11/23/20

Today's Lecture

* IIR Filter Design Techniques (Digital)

    - Prony's method

    - Frequency sampling design

* Digital IIR Design From Analog IIR Filters

* Matlab examples

Last Time

- Prony's method

$$H(z) = \frac{b_0 + b_1 z^{-1} + \dots + b_M z^{-M}}{1 + a_1 z^{-1} + \dots + a_N z^{-N}} = \frac{B(z)}{A(z)} = H(z)$$

Goal : $H_d(z) A(z) = B(z)$

$$(h_d[0] + h_d[1] z^{-1} + \dots)(1 + a_1 z^{-1} + \dots + a_N z^{-N})$$
$$= (b_0 + b_1 z^{-1} + \dots + b_M z^{-M})$$

$$\underbrace{\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_M \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{K+1 \times 1} = \underbrace{\begin{bmatrix} h_d[0] & 0 & \dots & & 0 \\ h_d[1] & h_d[0] & 0 & \dots & 0 \\ h_d[2] & h_d[1] & h_d[0] & \dots & 0 \\ \vdots & & & & \vdots \\ h_d[M] & h_d[M-1] & h_d[M-2] & \dots & h_d[M-N] \\ \vdots & & & & \\ h_d[K] & h_d[K-1] & h_d[K-2] & \dots & h_d[K-N] \end{bmatrix}}_{K+1 \times N+1} \underbrace{\begin{bmatrix} 1 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix}}_{(N+1) \times 1}$$

Let $\underline{b} = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ b_M \end{bmatrix}$, $\underline{a} = \begin{bmatrix} 1 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix}$ and $\underline{a}^* = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix}$, $\underline{0} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$

$$\begin{bmatrix} \underline{b} \\ \underline{0} \end{bmatrix} = \left[\begin{array}{c|c} \multicolumn{2}{c}{H_1} \\ \hline \underline{h_1} & H_2 \end{array}\right] \begin{bmatrix} 1 \\ \underline{a}^* \end{bmatrix}$$

where $H_1 = (M+1) \times (N+1)$ matrix

$h_1 : (K-M) \times 1$ vector

$H_2 : (K-M) \times N$ matrix

$\underline{b} = H_1 \cdot \underline{a} \rightarrow M+1$ equations

$\underline{0} = \underline{h_1} + H_2 \underline{a}^* \rightarrow K-M$ equations

$\Rightarrow H_2 \underline{a}^* = -\underline{h_1}$

- if $K-M=N$, then H_2 is $N \times N$ and $\underline{a}^* = -H_2^{-1} h_1$

- if $K-M>N$, then H_2 is tall and $\underline{a}^* = -\underbrace{(H_2^T H_2)^{-1} H_2^T}_{\text{pseudoinverse}} h_1$

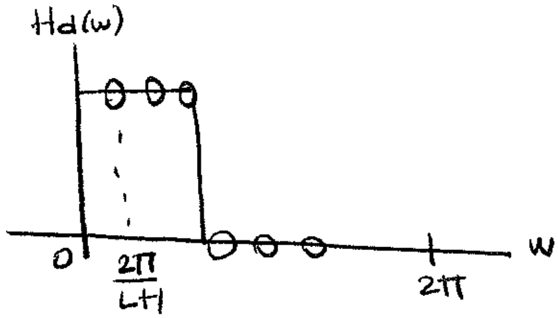
$$\underline{b} = H_1 \begin{bmatrix} 1 \\ \underline{a}^* \end{bmatrix}$$

– Frequency Sampling Design of IIR Filters

$H_d(\omega)$: desired frequency response

Find a, b values that satisfies $\quad H_d(z) = \dfrac{B(z)}{A(z)}$ $\longrightarrow$ $M+1$ unknowns (for $B(z)$), N unknowns (for $A(z)$)

$M + N = L$



Sample $H_d(\omega)$ at $\dfrac{2\pi k}{L+1}$

Length $L+1$ DFT.

$$H_d[k] = \underbrace{\frac{\mathrm{DFT}(b[n])}{\mathrm{DFT}(a[n])}}_{\text{Element by element division}} = \frac{B[k]}{A[k]} \implies B[k] = A[k]H_d[k]$$

