; clc
a = 1/4; b = 1/3;
N = 20;
n = 0:N-1;
x = a.^n;
h = b.^n;
y = conv(h,x);
```

```
% Plot:
hf = figconfg('P0227','small');
stem(n,y(1:N),'fill')
axis([n(1)-1,n(end)+1,min(y)-0.2,max(y)+0.2])
xlabel('n','fontsize',LFS); ylabel('y[n]','fontsize',LFS);
title('y[n] = h[n]*x[n]','fontsize',TFS)
```

28. (a) Solution:

$$y[n] = x[n] * h[n] = \sum_{m=-\infty}^{\infty} (0.9)^m u[m](0.9)^{n-m} u[n-m]$$

$$= u[n] \sum_{m=0}^{n} (0.9)^n = (n+1)(0.9)^n u[n]$$

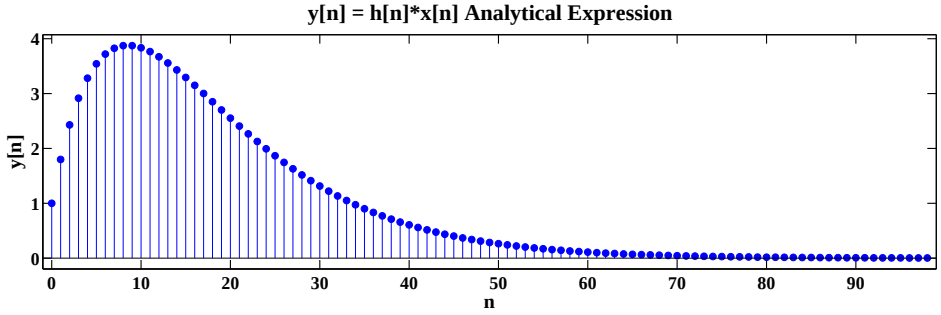


FIGURE 2.35: $y[n]$ plot determined analytically.

(b) $y[n]$ computed by `conv` function.

(c) $y[n]$ computed by `filter` function.

(d) Comments: (c) comes closer to (a). Because in (b) the tail parts (samples from $n = 50$) of both $x[n]$ and $h[n]$ are curtailed, the second part samples (samples from $n = 50$) of (b) differ from the ones in (a).

MATLAB script:

```
% P0228: Verify the analytical expression
close all; clc
a = 0.9;
% Part (a): Analytical Result:
```

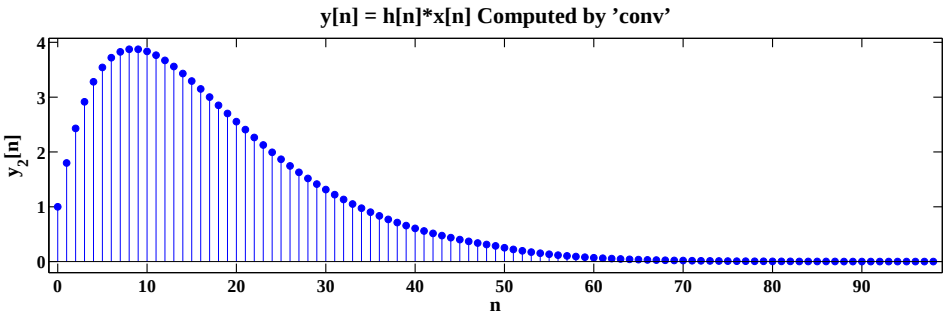


FIGURE 2.36: $y[n]$ plot determined by `conv` function.

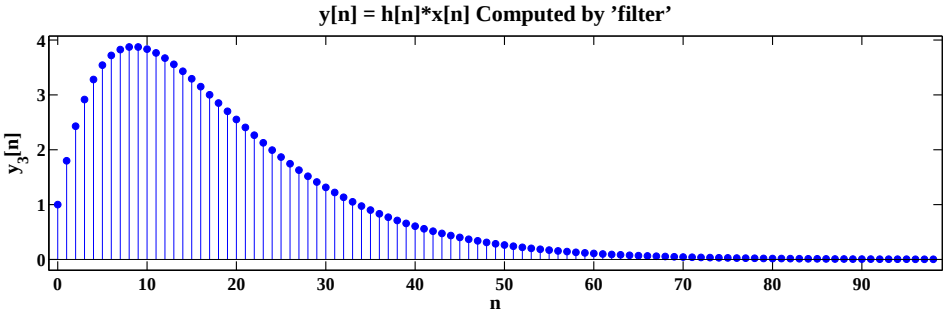


FIGURE 2.37: $y[n]$ plot determined by `filter` function.

```
n = 0:98;
y1 = (n+1).*a.^n;
% Plot:
hf1 = figconfg('P0228a','long');
stem(n,y1,'fill')
axis([n(1)-1,n(end)+1,min(y1)-0.2,max(y1)+0.2])
xlabel('n','fontsize',LFS); ylabel('y[n]','fontsize',LFS);
title('y[n] = h[n]*x[n] Analytical Expression','fontsize',TFS)
% Part (b): Using 'conv'
N = 50;
n = 0:N-1;
x = a.^n;
h = a.^n;
y2 = conv(h,x);
ny = 0:length(y2)-1;
```