Filters

ELEC 3004: **Systems**: Signals & Controls
Dr. Surya Singh

Lecture 10
(with material from Lathi and Cannon (Discrete Systems))

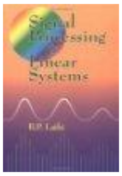
elec3004@itee.uq.edu.au
http://robotics.itee.uq.edu.au/~elec3004/

April 7, 2016

Lecture Schedule:

Week	Date	Lecture Title
1	29-Feb	Introduction
	3-Mar	Systems Overview
2	7-Mar	Systems as Maps & Signals as Vectors
	10-Mar	Data Acquisition & Sampling
3	14-Mar	Sampling Theory
	17-Mar	Antialiasing Filters
4	21-Mar	Discrete System Analysis
	24-Mar	Convolution Review
	28-Mar	Holiday
	31-Mar	
5	4-Apr	Frequency Response & Filter Analysis
	7-Apr	Filters
6	11-Apr	Digital Filters
	14-Apr	Digital Filters
7	18-Apr	Digital Windows
	21-Apr	FFT
8	25-Apr	Holiday
	28-Apr	Feedback
9	3-May	Introduction to Feedback Control
	5-May	Servoregulation/PID
10	9-May	Introduction to (Digital) Control
	12-May	Digitial Control
11	16-May	Digital Control Design
	19-May	Stability
12	23-May	Digital Control Systems: Shaping the Dynamic Response & Estimation
	26-May	Applications in Industry
13	30-May	System Identification & Information Theory
	2-Jun	Summary and Course Review

Follow Along Reading:



B. P. Lathi
Signal processing and linear systems
1998
TK5102.9.L38 1998

Today & Next Week

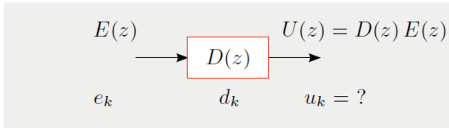
- Chapter 7 (**Frequency Response and Analog Filters**)
  - § 7.5-6 Butterworth/Chebyshev Filters
  - § 7.4 Filter Design
    by Placement of Poles and Zeros of H(s)
  - § 7.7 Frequency Transformations
  - § 7.8 Filters to Satisfy Distortionless Transmission Conditions

- Chapter 10
  (**Discrete-Time System Analysis Using the *z*-Transform**)
  - § 10.3 Properties of DTFT
  - § 10.5 Discrete-Time Linear System analysis by DTFT
  - § 10.7 Generalization of DTFT
    to the $\mathcal{Z}$-Transform

Next, Next Time

Convolution (Quick Point)

Convolution Review



How to calculate u_k from e_k & the pulse response d_k?

- ⋆ Any sequence e_k can be decomposed into a series of pulses, e.g.

  $e_0, e_1, e_2, e_3, \ldots = 1.0, 1.2, 1.3, 0, 0, \ldots$

  $\implies \quad e_k = 1.0\delta_k + 1.2\delta_{k-1} + 1.3\delta_{k-2}$

- ⋆ but

  $e_k = \delta_k$ gives $u_k = d_k =$ system pulse response

  hence

  u_k can be computed by scaling & superimposing system pulse responses

  (because the system is linear)

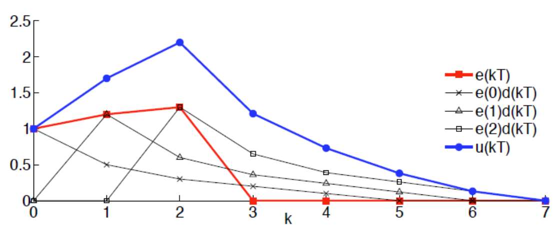
Convolution

Example

System pulse response: $d_0, d_1, d_2, d_3, d_4, d_5 \ldots = 1.0, 0.5, 0.3, 0.2, 0.1, 0, \ldots$

so if $e_0, e_1, e_2, e_3, e_4, \ldots = 1.0, 1.2, 1.3, 0, 0, \ldots$, then

k	0	1	2	3	4	5	6	7
e_k	1	1.2	1.3	0	0	0	0	0
$e_0\,d_k$	1	0.5	0.3	0.2	0.1	0	0	0
$e_1\,d_{k-1}$	0	1.2	0.6	0.36	0.24	0.12	0	0
$e_2\,d_{k-2}$	0	0	1.3	0.65	0.39	0.26	0.13	0
u_k	1	1.7	2.2	1.21	0.73	0.38	0.13	0



