Digital Signal Processing Laboratory (DSP Lab)

Dr. Roozbeh Rajabi

Spring 2018

Reference

- Vinay K. Ingle, John G. Proakis, “Digital Signal Processing Using MATLAB”, Third Edition, Cengage Learning, 2011

Contents

Software

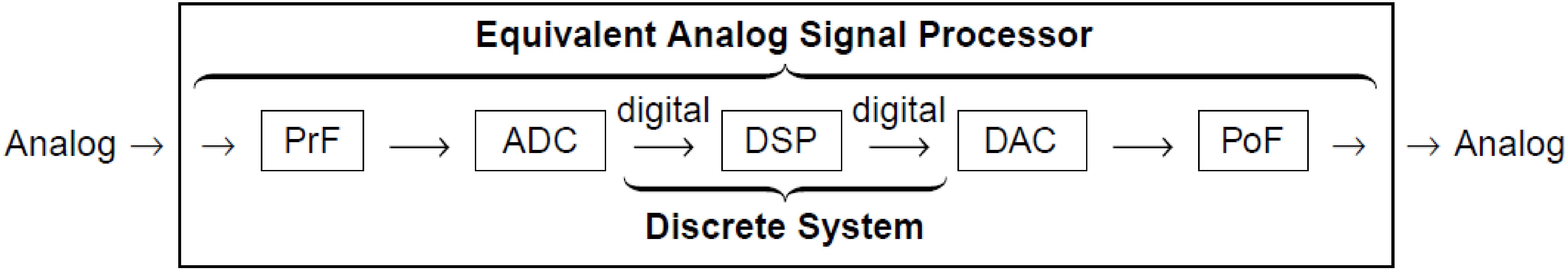
- MATLAB R2017b

- Code Composer Studio

1. Introduction

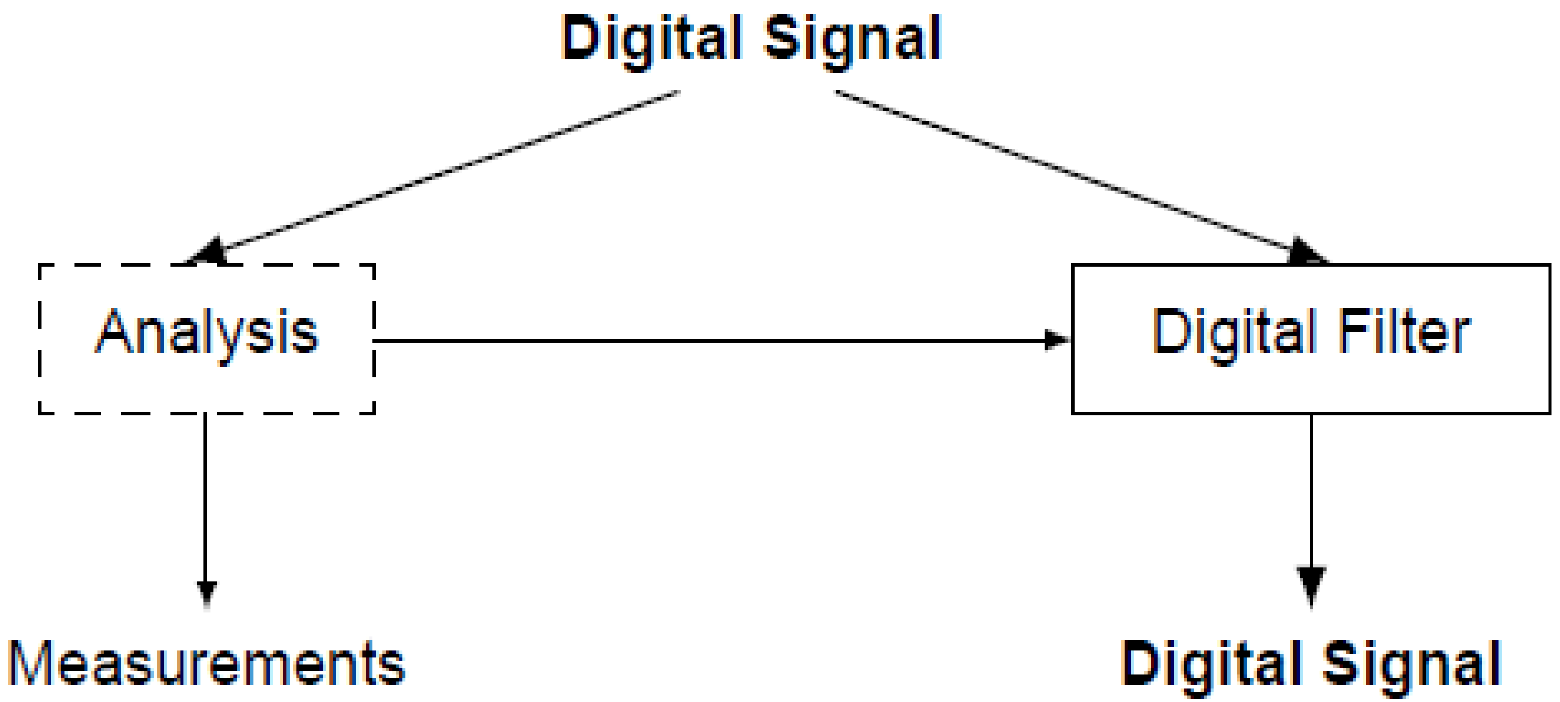
- How are signals processed?