In time domain, this corresponds to circular convolution :

$$\underline{b} = \underline{a} \circledast \underline{h_d}.$$

$$\underbrace{\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_M \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{L+1 \times 1} = \begin{bmatrix} g_0 & g_L & g_{L-1} & \cdots & g_1 \\ g_1 & g_0 & g_L & \cdots & g_2 \\ g_2 & g_1 & g_0 & \cdots & g_3 \\ \vdots & & & & \vdots \\ g_{L-2} & g_{L-3} & g_{L-4} & \cdots & g_{L-1} \\ g_{L-1} & g_{L-2} & g_{L-3} & \cdots & g_L \\ g_L & g_{L-1} & g_{L-2} & \cdots & g_0 \end{bmatrix} \underbrace{\begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_N \\ 0 \\ \vdots \\ 0 \end{bmatrix}}_{L+1 \times 1}$$

$$g_k = \mathrm{IDFT}\left(H_d\left(\frac{2\pi k}{L+1}\right)\right)$$ (Similar to Prony's method)

$L = N+M$

$$\begin{bmatrix} \underline{b} \\ \underline{0} \end{bmatrix} = \begin{bmatrix} \multicolumn{2}{c}{G_1} \\ \hline \underline{g}_1 & G_2 \end{bmatrix} \underbrace{\begin{bmatrix} 1 \\ \underline{a}^* \end{bmatrix}}_{\underline{a}}$$

where G_1: $(M+1) \times (L+1)$ matrix

$\underline{g}_1$: $N \times 1$ vector

G_2: $N \times L$ matrix

where $L = N+M$

$$\underline{b} = G_1 \underline{a}$$

$$\underline{0} = \underline{g}_1 + G_2 \underline{a}^* \quad \text{where} \quad \underline{a}^* = \begin{bmatrix} a_1 \\ \vdots \\ a_N \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\Rightarrow G_2 \underline{a}^* = -\underline{g}_1$$ (Same structure as Prony's method)

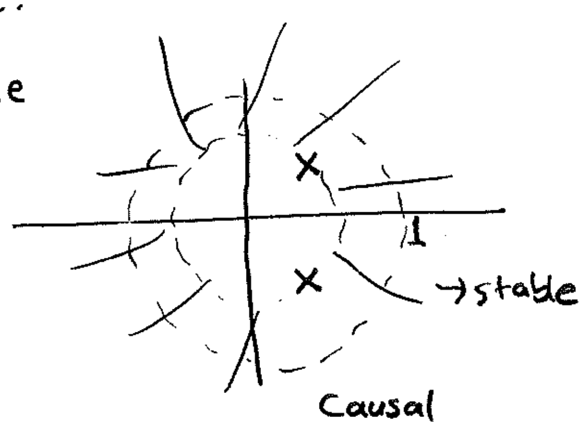
* Least squares approximation (similar to Prony's method). This time more frequency samples than the coefficients:

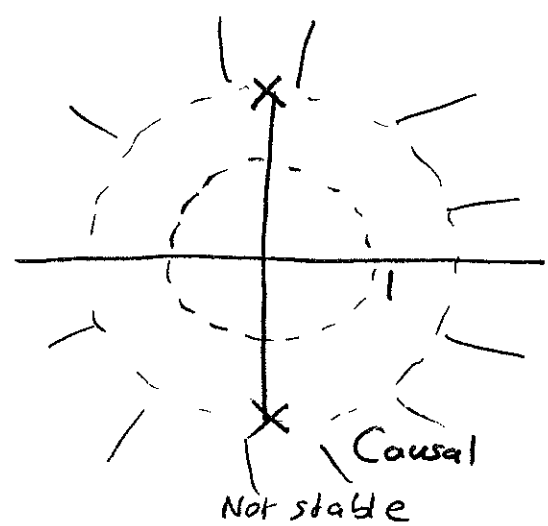
$$L+1 = M+N+1$$

* Like FIR, this is an interpolation method. We do not have any control over the sample points in between.

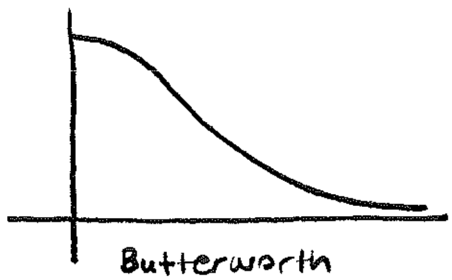
* Unlike FIR, we cannot guarantee that the designed filters are stable.

