ESE 5310: Digital Signal Processing

Lecture 9: February 20, 2024

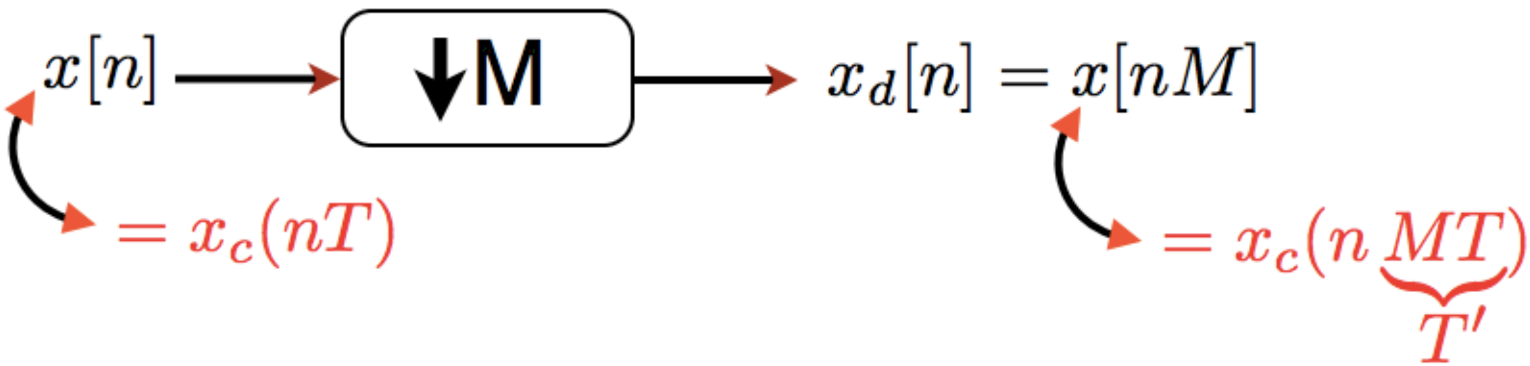
Non-Integer and Multi-rate Sampling

Lecture Outline

- Review: Downsampling/Upsampling
- Non-integer Resampling
- Multi-Rate Processing
  - Interchanging Operations
  - Filter cost

Downsampling

- Definition: Reducing the sampling rate by an integer number

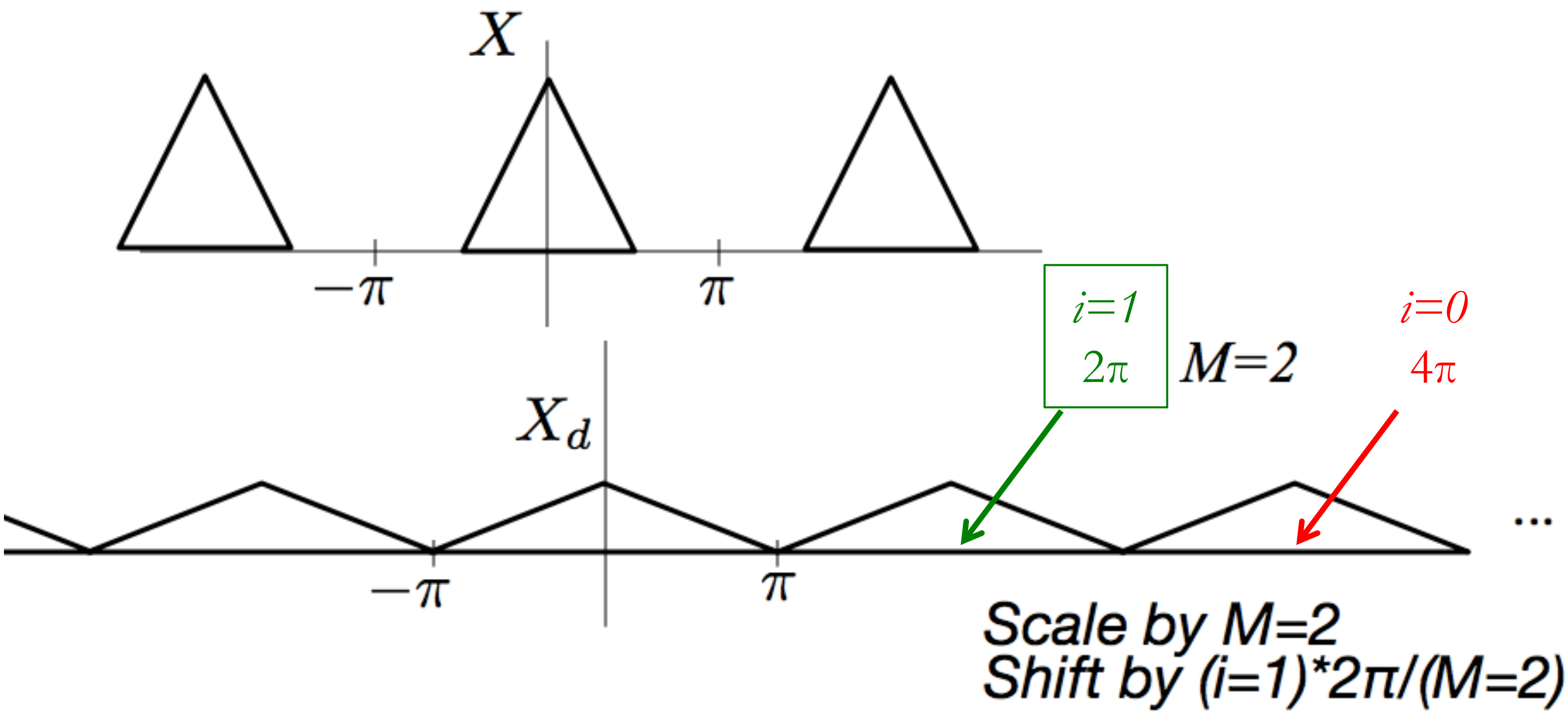


$$X_d(e^{j\omega}) = \frac{1}{M}\sum_{i=0}^{M-1} X\left(e^{j\left(\frac{\omega}{M} - \frac{2\pi}{M}i\right)}\right)$$

stretch by M — replicate

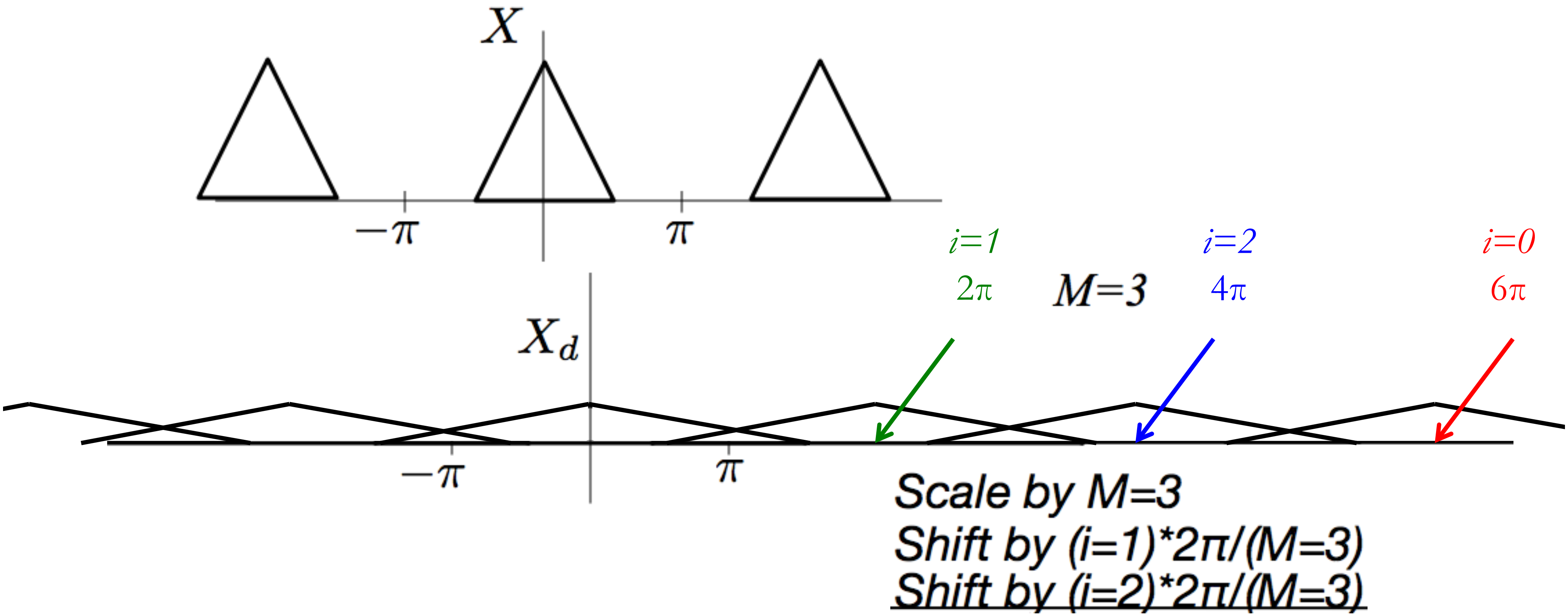
Example

$$X_d(e^{j\omega}) = \frac{1}{M}\sum_{i=0}^{M-1} X\left(e^{j\left(\frac{w}{M} - \frac{2\pi}{M}i\right)}\right)$$