$$\text{Analog signal: } x_a(t) \longrightarrow \boxed{\text{Analog signal processor}} \longrightarrow y_a(t) \text{ :Analog signal}$$



1. Introduction

- Two important categories of DSP



1. Introduction

- A Brief Introduction to MATLAB

1. Introduction

- Example 1.1.

$$x(t) = \sin(2\pi t) + \tfrac{1}{3}\sin(6\pi t) + \tfrac{1}{5}\sin(10\pi t) = \sum_{k=1}^{3} \frac{1}{k}\sin(2\pi k t), \qquad 0 \le t \le 1$$

- 0:0.01:1
- Three Approaches.

1. Introduction

- Scripts and Functions
- Write a script file to implement:

$$x(t) = \sum_{k=1}^{K} c_k \sin(2\pi k t)$$

- Functions:
- Write a function
  - Name: sinsum
  - Inputs: t, ck
  - Output: xt

1. Introduction

- Plotting:
  - Plot sin(2*pi*t)
  - Stem plot
  - TeX Markup: \pi
  - Set properties using handle
  - Subplot

1.3 Applications of DSP

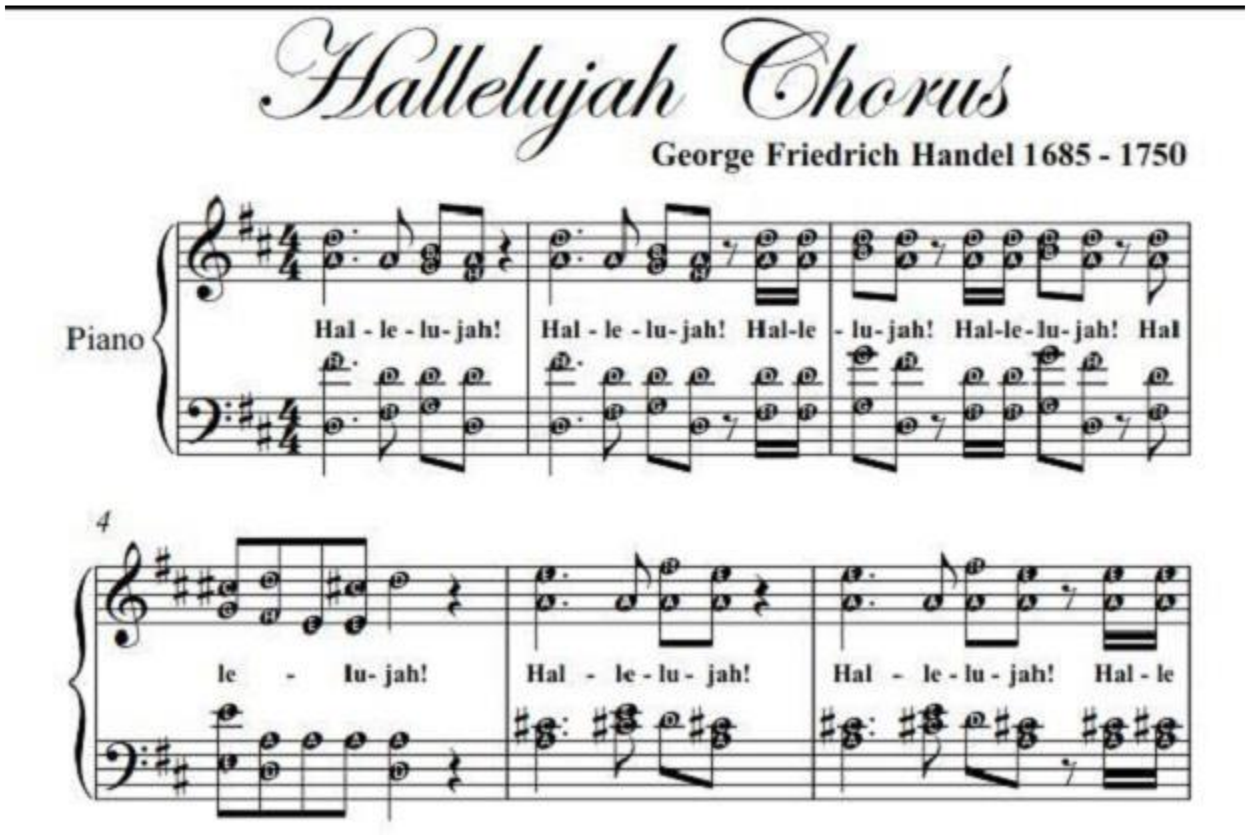
- speech/audio (speech recognition/synthesis, digital audio, equalization, etc.),
- image/video (enhancement, coding for storage and transmission, robotic vision, animation, etc.),
- military/space (radar processing, secure communication, missile guidance, sonar processing, etc.),
- biomedical/health care (scanners, ECG analysis, X-ray analysis, EEG brain mappers, etc.)
- consumer electronics (cellular/mobile phones, digital television, digital camera, Internet voice/music/video, interactive entertainment systems, etc) and many more

Musical Sound Processing

- a short snippet of
- Handel's hallelujah chorus

- Available in MATLAB