Convolution

Equivalently: find u_k from
$$\begin{aligned} U(z) &= D(z)E(z) \\ &= (d_0 + d_1 z^{-1} + \cdots)(e_0 + e_1 z^{-1} + \cdots) \\ &= d_0 e_0 + (e_0 d_1 + e_1 d_0) z^{-1} + \cdots \\ &= u_0 + u_1 z^{-1} + \cdots \end{aligned}$$

$$\Downarrow$$

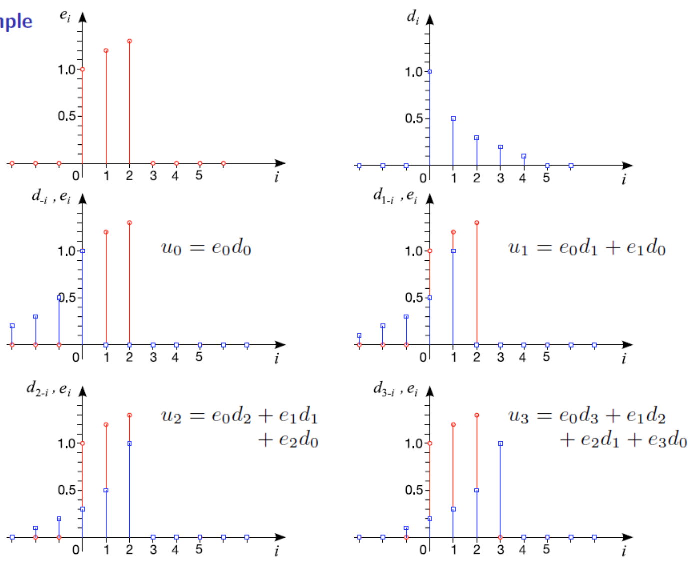
$$\begin{aligned} u_0 &= e_0 d_0 \\ u_1 &= e_0 d_1 + e_1 d_0 \\ u_2 &= e_0 d_2 + e_1 d_1 + e_2 d_0 \\ u_3 &= e_0 d_3 + e_1 d_2 + e_2 d_1 + e_3 d_0 \\ &\vdots \end{aligned}$$

etc.

i.e. $u_k = \sum_{i=0}^{k} e_i \, d_{k-i} = e_k * d_k$

Convolution

Example



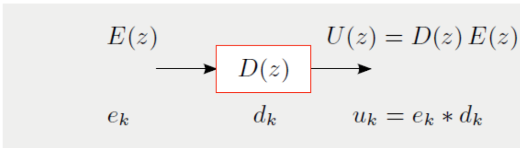
Convolution

The convolution of a pair of sequences e_k and d_k is defined as

$$u_k = \sum_{i=0}^{k} e_i \, d_{k-i} = e_k * d_k$$

So. . .

multiplying the input by the transfer function in the frequency domain
is equivalent to
convolving the input with the pulse response in the time domain



Recall: Graphical Understanding of Convolution

➔ For c(τ)= $(f*g)(t) = \int_{-\infty}^{\infty} f(\tau) g(t-\tau)\, d\tau$:

1. Keep the function $f\,(\tau)$ fixed
2. **Flip** (invert) the function $g(\tau)$ about the vertical axis (τ=0)
   = this is $g(-\tau)$
3. **Shift** this frame ($g(-\tau)$) along τ (horizontal axis) by t_0.
   = this is $g(t_0 - \tau)$

➔ For c(t_0):

4. **c(t_0)** = the area under the product of **$f\,(\tau)$ and $g(t_0 - \tau)$**
5. Repeat this procedure, shifting the frame by different values (positive and negative) to obtain **c(t) for all values of t**.

Another (Better) View

e.g. convolution

$x(n) = 1\ 2\ 3\ 4\ 5$

$h(n) = 3\ 2\ 1$

x(k)	0 0 1 2 3 4 5	0 0 1 2 3 4 5	0 0 1 2 3 4 5	
h(n,k)	1 2 3 0 0 0 0	0 1 2 3 0 0 0	0 0 1 2 3 0 0	h(n-k)
y(n,k)	3	2 6	1 4 9	
y(n)	3	8	14	