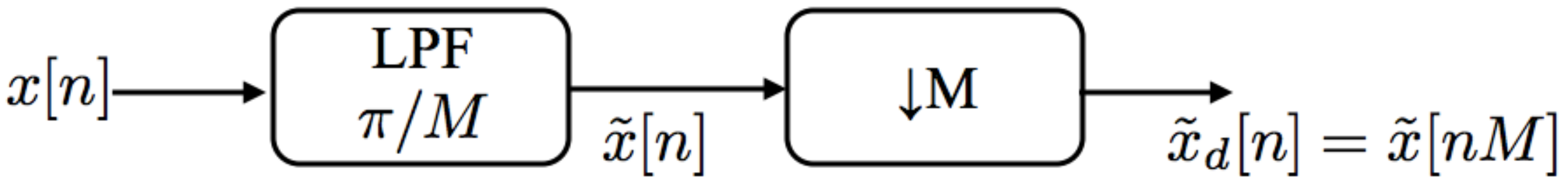


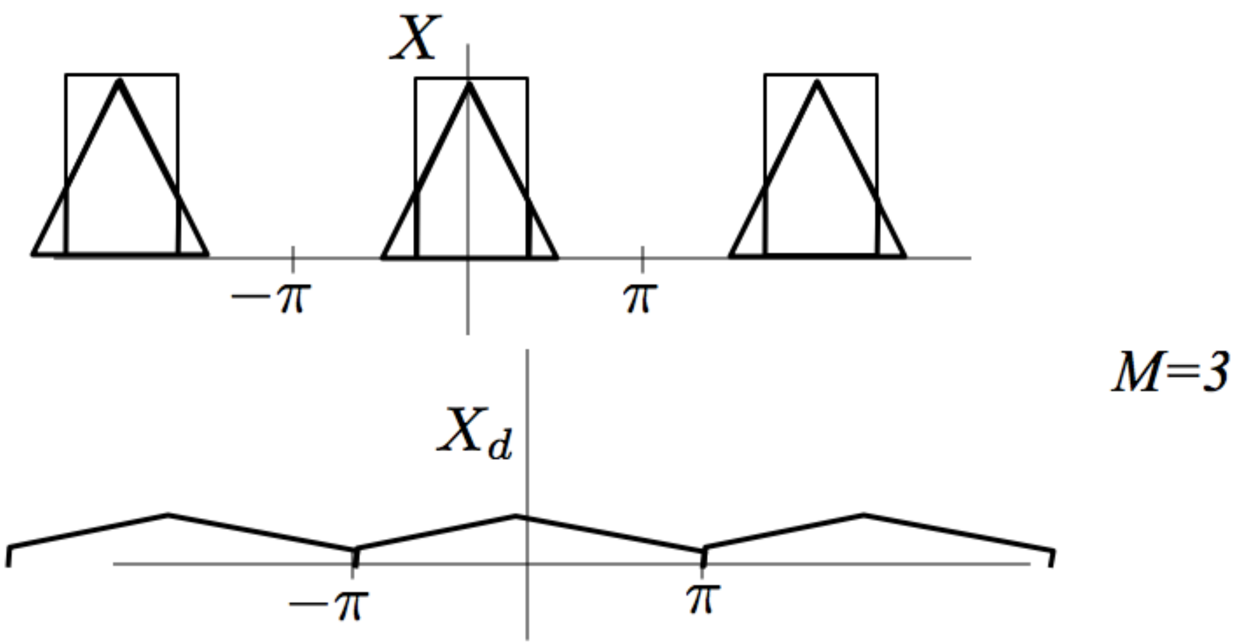
Example

$$X_d(e^{j\omega}) = \frac{1}{M}\sum_{i=0}^{M-1} X\left(e^{j\left(\frac{w}{M} - \frac{2\pi}{M}i\right)}\right)$$



Example


$x[n]$
LPF
π/M
$\tilde{x}[n]$
$\downarrow$M
$\tilde{x}_d[n] = \tilde{x}[nM]$


X
$-\pi$
π
X_d
$-\pi$
π
M=3

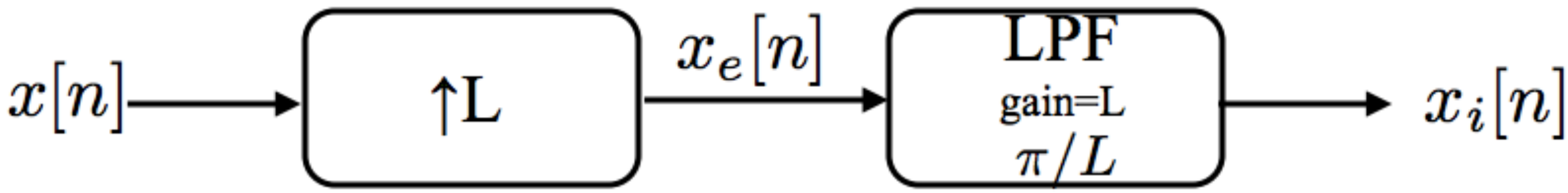
Upsampling

- Definition: Increasing the sampling rate by an integer number

$$x[n] = x_c(nT)$$

$$x_i[n] = x_c(nT') \quad \text{where } T' = \frac{T}{L} \qquad L \text{ integer}$$

Frequency Domain Interpretation

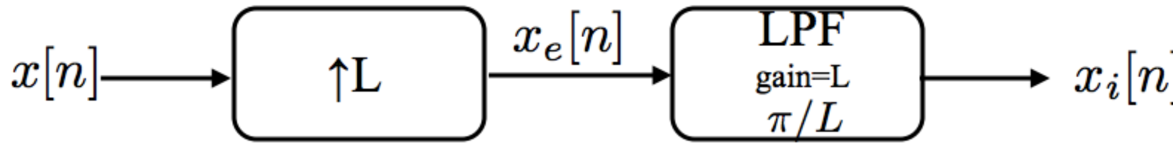


$$X_e(e^{j\omega}) = \sum_{n=-\infty}^{\infty} \underbrace{x_e[n]}_{\neq 0 \text{ only for } n=mL \text{ (integer m)}} e^{-j\omega n}$$

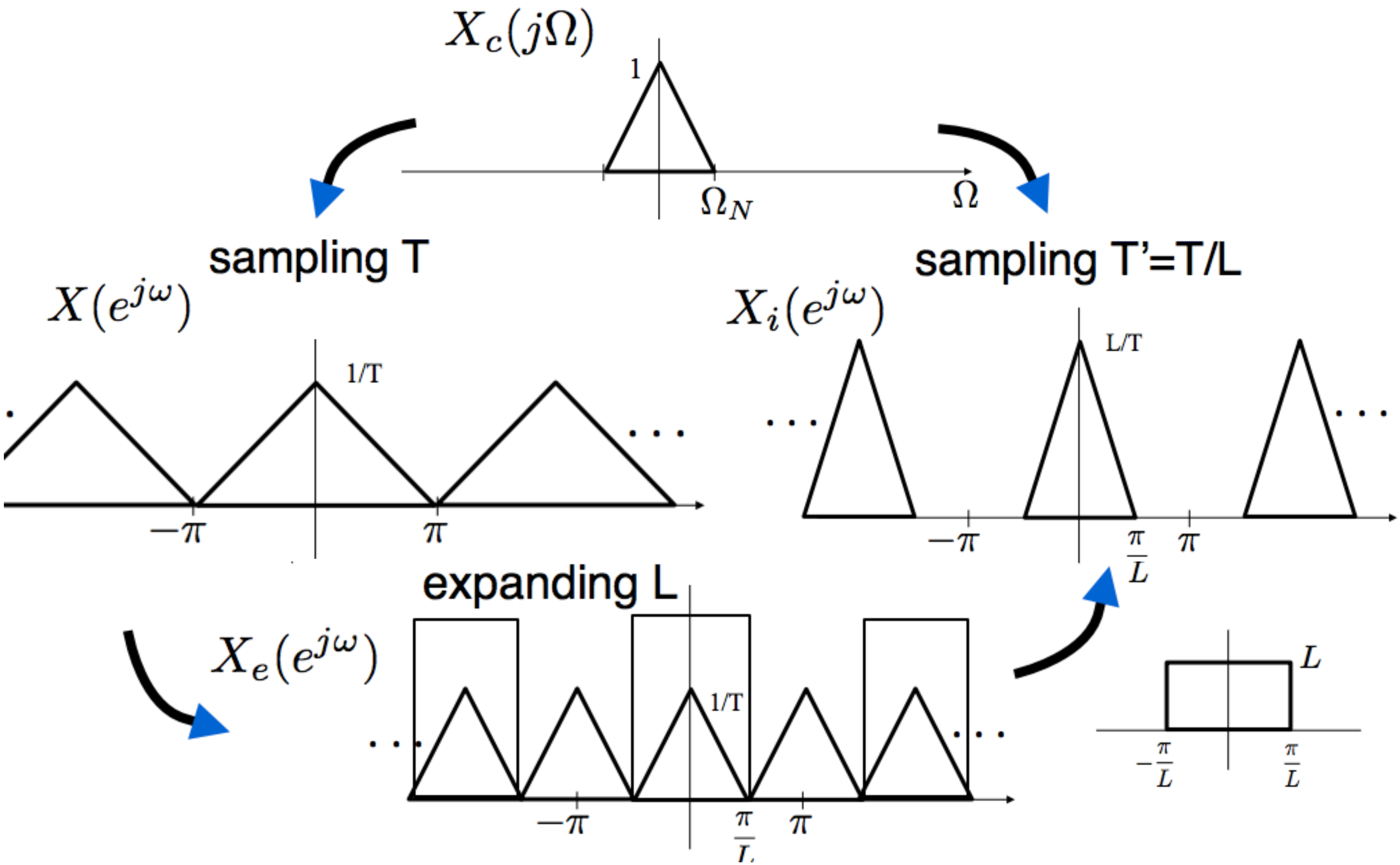
$$= \sum_{m=-\infty}^{\infty} \underbrace{x_e[mL]}_{=x[m]} e^{-j\omega mL} = \boxed{X(e^{j\omega L})}$$

Compress DTFT by a factor of L!