- load handel;



Musical Sound Processing

- Echo Generation:

$$x[n] = y[n] + \alpha y[n - D], \qquad |\alpha| < 1$$

  - Add echo to original sound using filter

- Echo Removal

  - Remove echo using inverse filtering

Musical Sound Processing

- Digital Reverberation:

$$x[n] = \sum_{k=0}^{N-1} \alpha^k y[n - kD]$$

- Another Reverberation Model:

$$x[n] = \alpha y[n] + y[n - D] + \alpha x[n - D], \quad |\alpha| < 1$$

2. Discrete-time Signals and Systems

- Discrete-time Signal:

$$x(n) = \{2, 1, -1, \underset{\uparrow}{0}, 1, 4, 3, 7\}$$

```
>> n=[-3,-2,-1,0,1,2,3,4];  x=[2,1,-1,0,1,4,3,7];
```

- Unit sample sequence:

$$\delta(n) = \begin{cases} 1, & n = 0 \\ 0, & n \neq 0 \end{cases} = \left\{ \ldots, 0, 0, \underset{\uparrow}{1}, 0, 0, \ldots \right\}$$

$$\delta(n - n_0) = \begin{cases} 1, & n = n_0 \\ 0, & n \neq n_0 \end{cases}$$

```
function [x,n] = impseq(n0,n1,n2)
% Generates x(n) = delta(n-n0); n1 <= n <= n2
% ----------------------------------------------
% [x,n] = impseq(n0,n1,n2)
%
n = [n1:n2]; x = [(n-n0) == 0];
```

2. Discrete-time Signals and Systems

- Unit step sequence:

$$u(n) = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases} = \{\ldots, 0, 0, \underset{\uparrow}{1}, 1, 1, \ldots\}$$

$$u(n - n_0) = \begin{cases} 1, & n \geq n_0 \\ 0, & n < n_0 \end{cases}$$

```
function [x,n] = stepseq(n0,n1,n2)
% Generates x(n) = u(n-n0); n1 <= n <= n2
% ------------------------------------------
% [x,n] = stepseq(n0,n1,n2)
%
n = [n1:n2]; x = [(n-n0) >= 0];
```