Sum over all k

Notice the gain

Matrix Formulation of Convolution

$$\mathbf{y} = \mathbf{Hx}$$

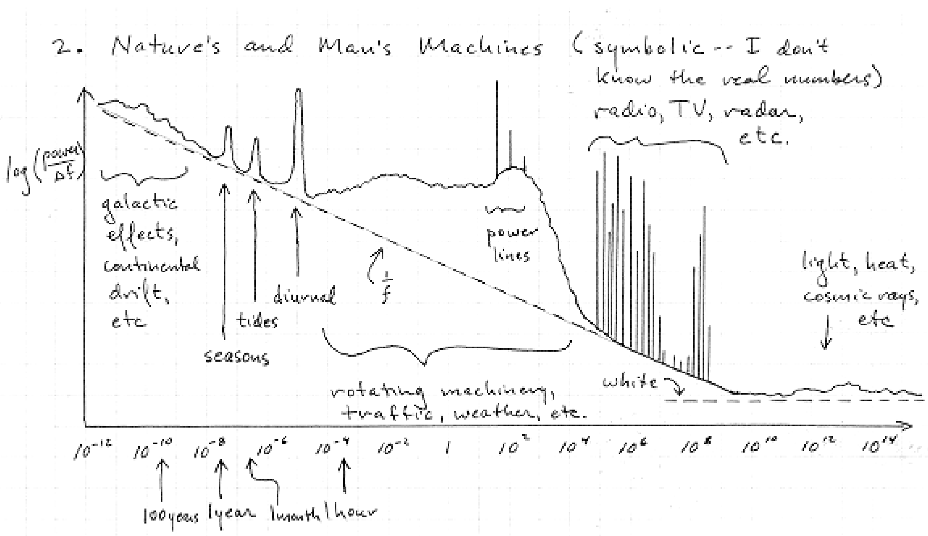
$$\begin{bmatrix} 3 \\ 8 \\ 14 \\ 20 \\ 26 \\ 14 \\ 5 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{bmatrix} . \begin{bmatrix} 0 \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 0 \\ 0 \end{bmatrix}$$

Toeplitz Matrix

Now ..

How to **Beat** the Noise ♫

Remember: There is Noise …

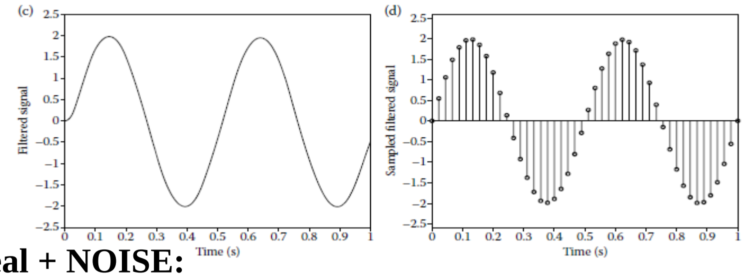


Note: this picture illustrates the concepts but it is not quantitatively precise

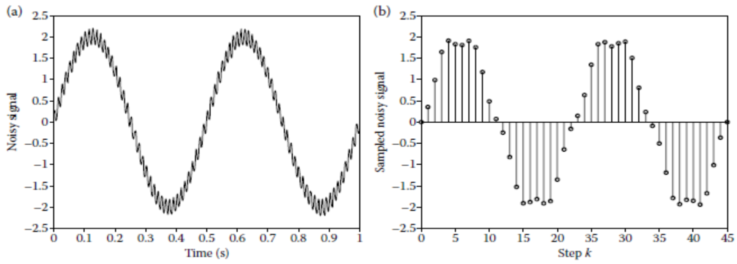
Source: Prof. M. Siegel, CMU

Remember: Effect of Noise…

- Ideal:

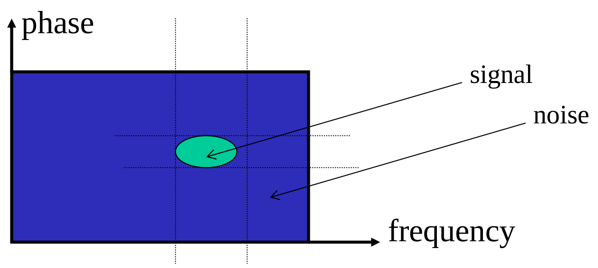


- **Ideal + NOISE:**



How to beat the noise

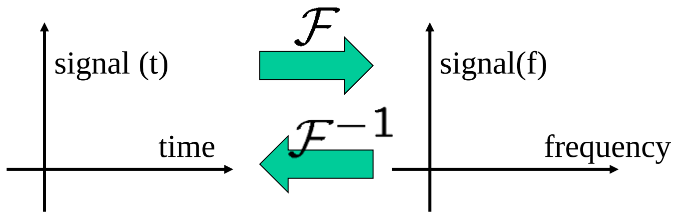
- **Filtering** (Narrow-banding):
  Only look at particular portion of **frequency space**
- Multiple measurements …
- Other (modulation, etc.) …



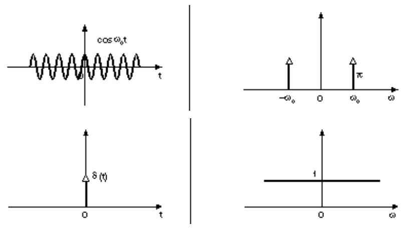
➡ By adding shared **information** (structure) between the sender and receiver (the noise doesn't know your structure)