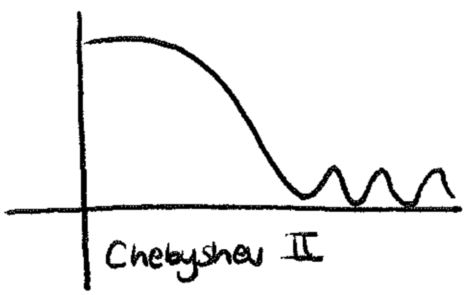
Example

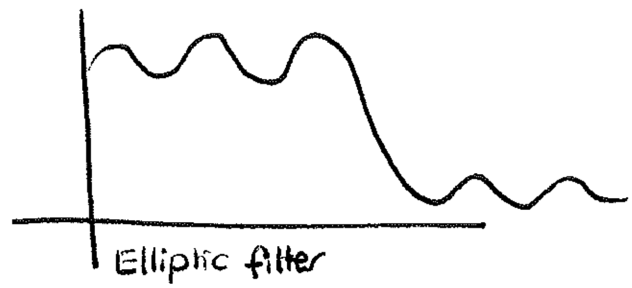




* We can design digital IIR filters from analog IIR filters.

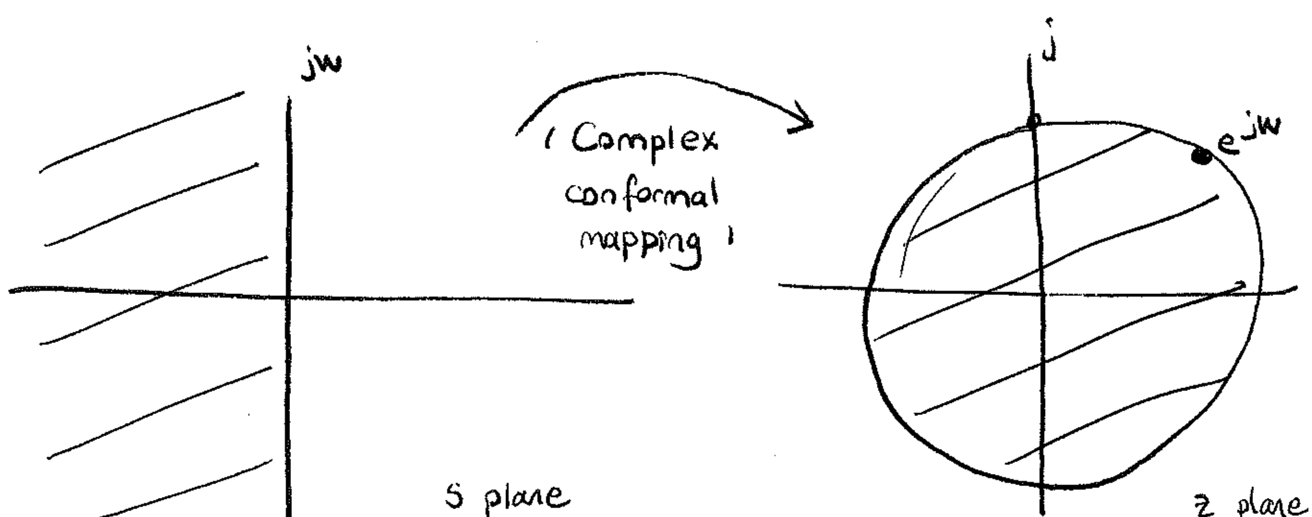






Analog domain $\Rightarrow$ Digital

Continuous Time		Discrete Time
$H_c(s)$	$\xrightarrow{T}$	$H(z)$
$h_c(t)$		$h[n]$