2. Discrete-time Signals and Systems

- Real-valued exponential sequence:

$$x(n) = a^n, \forall n; \quad a \in \mathbb{R}$$

- Complex-valued exponential sequence:

$$x(n) = e^{(\sigma + j\omega_0)n}, \forall n$$

- Sinusoidal sequence:

$$x(n) = A\cos(\omega_0 n + \theta_0), \forall n$$

2. Discrete-time Signals and Systems

- Random sequences:
  - Uniform distribution: rand
  - Gaussian distribution: randn

- Periodic sequence:

```
>> xtilde = [x,x,...,x];
```

```
>> xtilde = x' * ones(1,P);   % P columns of x; x is a row vector
>> xtilde = xtilde(:);        % long column vector
>> xtilde = xtilde';          % long row vector
```

2. Discrete-time Signals and Systems

- Operations on sequences:
  - Signal addition:

```
function [y,n] = sigadd(x1,n1,x2,n2)
% implements y(n) = x1(n)+x2(n)
% -----------------------------
% [y,n] = sigadd(x1,n1,x2,n2)
%   y = sum sequence over n, which includes n1 and n2
%  x1 = first sequence over n1
%  x2 = second sequence over n2 (n2 can be different from n1)
%
n = min(min(n1),min(n2)):max(max(n1),max(n2));  % duration of y(n)
y1 = zeros(1,length(n)); y2 = y1;               % initialization
y1(find((n>=min(n1))&(n<=max(n1))==1))=x1;     % x1 with duration of y
y2(find((n>=min(n2))&(n<=max(n2))==1))=x2;     % x2 with duration of y
y = y1+y2;                                      % sequence addition
```

2. Discrete-time Signals and Systems

- Operations on sequences:
  - Signal multiplication
  - Scaling
  - Shifting:

```
function [y,n] = sigshift(x,m,k)
% implements y(n) = x(n-k)
% -------------------------
% [y,n] = sigshift(x,m,k)
%
n = m+k; y = x;
```

2. Discrete-time Signals and Systems

- Folding:

```
function [y,n] = sigfold(x,n)
% implements y(n) = x(-n)
% -----------------------
% [y,n] = sigfold(x,n)
%
y = fliplr(x); n = -fliplr(n);
```

- Sample summation: sum
- Sample products: prod

2. Discrete-time Signals and Systems

- Signal energy:

$$\mathcal{E}_x = \sum_{-\infty}^{\infty} x(n)x^*(n) = \sum_{-\infty}^{\infty} |x(n)|^2$$

- Signal power:

$$\mathcal{P}_x = \frac{1}{N}\sum_{0}^{N-1} |\tilde{x}(n)|^2$$

2. Discrete-time Signals and Systems

EXAMPLE 2.1 Generate and plot each of the following sequences over the indicated interval.

a. $x(n) = 2\delta(n+2) - \delta(n-4), \quad -5 \le n \le 5.$
b. $x(n) = n[u(n) - u(n-10)] + 10e^{-0.3(n-10)}[u(n-10) - u(n-20)], \ 0 \le n \le 20.$
c. $x(n) = \cos(0.04\pi n) + 0.2w(n), \ 0 \le n \le 50,$ where $w(n)$ is a Gaussian random sequence with zero mean and unit variance.
d. $\tilde{x}(n) = \{..., 5, 4, 3, 2, 1, \underset{\uparrow}{5}, 4, 3, 2, 1, 5, 4, 3, 2, 1, ...\}; \ -10 \le n \le 9.$

EXAMPLE 2.2 Let $x(n) = \{1, 2, \underset{\uparrow}{3}, 4, 5, 6, 7, 6, 5, 4, 3, 2, 1\}$. Determine and plot the following sequences.

a. $x_1(n) = 2x(n-5) - 3x(n+4)$
b. $x_2(n) = x(3-n) + x(n)\,x(n-2)$

2. Discrete-time Signals and Systems

EXAMPLE 2.3 Generate the complex-valued signal

$$x(n) = e^{(-0.1+j0.3)n}, \quad -10 \le n \le 10$$

and plot its magnitude, phase, the real part, and the imaginary part in four separate subplots.