Example



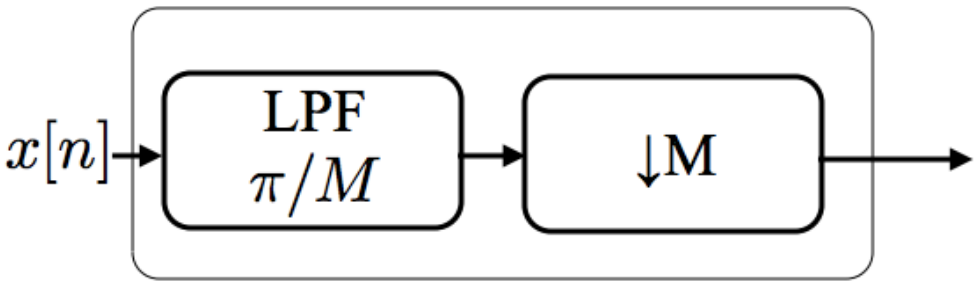
$x[n]$
↑L
$x_e[n]$
LPF
gain=L
π/L
$x_i[n]$



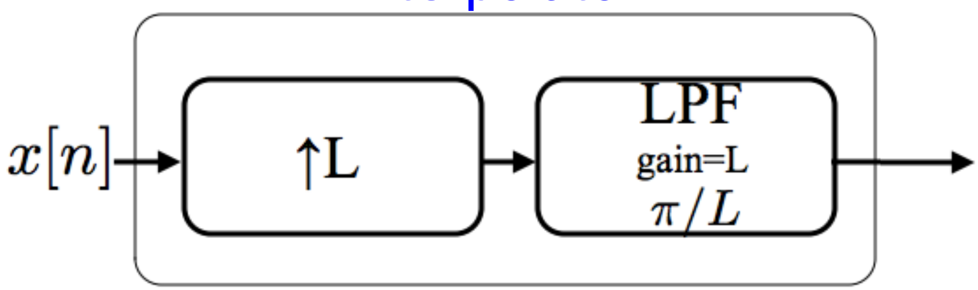
$X_c(j\Omega)$
1
Ω_N
Ω
sampling T
$X(e^{j\omega})$
1/T
$-\pi$
π
sampling T'=T/L
$X_i(e^{j\omega})$
L/T
$-\pi$
$\frac{\pi}{L}$
π
expanding L
$X_e(e^{j\omega})$
1/T
$-\pi$
$\frac{\pi}{L}$
π
L
$-\frac{\pi}{L}$
$\frac{\pi}{L}$

Interpolation and Decimation

decimator

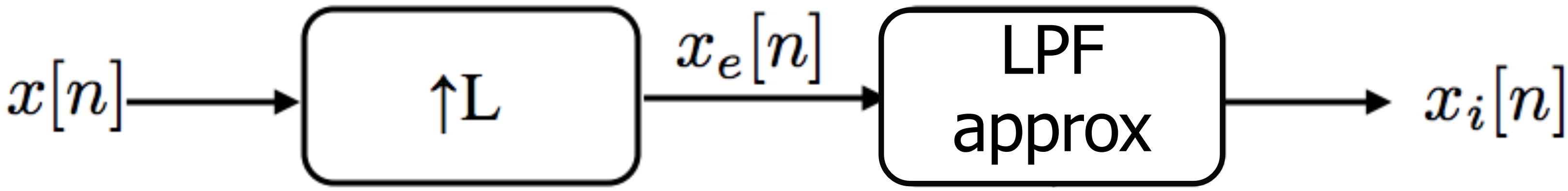


interpolator

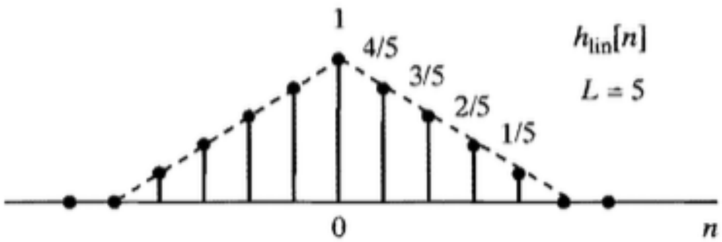


Linear Interpolation -- Frequency Domain

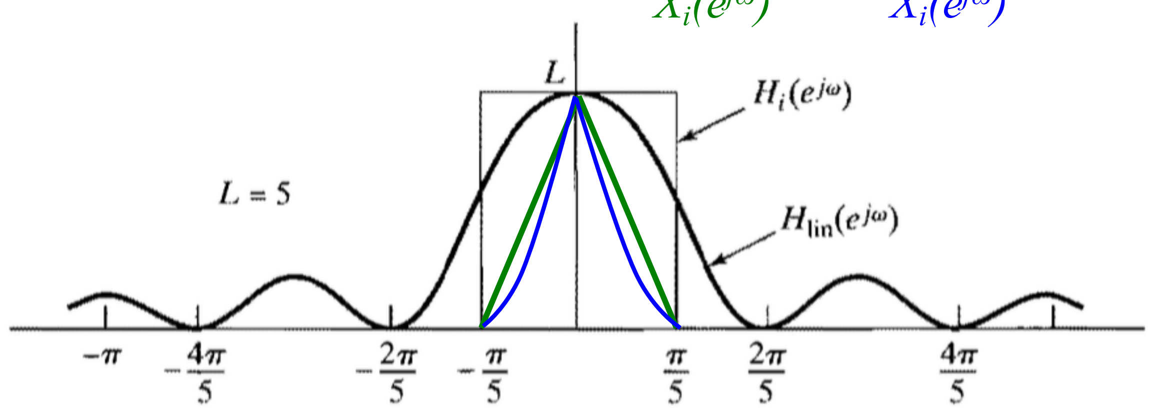
$$x_i[n] = x_e[n] * h_{lin}[n]$$



$$h_{\text{lin}}[n] = \begin{cases} 1 - |n|/L, & |n| \le L, \\ 0, & \text{otherwise,} \end{cases}$$