Frequency

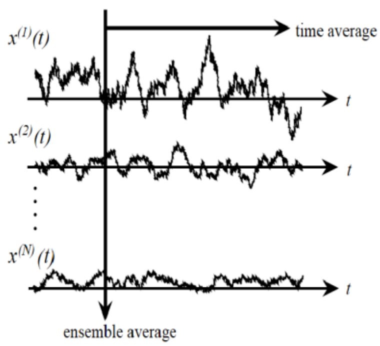
- How often the signal repeats
- Can be analyzed through Fourier Transform



- Examples:



Treating Uncertainty with Multiple Measurements



1. **Over time:** multiple readings of a quantity over time
   - "stationary" or "ergodic" system
   - Sometimes called "integrating"

2. **Over space:** single measurement (summed) from multiple sensors each distributed in space

3. **Same Measurand**: multiple measurements take of the **same observable quantity** by multiple, related instruments

   e.g., measure position & velocity simultaneously

→ Basic "sensor fusion"

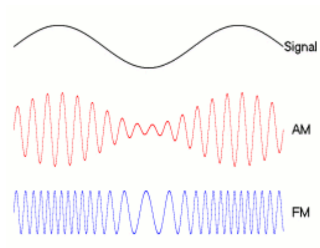
.

$$\sigma_{\text{final}} = \left[\sigma_1^{-1} + \sigma_2^{-1} + \cdots + \sigma_n^{-1}\right]^{-1}$$

Modulation

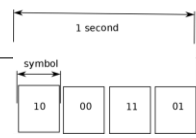
Analog Methods:

- AM - Amplitude modulation
  - Amplitude of a (carrier) is modulated to the (data)
- FM - Frequency modulation
  - Frequency of a (carrier) signal is varied in accordance to the amplitude of the (data) signal
- PM – Phase Modulation



Source: http://en.wikipedia.org/wiki/Modulation

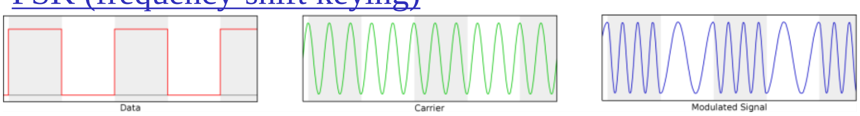
Modulation [Digital Methods]



Start with a "**symbol**" & place it on a channel

- ASK (amplitude-shift keying)

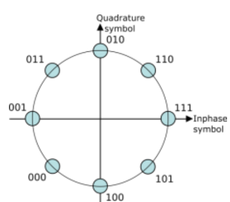
- FSK (frequency-shift keying)



- PSK (phase-shift keying)
- QAM (quadrature amplitude modulation)

$s(t) = A \cdot cos(\omega_c + \phi_i(t))$

$= x_i(t)\cos(\omega_c t) + x_q(t)\sin(\omega_c t)$



Source: http://en.wikipedia.org/wiki/Modulation | http://users.ecs.soton.ac.uk/sqc/EL334 | http://en.wikipedia.org/wiki/Constellation_diagram

Modulation [Example – V.32bis Modem]

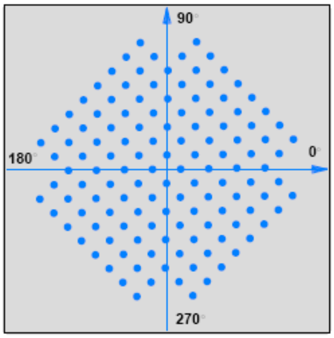
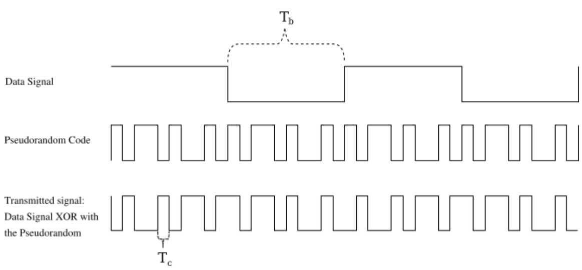


Figure 10.13 Illustration of the QAM constellation for a V.32bis dialup modem.

Source: Computer Networks and Internets, 5e, Douglas E. Comer

Multiple Access (Channel Access Method)

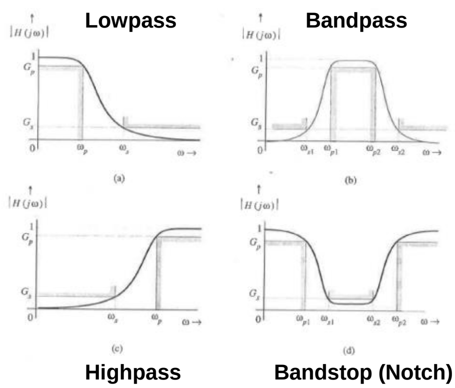
- Send multiple signals on 1 to N channel(s)
  - Frequency-division multiple access (FDMA)
  - Time-division multiple access (TDMA)
  - Code division multiple access (CDMA)
  - Space division multiple access (SDMA)
- CDMA:
  - Start with a pseudorandom code (the noise doesn't know your code)