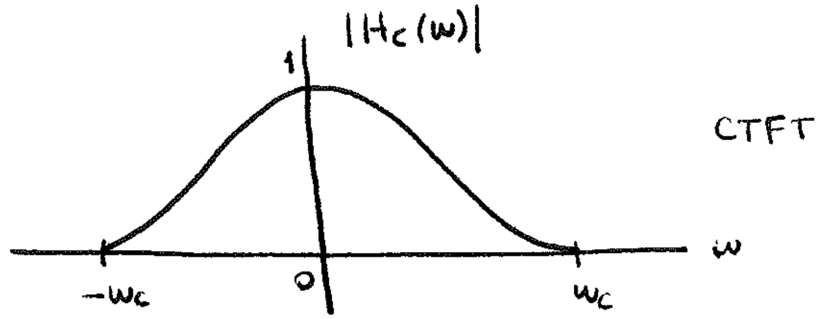


Stability

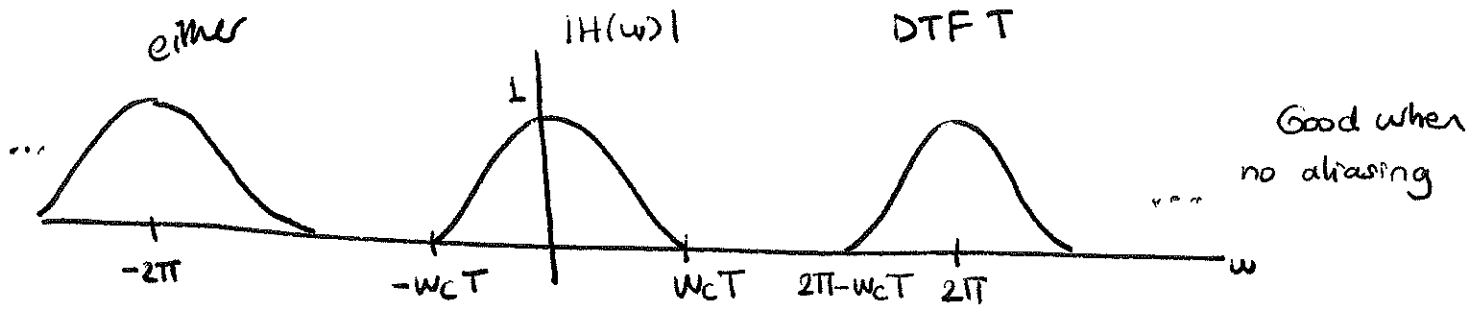
Two main approaches to Digital IIR Design from Analog IIR.

1. Impulse invariance

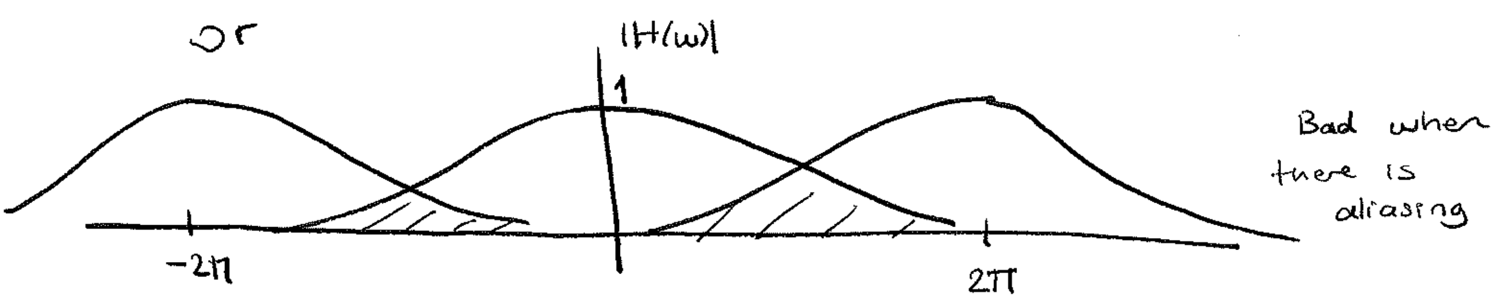
Given $h_c(t)$ create $h[n] = T h_c(nT)$ where T is the sampling period.



either



or



Note: Convolution of two sampled signals $h_1[n] = h_1(nT)$ and $h_2[n] = h_2(nT)$ is not the same as the sampled convolution of $\left(h_1(t) * h_2(t)\right)\Big|_{t=nT}$

$$H(w) = \sum_{k=-\infty}^{+\infty} H_c\left(\frac{w}{T} - k\frac{2\pi}{T}\right)$$ → sum of shifted copies of the frequency response of the continuous time system.