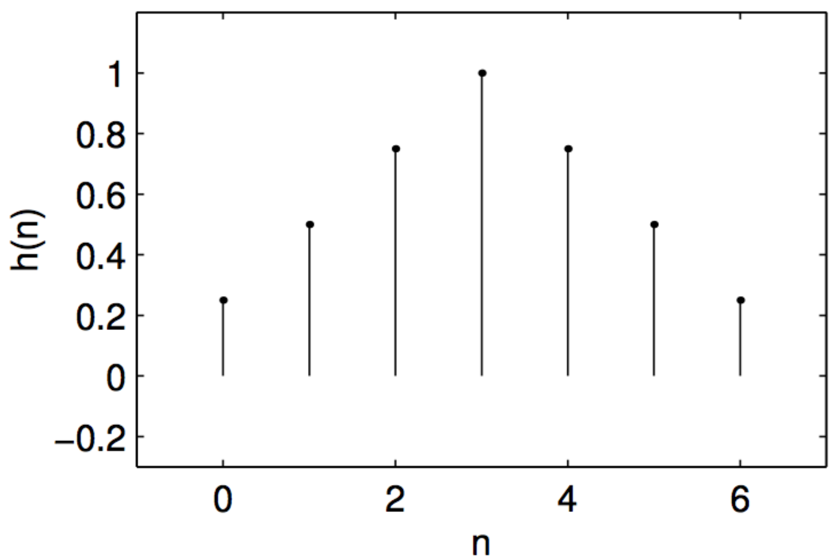


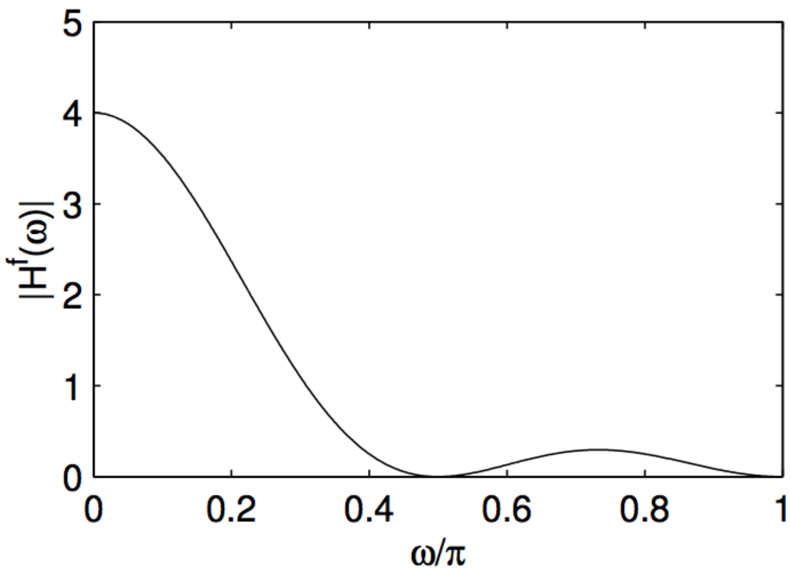
DTFT ⇒



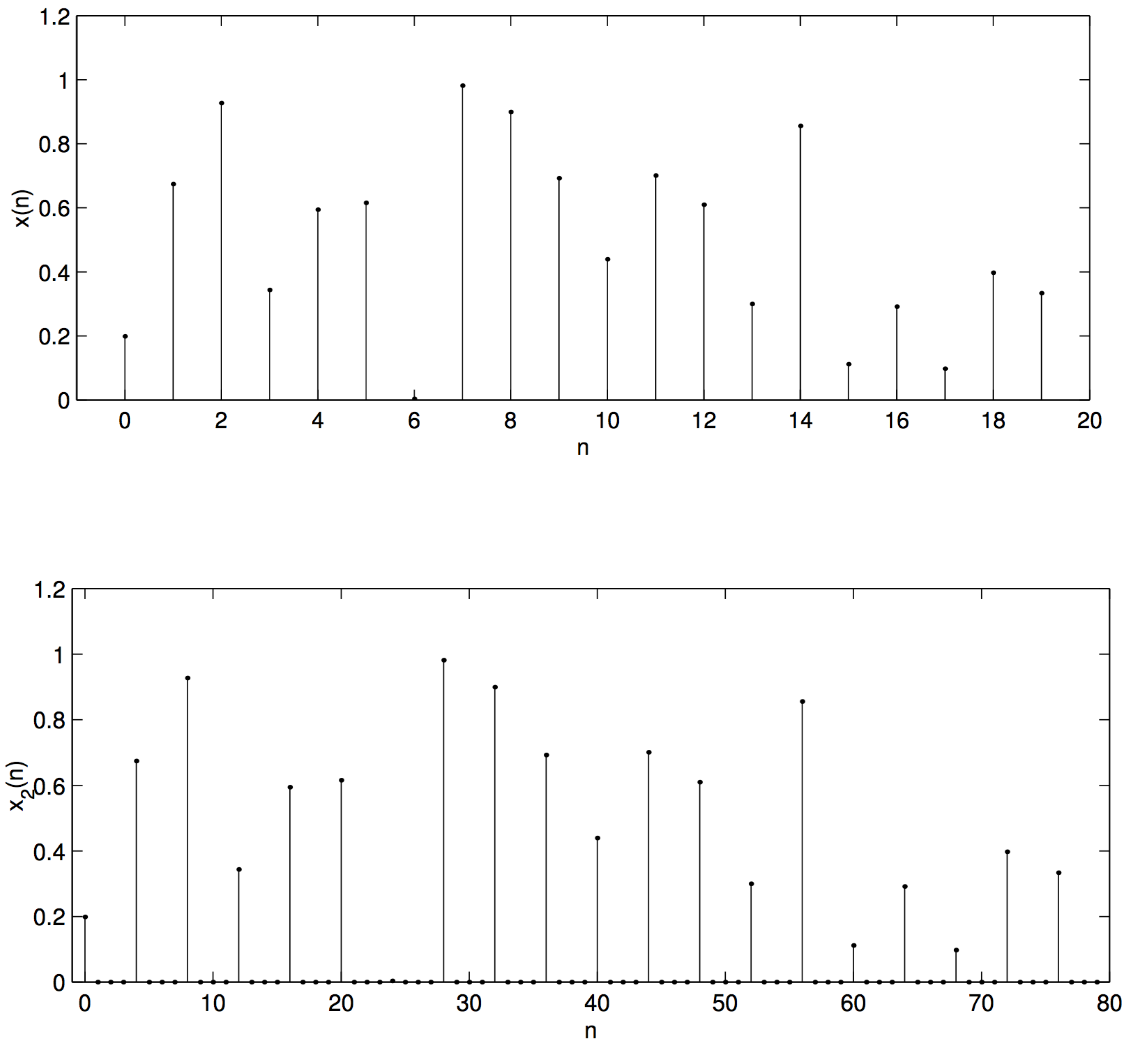
Interpolation Filter Example 1

- This time we use a filter of length 7 with the effect of linear interpolation

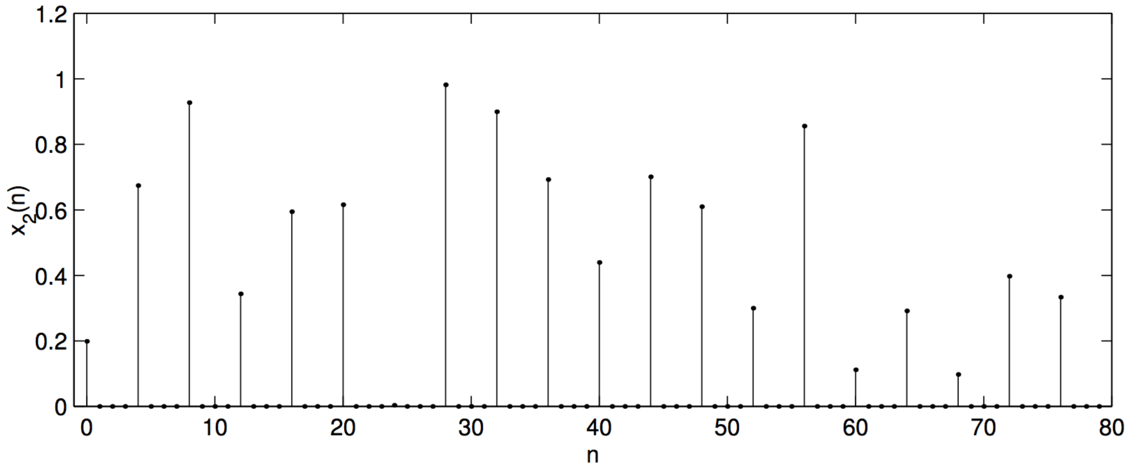




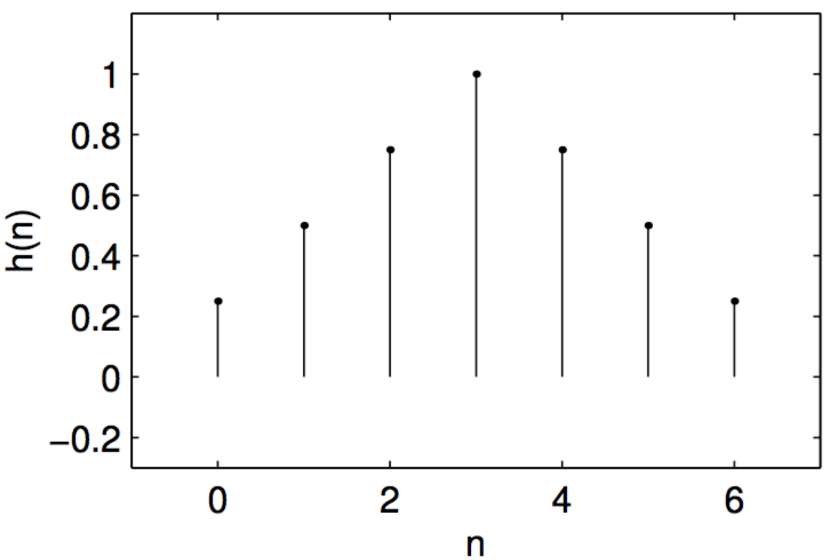
Interpolation Filter Example 1

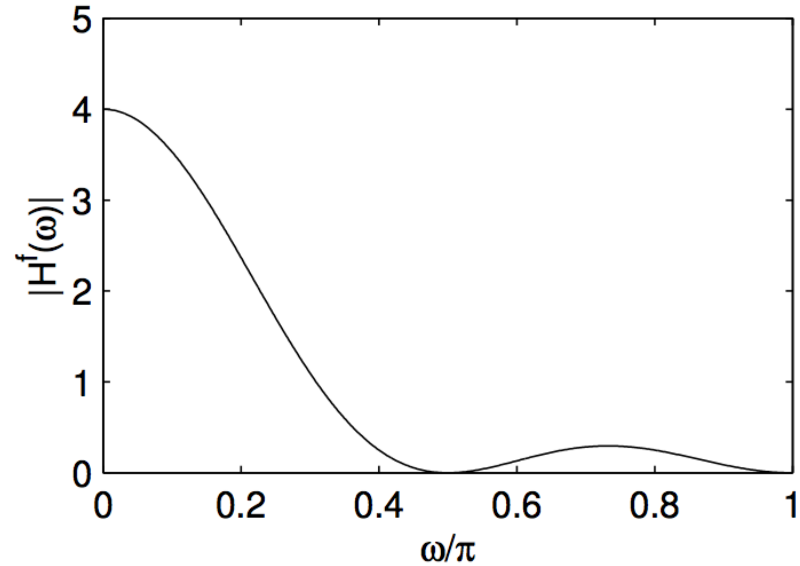


Interpolation Filter Example 1

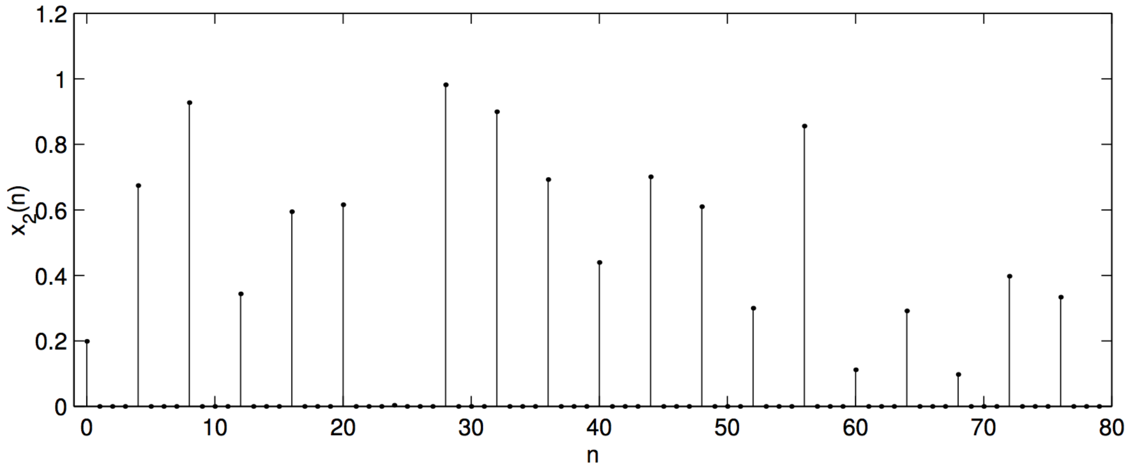


*
(convolve)