Source: http://en.wikipedia.org/wiki/Code_division_multiple_access

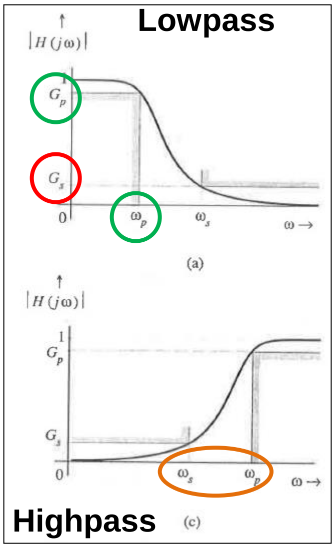
Now: (analog) **Filters!**

Filters



- *Frequency-shaping filters*: LTI systems that change the shape of the spectrum
- *Frequency-selective filters:* Systems that pass some frequencies undistorted and attenuate others

Filters



Specified Values:

- G_p = minimum passband gain

Typically:

$$G_p = \frac{1}{\sqrt{2}} = -3dB$$

- G_s = maximum stopband gain
  - **Low**, not zero (sorry!)
  - For realizable filters, the gain cannot be zero over a finite band (Paley-Wiener condition)
- Transition Band:

  transition from the passband to the stopband ➔ $\omega_p \neq \omega_s$

Filter Design & z-Transform

Filter Type	Mapping	Design Parameters
Low-pass	$z^{-1} \to \frac{z^{-1} - \alpha}{1 - \alpha z^{-1}}$	$\alpha = \frac{\sin[(\omega_c - \omega_c')/2]}{\sin[(\omega_c + \omega_c')/2]}$ ω_c' = desired cutoff frequency
High-pass	$z^{-1} \to -\frac{z^{-1} + \alpha}{1 + \alpha z^{-1}}$	$\alpha = -\frac{\cos[(\omega_c + \omega_c')/2]}{\cos[(\omega_c - \omega_c')/2]}$ ω_c' = desired cutoff frequency
Bandpass	$z^{-1} \to -\frac{z^{-2} - [2\alpha\beta/(\beta+1)]z^{-1} + [(\beta-1)/(\beta+1)]}{[(\beta-1)/(\beta+1)]z^{-2} - [2\alpha\beta/(\beta+1)]z^{-1} + 1}$	$\alpha = \frac{\cos[(\omega_{c2} + \omega_{c1})/2]}{\cos[(\omega_{c2} - \omega_{c1})/2]}$ $\beta = \cot[(\omega_{c2} - \omega_{c1})/2]\tan(\omega_c/2)$ ω_{c1} = desired lower cutoff frequency ω_{c2} = desired upper cutoff frequency
Bandstop	$z^{-1} \to \frac{z^{-2} - [2\alpha/(\beta+1)]z^{-1} + [(1-\beta)/(1+\beta)]}{[(1-\beta)/(1+\beta)]z^{-2} - [2\alpha/(\beta+1)]z^{-1} + 1}$	$\alpha = \frac{\cos[(\omega_{c1} + \omega_{c2})/2]}{\cos[(\omega_{c1} - \omega_{c2})/2]}$ $\beta = \tan[(\omega_{c2} - \omega_{c1})/2]\tan(\omega_c/2)$ ω_{c1} = desired lower cutoff frequency ω_{c2} = desired upper cutoff frequency

Butterworth Filters

- Butterworth: Smooth in the pass-band
- The amplitude response $|H(j\omega)|$ of an n[th] order Butterworth low pass filter is given by:

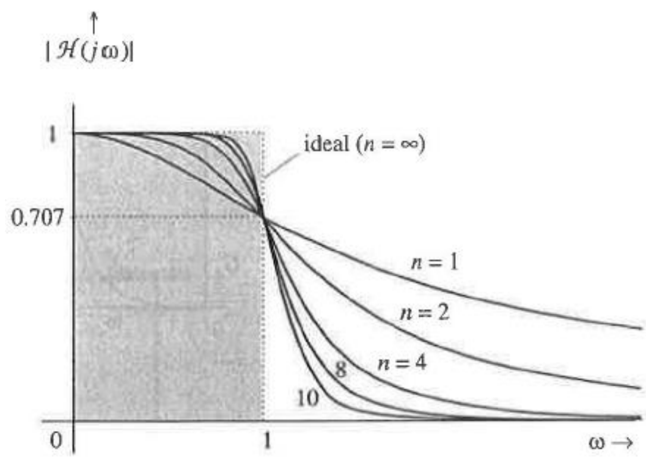
$$|H(j\omega)| = \frac{1}{\sqrt{1+\left(\frac{\omega}{\omega_c}\right)^{2n}}}$$