2. Discrete-time Signals and Systems

- Systems
  - Linearity
  - LTI
  - Stability
  - Causality
  - Convolution

2. Discrete-time Signals and Systems

- MATLAB Implementation
  - Convolution
  - y=conv(x,h)
    - Without timing information

```
>> x = [3, 11, 7, 0, -1, 4, 2];  h = [2, 3, 0, -5, 2, 1];
>> y = conv(x, h)
y =
   6    31    47     6    -51    -5    41    18    -22    -3    8    2
```

2. Discrete-time Signals and Systems

- MATLAB Implementation
  - Modified Convolution
  - y=conv_m(x,nx,h,nh)
    - Including timing information

```
function [y,ny] = conv_m(x,nx,h,nh)
% Modified convolution routine for signal processing
% ----------------------------------------------
%  [y,ny] = conv_m(x,nx,h,nh)
%  [y,ny] = convolution result
%  [x,nx] = first signal
%  [h,nh] = second signal
%
nyb = nx(1)+nh(1); nye = nx(length(x)) + nh(length(h));
ny = [nyb:nye];   y = conv(x,h);
```

2. Discrete-time Signals and Systems

- MATLAB Implementation
  - Modified Convolution
  - y=conv_m(x,nx,h,nh)
    - Example

```
>> x = [3, 11, 7, 0, -1, 4, 2]; nx = [-3:3];
>> h = [2, 3, 0, -5, 2, 1]; ny = [-1:4];
```

```
>> [y,ny] = conv_m(x,nx,h,nh)
y =
   6    31    47     6   -51    -5    41    18   -22    -3     8     2
ny =
  -4    -3    -2    -1     0     1     2     3     4     5     6     7
```

2. Discrete-time Signals and Systems

- MATLAB Implementation
  - Modified Convolution
  - y=conv_m(x,nx,h,nh)
    - Example

```
>> x = [3, 11, 7, 0, -1, 4, 2]; nx = [-3:3];
>> h = [2, 3, 0, -5, 2, 1]; ny = [-1:4];
```

```
>> [y,ny] = conv_m(x,nx,h,nh)
y =
  6    31    47     6    -51    -5    41    18    -22    -3     8     2
ny =
 -4    -3    -2    -1     0     1     2     3     4     5     6     7
```

2. Discrete-time Signals and Systems

- MATLAB Implementation
  - Crosscorrelation between vectors x and y
    - xcorr(x,y)

  - Autocorrelation of vector x
    - xcorr(x)

  - Without timing information

2. Discrete-time Signals and Systems

- MATLAB Implementation
  - Crosscorrelation using conv_m

$$r_{yx}(\ell) = y(\ell) * x(-\ell)$$

EXAMPLE 2.10 In this example we will demonstrate one application of the crosscorrelation sequence. Let

$$x(n) = [3, 11, 7, \underset{\uparrow}{0}, -1, 4, 2]$$

be a prototype sequence, and let $y(n)$ be its noise-corrupted-and-shifted version

$$y(n) = x(n-2) + w(n)$$

where $w(n)$ is Gaussian sequence with mean 0 and variance 1. Compute the crosscorrelation between $y(n)$ and $x(n)$.