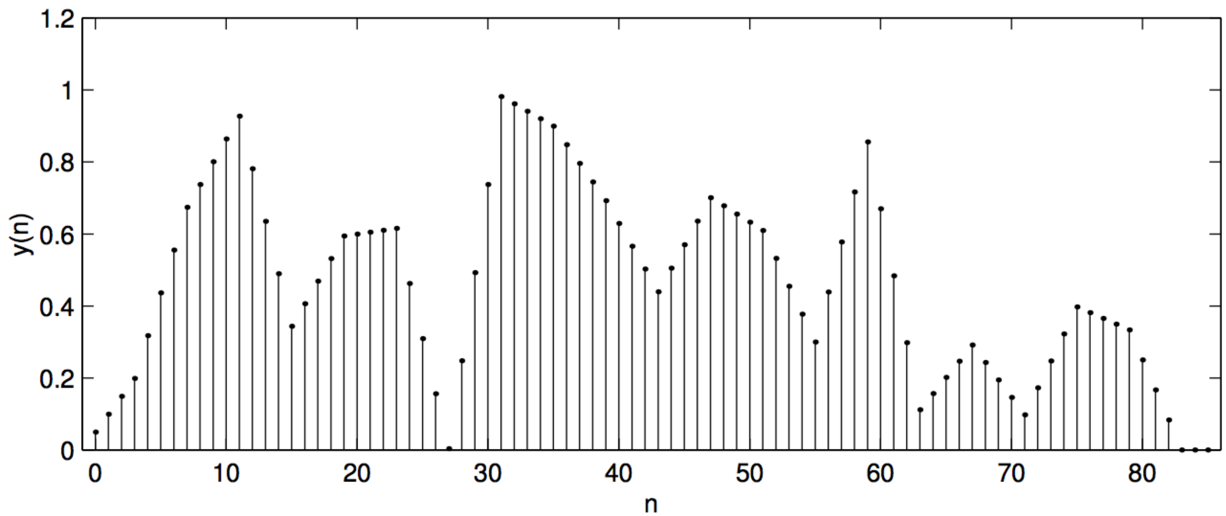


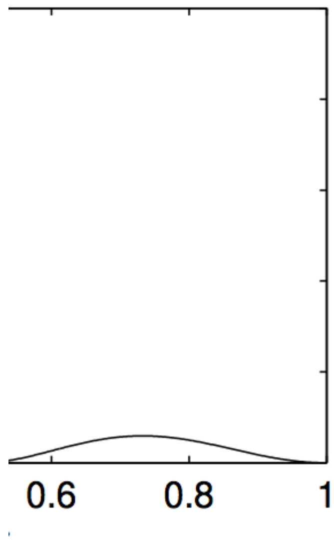
Interpolation Filter Example 1



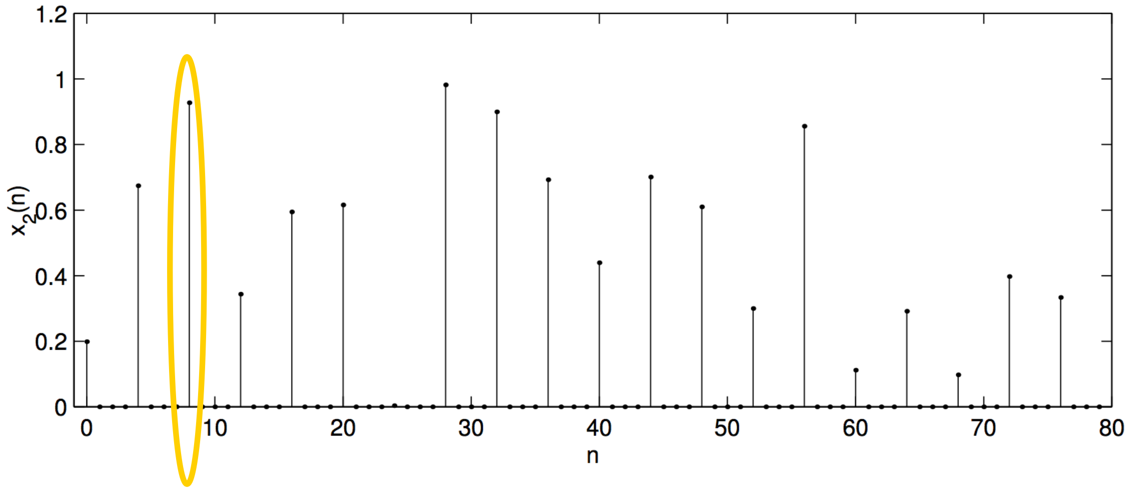
*
(convolve)

h(n)



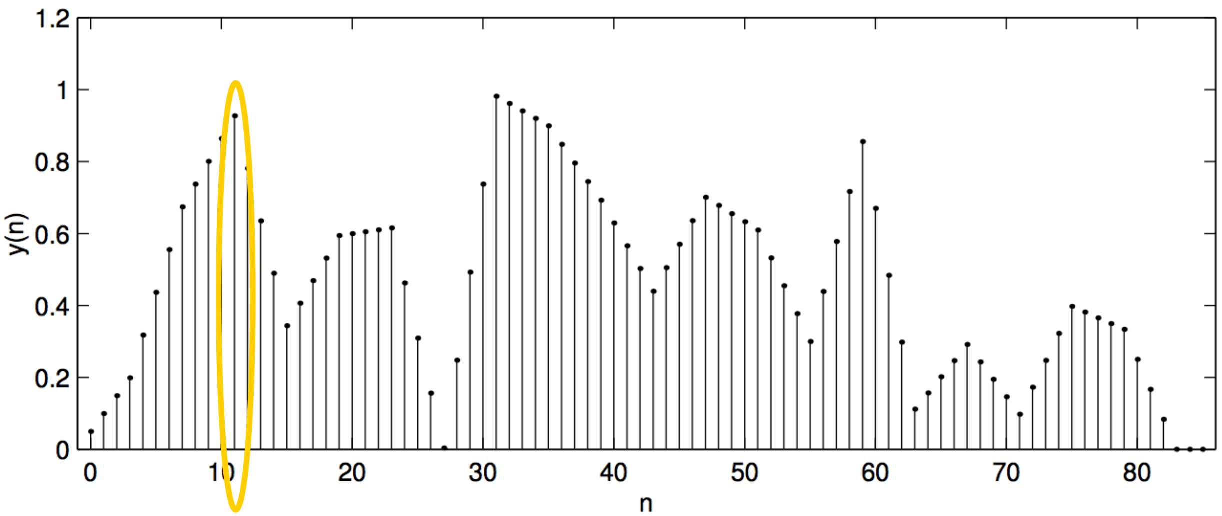


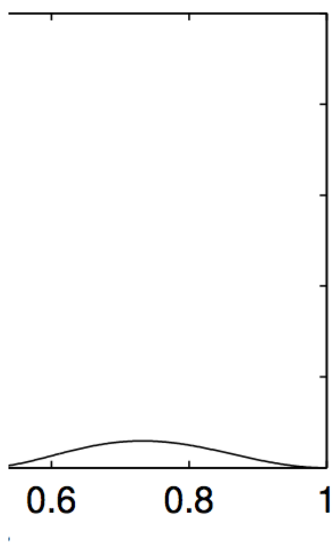
Interpolation Filter Example 1



*
(convolve)

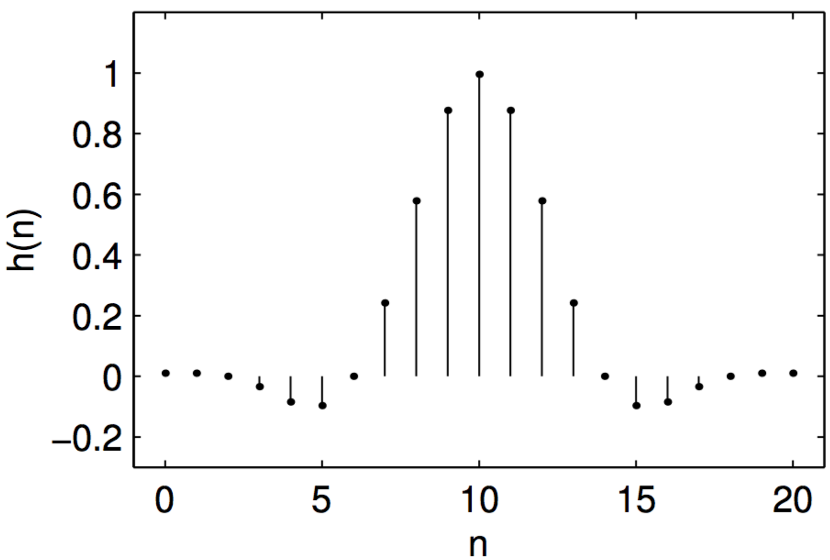
h(n)

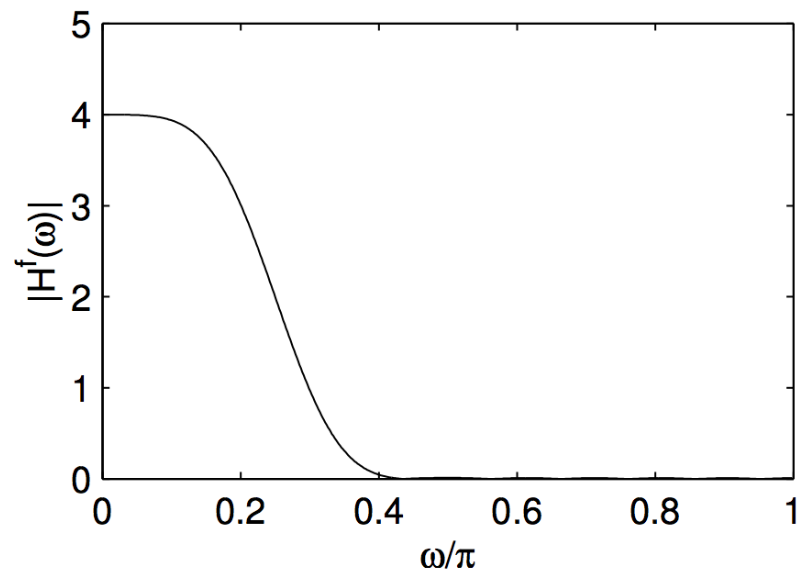




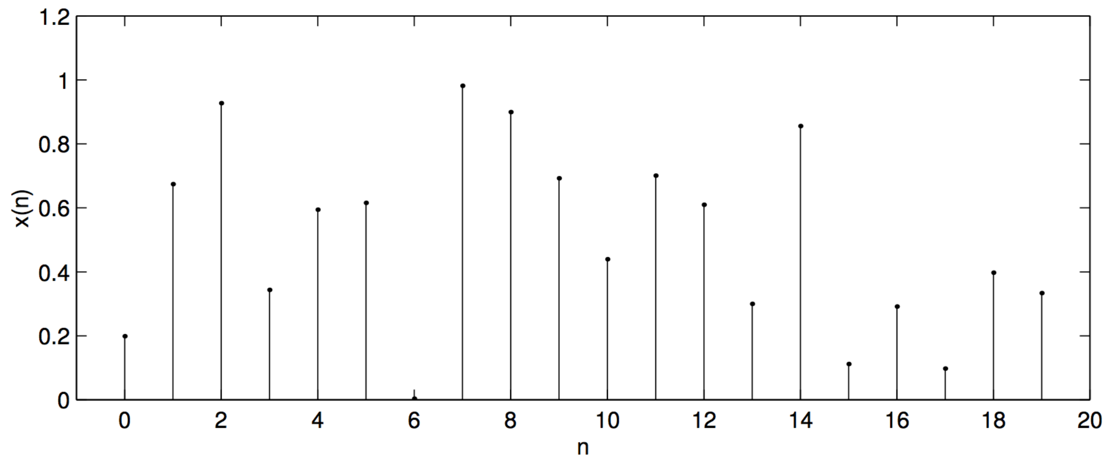
Interpolation Filter Example 2

- In this example, we interpolate a signal x(n) by a factor of 4.
- We use a linear phase Type I FIR lowpass filter of length 21.