- The normalized case (ω_c=1)

$$|\mathcal{H}(j\omega)| = \frac{1}{\sqrt{1+\omega^{2n}}} \Rightarrow \mathcal{H}(j\omega)\mathcal{H}(-j\omega) = |\mathcal{H}(j\omega)|^2 = \frac{1}{1+\omega^{2n}}$$

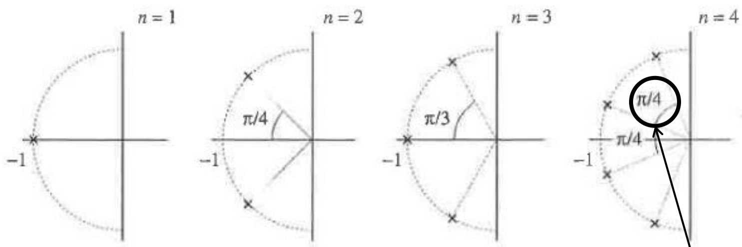
Recall that: $|H(j\omega)|^2 = H(j\omega)\, H(-j\omega)$

Butterworth Filters



Butterworth Filters of Increasing Order: Seeing this Using a Pole-Zero Diagram

- Increasing the order, increases the number of poles:



➔Odd orders (n=1,3,5…):

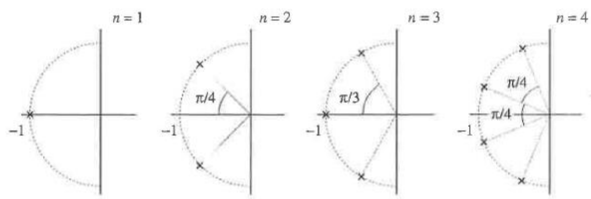
- Have a pole on the Real Axis

➔Even orders (n=2,4,6…):

- Have a pole on the off axis

Angle between poles: $\frac{\pi}{n}$

Butterworth Filters: Pole-Zero Diagram



- Since H(s) is stable and causal, its poles must lie in the LHP
- Poles of -H(s) are those in the RHP
- Poles lie on the unit circle (for a normalized filter)

n is the order of the filter

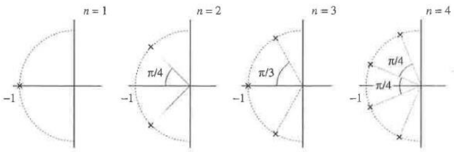
➔$H(s) = \dfrac{1}{(s - s_1)(s - s_2)\ldots(s - s_n)}$

Where:

$$s_k = e^{\frac{j\pi}{2n}(2k+n-1)}$$

$$= \cos\frac{\pi}{2n}(2k+n-1) + j\sin\frac{\pi}{2n}(2k+n-1) \qquad k = 1, 2, 3, \ldots, n$$

Butterworth Filters: 4th Order Filter Example



- Plugging in for n=4, k=1,…4:

$$\mathcal{H}(s) = \frac{1}{(s+0.3827-j0.9239)(s+0.3827+j0.9239)(s+0.9239-j0.3827)(s+0.9239+j0.3827)}$$

$$= \frac{1}{(s^2+0.7654s+1)(s^2+1.8478s+1)}$$

$$= \frac{1}{s^4+2.6131s^3+3.4142s^2+2.6131s+1}$$

- We can generalize ➔ Butterworth Table

n	a_1	a_2	a_3	a_4	a_5
2	1.41421356				
3	2.00000000	2.00000000			
4	2.61312593	3.41421356	2.61312593		
5	3.23606798	5.23606798	5.23606798	3.23606798	
6	3.86370331	7.46410162	9.14162017	7.46410162	3.86370331

This is for 3dB bandwidth at ω_c=1

Butterworth Filters: Scaling Back (from Normalized)

- Start with Normalized equation & Table
- Replace ω with $\frac{\omega}{\omega_c}$ in the filter equation

- For example:
  for f_c=100Hz → ω_c=200π rad/sec

  From the Butterworth table: for n=2, $a_1=\sqrt{2}$
  Thus:

$$H(s) = \frac{1}{\left(\frac{s}{200\pi}\right)^2+\sqrt{2}\left(\frac{s}{200\pi}\right)+1}$$

$$= \frac{1}{s^2+200\pi\sqrt{2}+40{,}000\pi^2}$$

Butterworth: Determination of Filter Order

- Define $\mathbf{G_x}$ as the gain of a lowpass Butterworth filter at $\omega = \omega_x$
- Then:

$$\hat{G}_x = 20\log_{10}|H(j\omega_x)| = -10\log\left[1+\left(\frac{\omega_x}{\omega_c}\right)^{2n}\right]$$