2. Discrete-time Signals and Systems

- MATLAB Implementation
  - Example:

```
% noise sequence 1
>> x = [3, 11, 7, 0, -1, 4, 2]; nx=[-3:3]; % given signal x(n)
>> [y,ny] = sigshift(x,nx,2);              % obtain x(n-2)
>> w = randn(1,length(y)); nw = ny;        % generate w(n)
>> [y,ny] = sigadd(y,ny,w,nw);             % obtain y(n) = x(n-2) + w(n)
>> [x,nx] = sigfold(x,nx);                 % obtain x(-n)
```

```
>> [rxy,nrxy] = conv_m(y,ny,x,nx);         % crosscorrelation
>> subplot(1,1,1), subplot(2,1,1);stem(nrxy,rxy)
>> axis([-5,10,-50,250]);xlabel('lag variable l')
>> ylabel('rxy');title('Crosscorrelation: noise sequence 1')
%
% noise sequence 2
>> x = [3, 11, 7, 0, -1, 4, 2]; nx=[-3:3]; % given signal x(n)
>> [y,ny] = sigshift(x,nx,2);              % obtain x(n-2)
>> w = randn(1,length(y)); nw = ny;        % generate w(n)
>> [y,ny] = sigadd(y,ny,w,nw);             % obtain y(n) = x(n-2) + w(n)
>> [x,nx] = sigfold(x,nx);                 % obtain x(-n)
>> [rxy,nrxy] = conv_m(y,ny,x,nx);         % crosscorrelation
>> subplot(2,1,2);stem(nrxy,rxy)
>> axis([-5,10,-50,250]);xlabel('lag variable l')
>> ylabel('rxy');title('Crosscorrelation: noise sequence 2')
```

2. Discrete-time Signals and Systems

- Difference Equations

$$\sum_{k=0}^{N} a_k y(n-k) = \sum_{m=0}^{M} b_m x(n-m), \quad \forall n$$

```
y = filter(b,a,x)
```

```
b = [b0, b1, ..., bM]; a = [a0, a1, ..., aN];
```

```
h = impz(b,a,n);
```

EXAMPLE 2.11 Given the following difference equation

$$y(n) - y(n-1) + 0.9y(n-2) = x(n); \quad \forall n$$

a. Calculate and plot the impulse response $h(n)$ at $n = -20, \ldots, 100$.
b. Calculate and plot the unit step response $s(n)$ at $n = -20, \ldots, 100$.
c. Is the system specified by $h(n)$ stable?

2. Discrete-time Signals and Systems

- Difference Equations
  - Solution:

```
b = [1]; a=[1, -1, 0.9];
```

```
>> b = [1]; a = [1, -1, 0.9];  n = [-20:120];
>> h = impz(b,a,n);
>> subplot(2,1,1); stem(n,h);
>> title('Impulse Response'); xlabel('n'); ylabel('h(n)')
```

```
>> x = stepseq(0,-20,120);   s = filter(b,a,x);
>> subplot(2,1,2); stem(n,s)
>> title('Step Response'); xlabel('n'); ylabel('s(n)')
```

```
>> sum(abs(h))
ans = 14.8785
```

```
>>z = roots(a);   magz = abs(z)
magz =  0.9487
        0.9487
```

2. Discrete-time Signals and Systems

- Difference Equations
  - Example:

EXAMPLE 2.12 Let us consider the convolution given in Example 2.7. The input sequence is of finite duration

$$x(n) = u(n) - u(n-10)$$

while the impulse response is of infinite duration

$$h(n) = (0.9)^n \, u(n)$$

Determine $y(n) = x(n) * h(n)$.

2. Discrete-time Signals and Systems

- Difference Equations
  - Example:

```
>> b = [1]; a = [1,-0.9];
>> n = -5:50; x = stepseq(0,-5,50) - stepseq(10,-5,50);
>> y = filter(b,a,x);
>> subplot(2,1,2); stem(n,y); title('Output sequence')
>> xlabel('n'); ylabel('y(n)'); axis([-5,50,-0.5,8])
```

2. Discrete-time Signals and Systems

- Digital Filters
  - FIR Filter:

$$y(n) = \sum_{m=0}^{M} b_m x(n-m)$$

  - IIR Filter:

$$\sum_{k=0}^{N} a_k y(n-k) = x(n)$$

3. The Discrete-Time Fourier Transform (DTFT)

- DTFT:

$$X(e^{j\omega}) \triangleq \mathcal{F}[x(n)] = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n}$$

- IDTFT:

$$x(n) \triangleq \mathcal{F}^{-1}[X(e^{j\omega})] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n}d\omega$$

- Properties:
  - 1. Periodicity
  - 2. Symmetry: real-valued

3. The Discrete-Time Fourier Transform (DTFT)

EXAMPLE 3.1 Determine the discrete-time Fourier transform of $x(n) = (0.5)^n\, u(n)$.