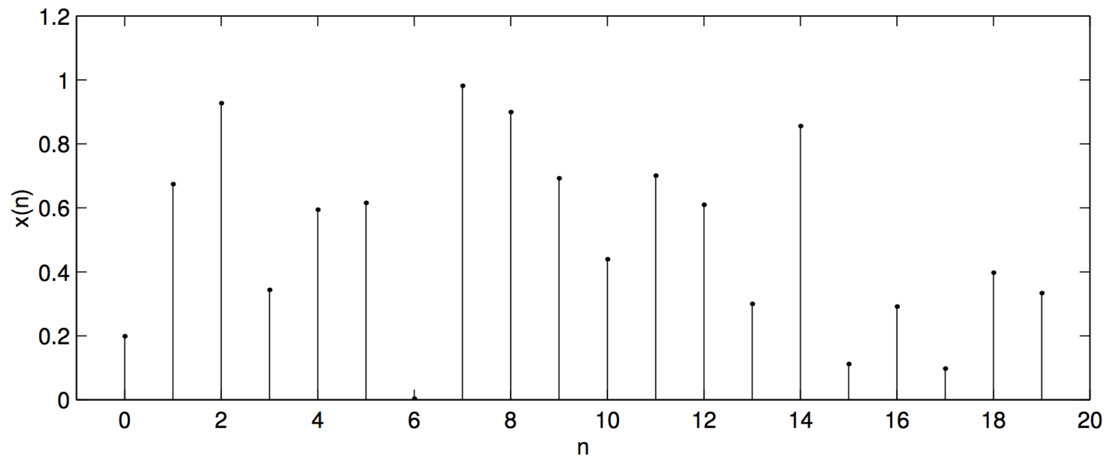


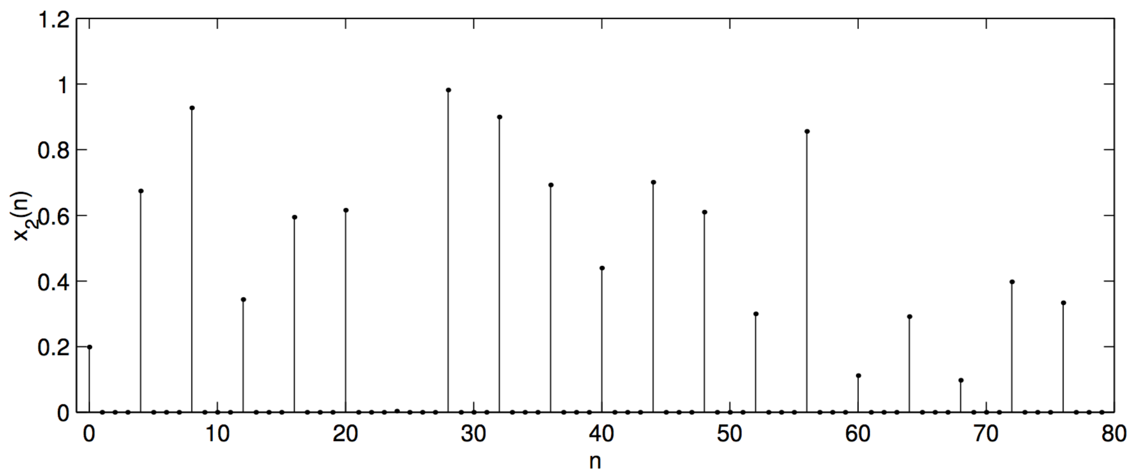


Interpolation Filter Example 2

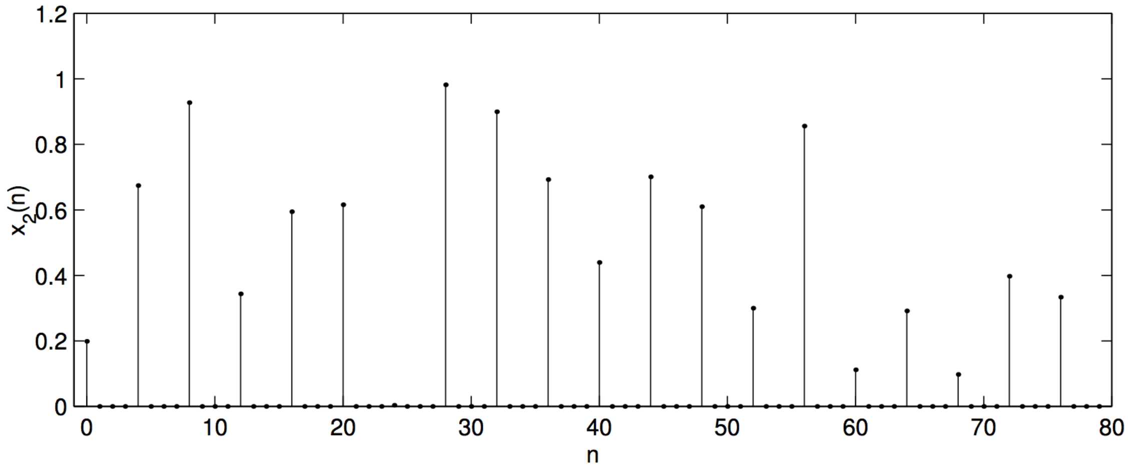


Interpolation Filter Example 2

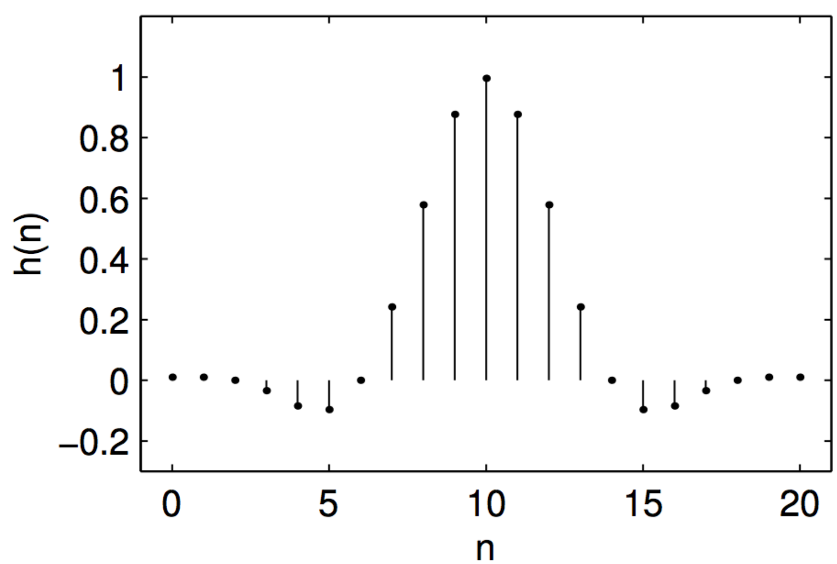


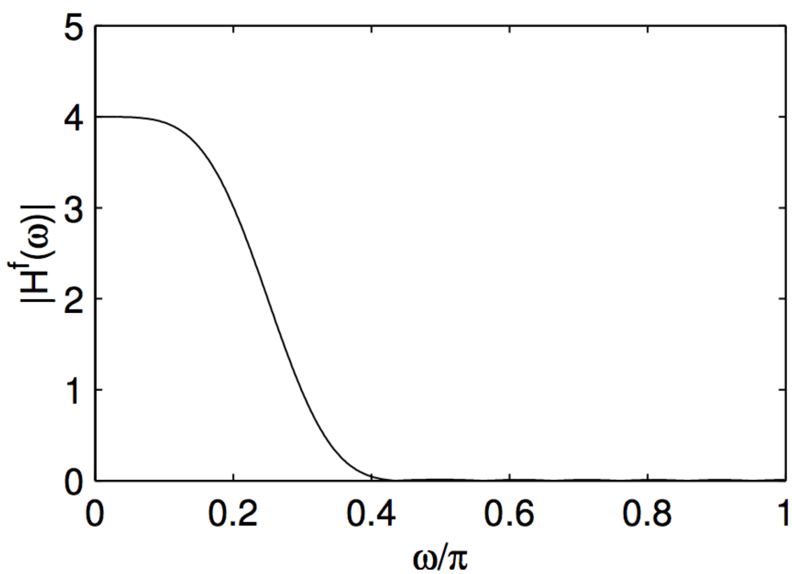


Interpolation Filter Example 2

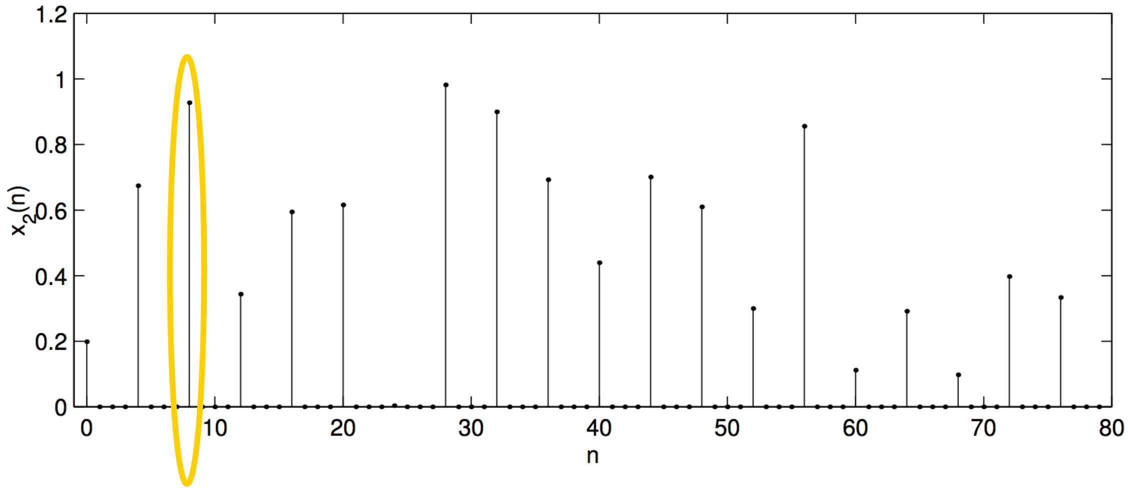


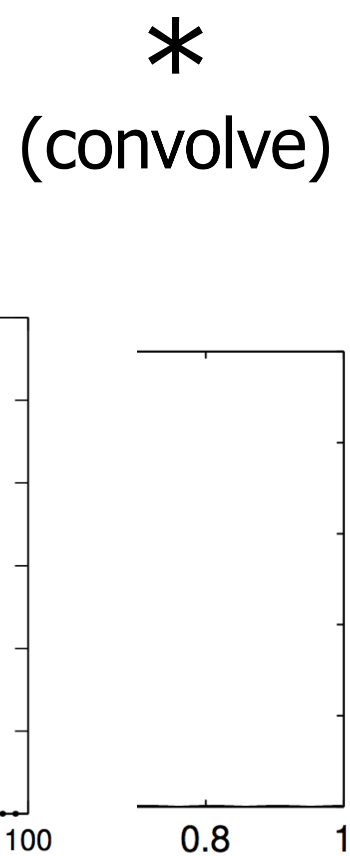
*
(convolve)

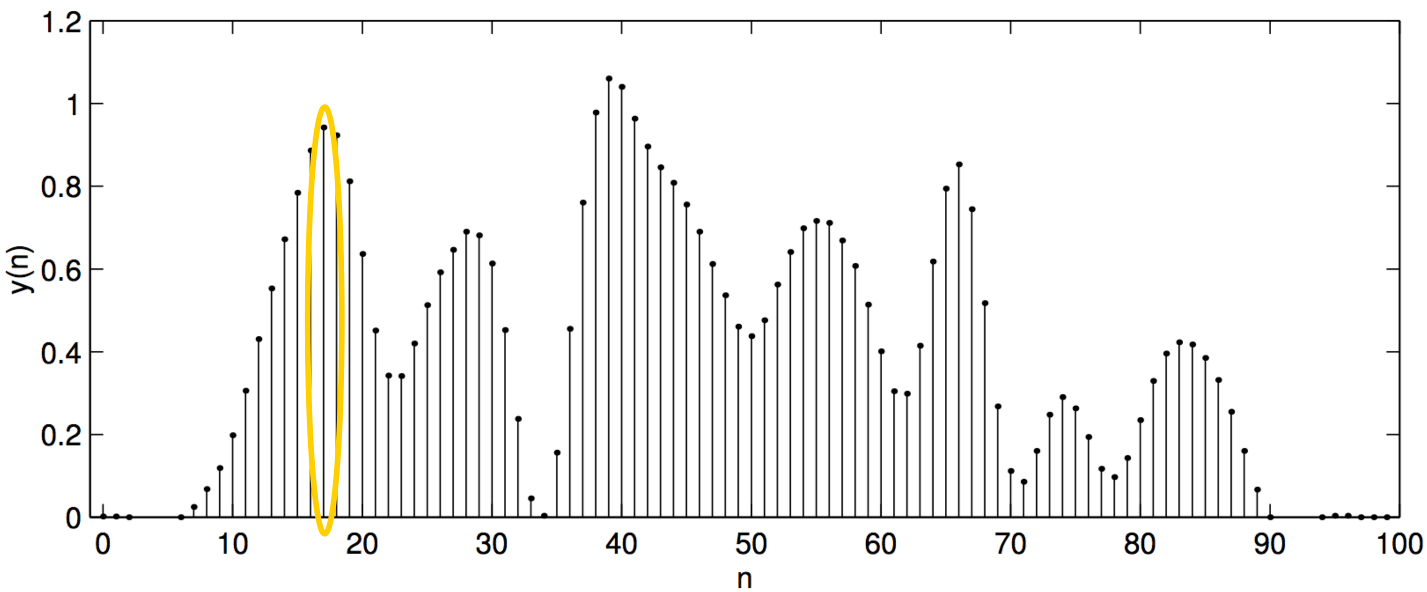