- If $w_c < \frac{w_s}{2}$ then $H(w) = H_c\left(\frac{w}{T}\right)$ for $|w| < w_s$

2. Bilinear transformation

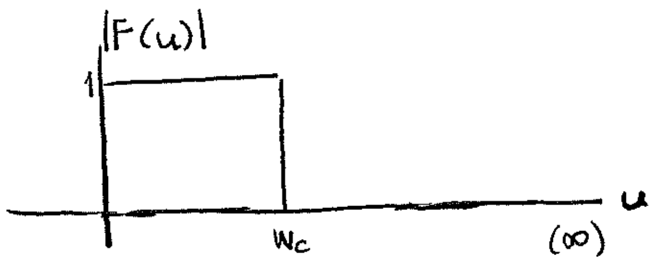
Consider a transformation of the s-plane into the z-plane.

$$s = \frac{2}{T}\,\frac{z-1}{z+1} \longrightarrow z = \frac{\frac{2}{T}+s}{\frac{2}{T}-s}$$

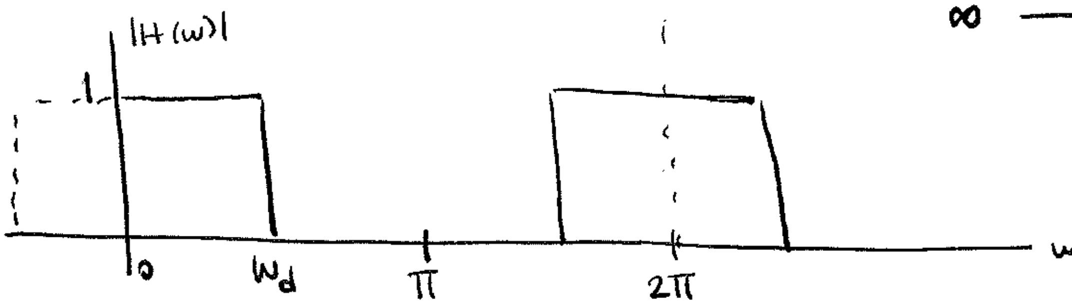
s	z	$w \in [0, 2\pi]$
0	1	0
$\pm\infty$	-1	π
$\frac{2}{T}j$	j	$\frac{\pi}{2}$
$-\frac{2}{T}$	0	NA.

Verify!

Analog filter



Digital filter

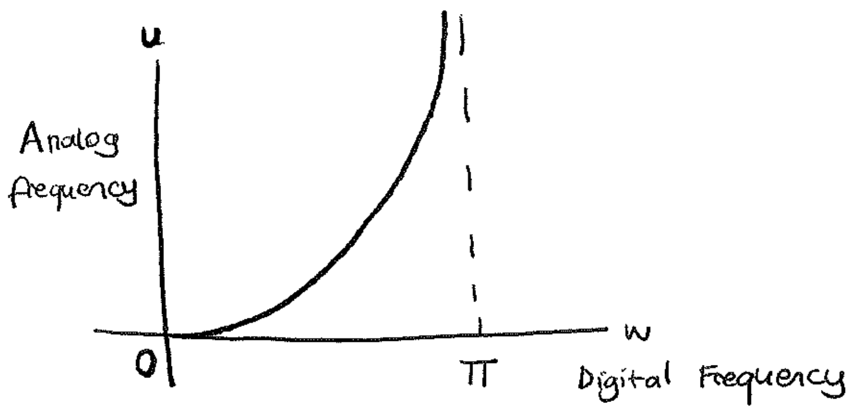


$w_c \longrightarrow w_d$

$\infty \longrightarrow \pi$

Steps to implement:

1. Prewarp the desired digital filter cutoff frequencies to corresponding cutoff frequencies in analog domain.
2. Design analog filter
3. Warp back to digital filter with bilinear transformation.



$$u = \frac{2}{T} \tan\left(\frac{w}{2}\right)$$

$$w = 2\arctan\left(\frac{T}{2} u\right)$$

Nonlinear warping of frequency axis

MATLAB tools:

- fdatool (FIR, IIR filters with desired specifications)
- fvtool
- sptool
- filter builder (quick filter design)