$$X(e^{j\omega}) = \sum_{-\infty}^{\infty} x(n)e^{-j\omega n} = \sum_{0}^{\infty} (0.5)^n\, e^{-j\omega n}$$

$$= \sum_{0}^{\infty} (0.5e^{-j\omega})^n = \frac{1}{1 - 0.5e^{-j\omega}} = \frac{e^{j\omega}}{e^{j\omega} - 0.5}$$

3. The Discrete-Time Fourier Transform (DTFT)

EXAMPLE 3.3 Evaluate $X(e^{j\omega})$ in Example 3.1 at 501 equispaced points between $[0, \pi]$ and plot its magnitude, angle, real, and imaginary parts.

```
>> w = [0:1:500]*pi/500;  % [0, pi]  axis divided into 501 points.
>> X = exp(j*w) ./ (exp(j*w) - 0.5*ones(1,501));
>> magX = abs(X); angX = angle(X);  realX = real(X); imagX = imag(X);
>> subplot(2,2,1); plot(w/pi,magX); grid
>> xlabel('frequency in pi units'); title('Magnitude Part'); ylabel('Magnitude')
>> subplot(2,2,3); plot(w/pi,angX); grid
>> xlabel('frequency in pi units'); title('Angle Part'); ylabel('Radians')
>> subplot(2,2,2); plot(w/pi,realX); grid
>> xlabel('frequency in pi units'); title('Real Part'); ylabel('Real')
>> subplot(2,2,4); plot(w/pi,imagX); grid
>> xlabel('frequency in pi units'); title('Imaginary Part'); ylabel('Imaginary')
```

3. The Discrete-Time Fourier Transform (DTFT)

- Finite Duration x(n)

EXAMPLE 3.2 Determine the discrete-time Fourier transform of the following finite-duration sequence:

$$x(n) = \{1, \underset{\uparrow}{2}, 3, 4, 5\}$$

$$X(e^{j\omega}) = \sum_{-\infty}^{\infty} x(n)e^{-j\omega n} = e^{j\omega} + 2 + 3e^{-j\omega} + 4e^{-j2\omega} + 5e^{-j3\omega}$$

3. The Discrete-Time Fourier Transform (DTFT)

- Finite Duration x(n)

$$\omega_k \triangleq \frac{\pi}{M}k, \quad k = 0, 1, \ldots, M$$

$$X(e^{j\omega_k}) = \sum_{\ell=1}^{N} e^{-j(\pi/M)kn_\ell} x(n_\ell), \quad k = 0, 1, \ldots, M$$

$$\mathbf{X} = \mathbf{W}\mathbf{x}$$

$$\mathbf{W} \triangleq \left\{ e^{-j(\pi/M)kn_\ell};\ n_1 \le n \le n_N, \quad k = 0, 1, \ldots, M \right\}$$

$$\mathbf{W} = \left[\exp\left(-j\frac{\pi}{M}\mathbf{k}^T\mathbf{n} \right) \right] \qquad \mathbf{X}^T = \mathbf{x}^T \left[\exp\left(-j\frac{\pi}{M}\mathbf{n}^T\mathbf{k} \right) \right]$$

3. The Discrete-Time Fourier Transform (DTFT)

- Finite Duration x(n)

```
>> k = [0:M]; n = [n1:n2];
>> X = x * (exp(-j*pi/M)) .^ (n'*k);
```

EXAMPLE 3.4 Numerically compute the discrete-time Fourier transform of the sequence $x(n)$ given in Example 3.2 at 501 equispaced frequencies between $[0, \pi]$.

```
>> n = -1:3; x = 1:5;  k = 0:500; w = (pi/500)*k;
>> X = x * (exp(-j*pi/500)) .^ (n'*k);
>> magX = abs(X); angX = angle(X);
>> realX = real(X); imagX = imag(X);
>> subplot(2,2,1); plot(k/500,magX);grid
>> xlabel('frequency in pi units'); title('Magnitude Part')
>> subplot(2,2,3); plot(k/500,angX/pi);grid
>> xlabel('frequency in pi units'); title('Angle Part')
>> subplot(2,2,2); plot(k/500,realX);grid
>> xlabel('frequency in pi units'); title('Real Part')
>> subplot(2,2,4); plot(k/500,imagX);grid
>> xlabel('frequency in pi units'); title('Imaginary Part')
```