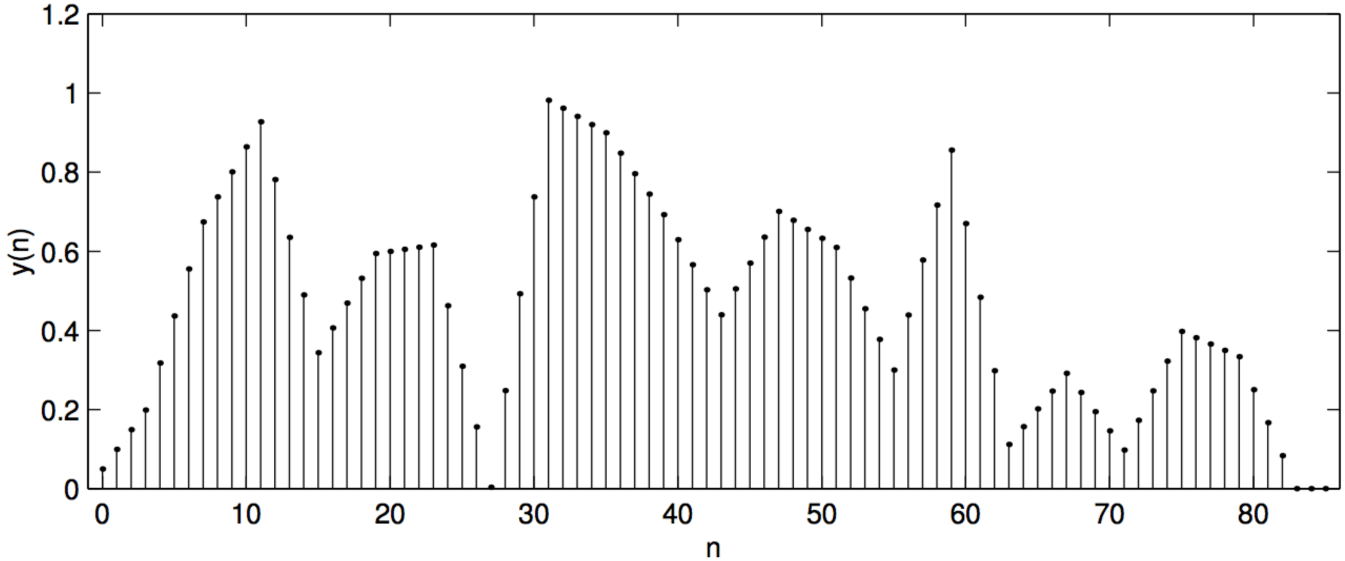
Interpolation Filter Example 2



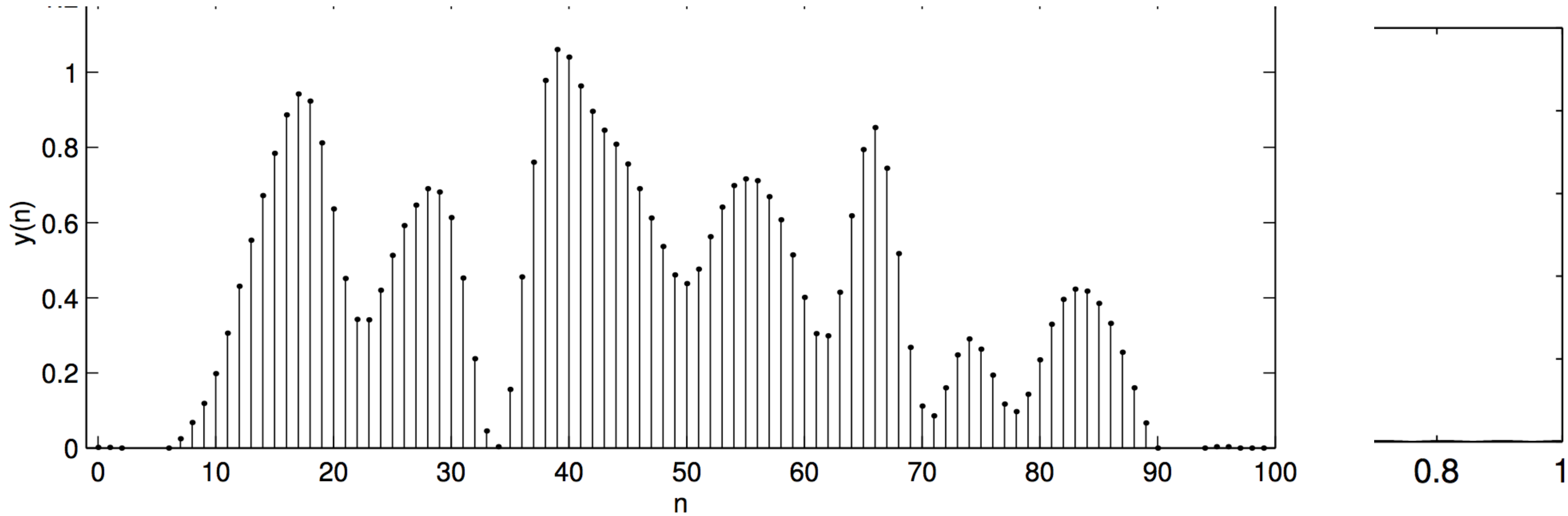




Interpolation Filter Example 2



*
ɔnvolve)



Interpolation Filter Example 3

- When interpolating a signal x(n), the interpolation filter h(n) will in general change the samples of x(n) in addition to filling in the zeros.
- Can a filter be designed so as to preserve the original samples x(n)?