And thus:

$$\hat{G}_p = -10\log\left[1+\left(\frac{\omega_p}{\omega_c}\right)^{2n}\right]$$

$$\hat{G}_s = -10\log\left[1+\left(\frac{\omega_s}{\omega_c}\right)^{2n}\right]$$

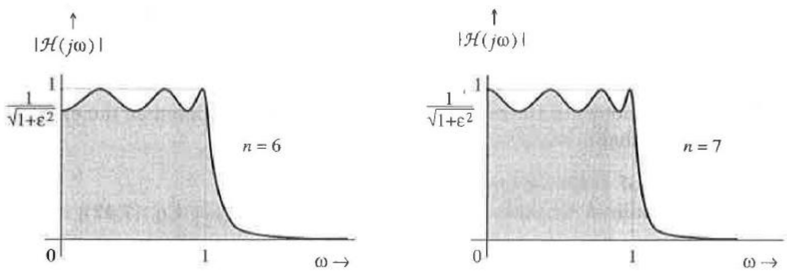
Or alternatively: $\omega_c = \dfrac{\omega_p}{\left[10^{-\hat{G}_p/10} - 1\right]^{1/2n}}$ & $\omega_c = \dfrac{\omega_s}{\left[10^{-\hat{G}_s/10} - 1\right]^{1/2n}}$

Solving for n gives:

$$n = \frac{\log\left[\left(10^{-\hat{G}_s/10} - 1\right) / \left(10^{-\hat{G}_p/10} - 1\right)\right]}{2\log(\omega_s/\omega_p)}$$

PS. See Lathi 4.10 (p. 453) for an example in MATLAB

Chebyshev Filters



- **equal-ripple:**
  Because all the ripples in the passband are of equal height
- If we reduce the ripple, the passband behaviour improves, but it does so at the cost of stopband behaviour

Chebyshev Filters

- Chebyshev Filters: Provide tighter transition bands (sharper cutoff) than the same-order Butterworth filter, but this is achieved at the expense of inferior passband behavior (rippling)

➔ For the lowpass (LP) case: at higher frequencies (in the stopband), the Chebyshev filter gain is smaller than the comparable Butterworth filter gain by about ***6(n - 1) dB***

- The amplitude response of a normalized Chebyshev lowpass filter is:

$$|\mathcal{H}(j\omega)| = \frac{1}{\sqrt{1+\epsilon^2 {C_n}^2(\omega)}}$$

Where Cn(ω), the nth-order Chebyshev polynomial, is given by:

$$C_n(\omega) = \cos\left(n \cos^{-1} \omega\right)$$

$$C_n(\omega) = \cosh\left(n \cosh^{-1} \omega\right)$$

and where C_n is given by:

n	$C_n(\omega)$
0	1
1	ω
2	$2\omega^2 - 1$
3	$4\omega^3 - 3\omega$
4	$8\omega^4 - 8\omega^2 + 1$
5	$16\omega^5 - 20\omega^3 + 5\omega$
6	$32\omega^6 - 48\omega^4 + 18\omega^2 - 1$

Normalized Chebyshev Properties

- It's normalized: The passband is 0<ω<1
- **Amplitude response**: has **ripples** in the passband and is **smooth** (monotonic) in the stopband
- **Number of ripples**: there is a total of ***n*** maxima and minima over the passband 0<ω<1
- $C_n^2(0) = \begin{cases} 0, & n : odd \\ 1, & n : even \end{cases}$ ➔ $|H(0)| = \begin{cases} 1, & n : odd \\ \frac{1}{\sqrt{1+\epsilon^2}}, & n : even \end{cases}$
- ϵ: ripple height → $r = \sqrt{1+\epsilon^2}$
- The Amplitude at ω=1: $\frac{1}{r} = \frac{1}{\sqrt{1+\epsilon^2}}$
- For Chebyshev filters, the ripple ***r*** dB takes the place of G_p

Determination of Filter Order

- The gain is given by: $\hat{G} = -10 \log \left[1 + \epsilon^2 {C_n}^2(\omega)\right]$

Thus, the gain at ω_s is: $\epsilon^2 {C_n}^2(\omega_s) = 10^{-\hat{G}_s/10} - 1$

- Solving:

$$n = \frac{1}{\cosh^{-1}(\omega_s)} \cosh^{-1} \left[\frac{10^{-\hat{G}_s/10} - 1}{10^{\hat{r}/10} - 1}\right]^{1/2}$$

- General Case:

$$n = \frac{1}{\cosh^{-1}(\omega_s/\omega_p)} \cosh^{-1} \left[\frac{10^{-\hat{G}_s/10} - 1}{10^{\hat{r}/10} - 1}\right]^{1/2}$$

Chebyshev Pole Zero Diagram

- Whereas Butterworth poles lie on a semi-circle,
  The poles of an n^{th}-order normalized Chebyshev filter lie on a **semiellipse** of the major and minor semiaxes:

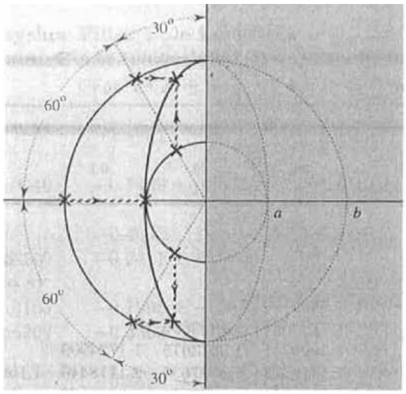
$$a = \sinh\left(\frac{1}{n}\sinh^{-1}\left(\frac{1}{\epsilon}\right)\right) \quad \& \quad b = \cosh\left(\frac{1}{n}\sinh^{-1}\left(\frac{1}{\epsilon}\right)\right)$$

And the poles are at the locations:

$$H(s) = \frac{1}{(s - s_1)(s - s_2)\ldots(s - s_n)}$$

$$s_k = -\sin\left[\frac{(2k-1)\pi}{2n}\right]\sinh x + j\cos\left[\frac{(2k-1)\pi}{2n}\right]\cosh x, \; k = 1, \ldots, n$$

Ex: Chebyshev Pole Zero Diagram for **n=3**



Procedure:

1. Draw two semicircles of radii **a** and **b** (from the previous slide).
2. Draw radial lines along the corresponding Butterworth angles (π/n) and locate the n^{th}-order Butterworth poles (shown by crosses) on the two circles.
3. The location of the k^{th} Chebyshev pole is the intersection of the horizontal projection and the vertical projection from the corresponding k^{th} Butterworth poles on the outer and the inner circle, respectively.

Chebyshev Values / Table

$$\mathcal{H}(s) = \frac{K_n}{C'_n(s)} = \frac{K_n}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0}$$

$$K_n = \begin{cases} a_0 & n \text{ odd} \\ \dfrac{a_0}{\sqrt{1+\epsilon^2}} = \dfrac{a_0}{10^{\hat{r}/20}} & n \text{ even} \end{cases}$$

n	a_0	a_1	a_2	a_3
1	1.9652267			
2	1.1025103	1.0977343		
3	0.4913067	1.2384092	0.9883412	
4	0.2756276	0.7426194	1.4539248	0.9528114

1 db ripple
$(\hat{r} = 1)$

Other Filter Types:
Chebyshev Type II = Inverse Chebyshev Filters

- Chebyshev filters passband has ripples and the stopband is smooth.
- **Instead:** this has **passband** have **smooth** response and **ripples** in the stopband.

➔ Exhibits maximally flat passband response and equi-ripple stopband

➔ `Cheby2` **in MATLAB**

$$|\mathcal{H}(\omega)|^2 = 1 - |\mathcal{H}_C(1/\omega)|^2 = \frac{\epsilon^2 C_n^2(1/\omega)}{1+\epsilon^2 C_n^2(1/\omega)}$$

Where: H_c is the Chebyshev filter system from before

- Passband behavior, especially for small ω, is **better** than Chebyshev
- **Smallest transition band** of the 3 filters (Butter, Cheby, Cheby2)
- Less time-delay (or phase loss) than that of the **Chebyshev**
- Both needs the **same order *n*** to meet a set of specifications.
- **$$$** (or number of elements):
  Cheby < Inverse Chebyshev < Butterworth (of the same **performance** [not order])

Other Filter Types:
Elliptic Filters (or Cauer) Filters

- Allow **ripple** in **both** the passband and the stopband,
  ➔ we can achieve **tighter** transition band

$$|\mathcal{H}(j\omega)| = \frac{1}{\sqrt{1+\epsilon^2 {R_n}^2(\omega)}}$$

Where: R_n is the n^{th}-order Chebyshev rational function determined from a given ripple spec.
ϵ controls the ripple
$Gp = \frac{1}{\sqrt{1+\varepsilon^2}}$

- Most efficient (η)
  - the **largest ratio** of the passband gain to stopband gain
  - **or** for a given ratio of passband to stopband gain, it requires the **smallest transition band**

➔ **in MATLAB:** `ellipord` followed by `ellip`

In Summary

Filter Type	Passband Ripple	Stopband Ripple	Transition Band	MATLAB Design Command
Butterworth	No	No	Loose	`butter`
Chebyshev	Yes	No	Tight	`cheby`
Chebyshev Type II (Inverse Chebyshev)	No	Yes	Tight	`cheby2`
Eliptic	Yes	Yes	Tightest	`ellip`

Next Time…

- **Digital Filters**

- Review:
  - Chapter 7 of Lathi

- FIR and IIR: Might finite > infinitie ?