3. The Discrete-Time Fourier Transform (DTFT)

- Finite Duration x(n)

EXAMPLE 3.5 Let $x(n) = (0.9 \exp(j\pi/3))^n$, $0 \leq n \leq 10$. Determine $X(e^{j\omega})$ and investigate its periodicity.

```
>> n = 0:10; x = (0.9*exp(j*pi/3)).^n;
>> k = -200:200; w = (pi/100)*k;
>> X = x * (exp(-j*pi/100)) .^ (n'*k);
>> magX = abs(X); angX =angle(X);
>> subplot(2,1,1); plot(w/pi,magX);grid
>> xlabel('frequency in units of pi'); ylabel('|X|')
>> title('Magnitude Part')
>> subplot(2,1,2); plot(w/pi,angX/pi);grid
>> xlabel('frequency in units of pi'); ylabel('radians/pi')
>> title('Angle Part')
```

3. The Discrete-Time Fourier Transform (DTFT)

- Finite Duration x(n)

EXAMPLE 3.6 Let $x(n) = (0.9)^n, \ -10 \leq n \leq 10$. Investigate the conjugate-symmetry property of its discrete-time Fourier transform.

```
>> n = -5:5; x = (-0.9).^n;
>> k = -200:200; w = (pi/100)*k;  X = x * (exp(-j*pi/100)) .^ (n'*k);
>> magX = abs(X); angX =angle(X);
>> subplot(2,1,1); plot(w/pi,magX);grid;  axis([-2,2,0,15])
>> xlabel('frequency in units of pi'); ylabel('|X|')
>> title('Magnitude Part')
>> subplot(2,1,2); plot(w/pi,angX/pi);grid;  axis([-2,2,-1,1])
>> xlabel('frequency in units of pi'); ylabel('radians/pi')
>> title('Angle Part')
```

3. The Discrete-Time Fourier Transform (DTFT)

- Finite Duration x(n)

EXAMPLE 3.7 In this example we will verify the linearity property (3.5) using real-valued finite-duration sequences. Let $x_1(n)$ and $x_2(n)$ be two random sequences uniformly distributed between $[0, 1]$ over $0 \leq n \leq 10$. Then we can use our numerical discrete-time Fourier transform procedure as follows.

```
>> x1 = rand(1,11); x2 = rand(1,11); n = 0:10;
>> alpha = 2; beta = 3;  k = 0:500; w = (pi/500)*k;
>> X1 = x1 * (exp(-j*pi/500)).^(n'*k);  % DTFT of x1
>> X2 = x2 * (exp(-j*pi/500)).^(n'*k);  % DTFT of x2
>> x = alpha*x1 + beta*x2;              % Linear combination of x1 & x2
>> X = x * (exp(-j*pi/500)).^(n'*k);    % DTFT of x
>> % verification
>> X_check = alpha*X1 + beta*X2;        % Linear Combination of X1 & X2
>> error = max(abs(X-X_check))          % Difference
error =
  7.1054e-015
```

3. The Discrete-Time Fourier Transform (DTFT)

- Finite Duration x(n)

EXAMPLE 3.8 Let $x(n)$ be a random sequence uniformly distributed between $[0, 1]$ over $0 \leq n \leq 10$ and let $y(n) = x(n-2)$. Then we can verify the sample shift property (3.6) as follows.

```
>> x = rand(1,11); n = 0:10;
>> k = 0:500; w = (pi/500)*k;
>> X = x * (exp(-j*pi/500)).^(n'*k);     % DTFT of x
>> % signal shifted by two samples
>> y = x; m = n+2;
>> Y = y * (exp(-j*pi/500)).^(m'*k);     % DTFT of y
>> % verification
>> Y_check = (exp(-j*2).^w).*X;          % multiplication by exp(-j2w)
>> error = max(abs(Y-Y_check))           % Difference
error =
  5.7737e-015
```