Interpolation Filter Example 3

- When interpolating a signal x(n), the interpolation filter h(n) will in general change the samples of x(n) in addition to filling in the zeros.
- Can a filter be designed so as to preserve the original samples x(n)?
- To be precise, if $y(n) = h(n) * [\uparrow 2]\, x(n)$ then can we design h(n) so that $y(2n) = x(n)$?

Interpolation Filter Example 3

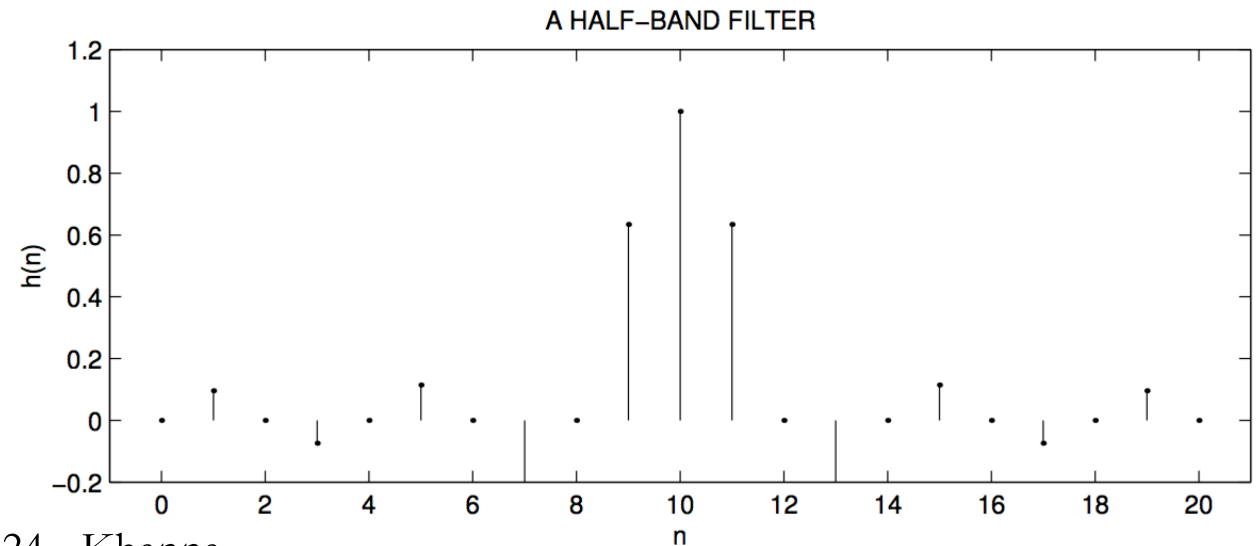
- When interpolating a signal x(n), the interpolation filter h(n) will in general change the samples of x(n) in addition to filling in the zeros.
- Can a filter be designed so as to preserve the original samples x(n)?
- To be precise, if $y(n) = h(n) * [\uparrow 2]\, x(n)$ then can we design h(n) so that $y(2n) = x(n)$?
  - Or more generally, so that $y(2n + n_o) = x(n)$?

Interpolation Filter Example 3

- When interpolating by a factor of 2, if h(n) is a half-band filter, then it will not change the samples x(n).
- A n_o-centered half-band filter h(n) is a filter that satisfies:

$$h(n) = \begin{cases} 1, & \text{for } n = n_o \\ 0, & \text{for } n = n_o \pm 2, 4, 6, \ldots \end{cases}$$

- That means, every second value of h(n) is zero, except for one such value, as shown in the figure.

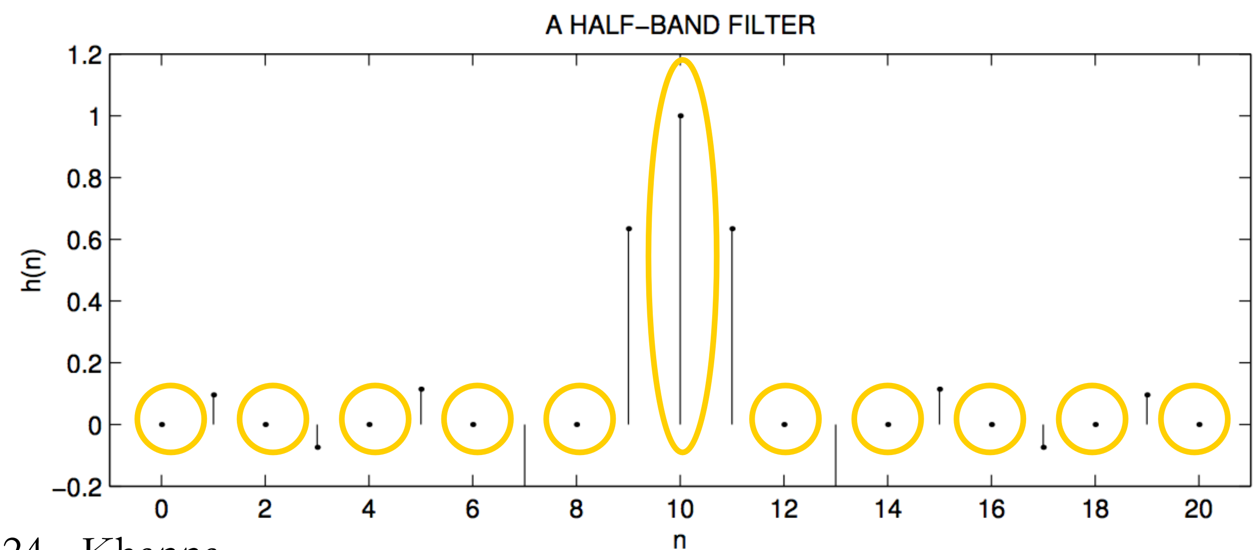


Interpolation Filter Example 3

- When interpolating by a factor of 2, if h(n) is a half-band filter, then it will not change the samples x(n).
- A n_o-centered half-band filter h(n) is a filter that satisfies:

$$h(n) = \begin{cases} 1, & \text{for } n = n_o \\ 0, & \text{for } n = n_o \pm 2, 4, 6, \ldots \end{cases}$$

- That means, every second value of h(n) is zero, except for one such value, as shown in the figure.

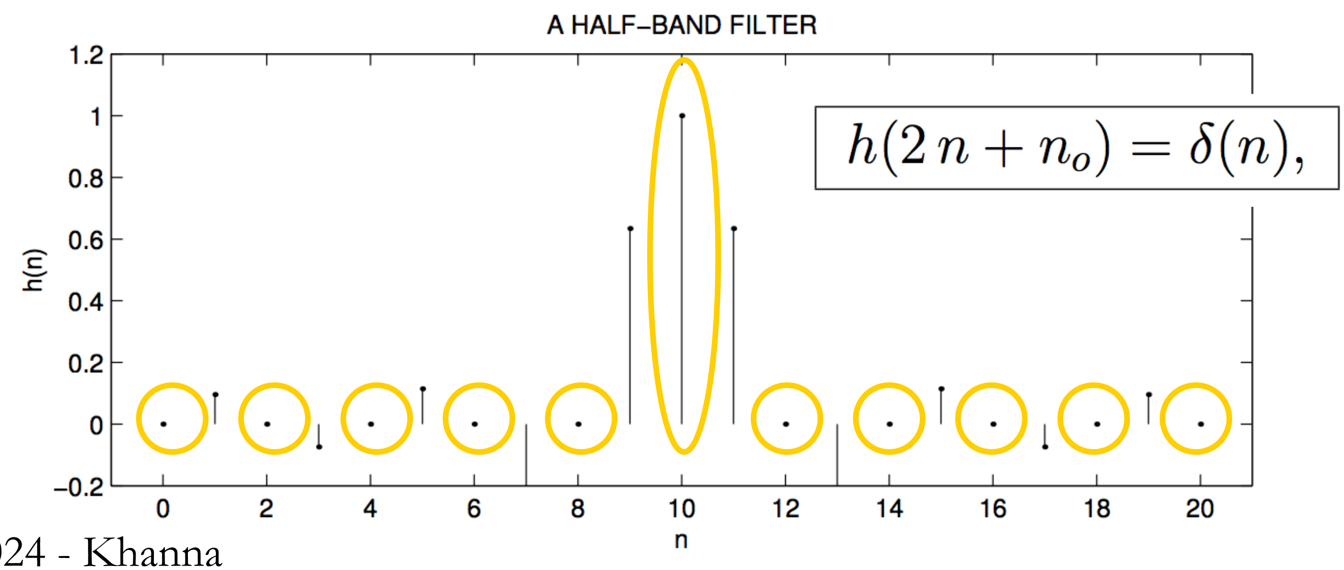


Interpolation Filter Example 3

- When interpolating by a factor of 2, if h(n) is a half-band filter, then it will not change the samples x(n).
- A n_o-centered half-band filter h(n) is a filter that satisfies:

$$h(n) = \begin{cases} 1, & \text{for } n = n_o \\ 0, & \text{for } n = n_o \pm 2, 4, 6, \ldots \end{cases}$$

- That means, every second value of h(n) is zero, except for one such value, as shown in the figure.



Interpolation Filter Example 4

- When interpolating a signal x(n) by a factor L, the original samples of x(n) are preserved if h(n) is a Nyquist-L filter.
- A Nyquist-L filter simply generalizes the notion of the halfband filter to $L > 2$.

Interpolation Filter Example 4

- When interpolating a signal x(n) by a factor L, the original samples of x(n) are preserved if h(n) is a Nyquist-L filter.
- A Nyquist-L filter simply generalizes the notion of the halfband filter to L > 2.
- A (0-centered) Nyquist-L filter h(n) is one for which

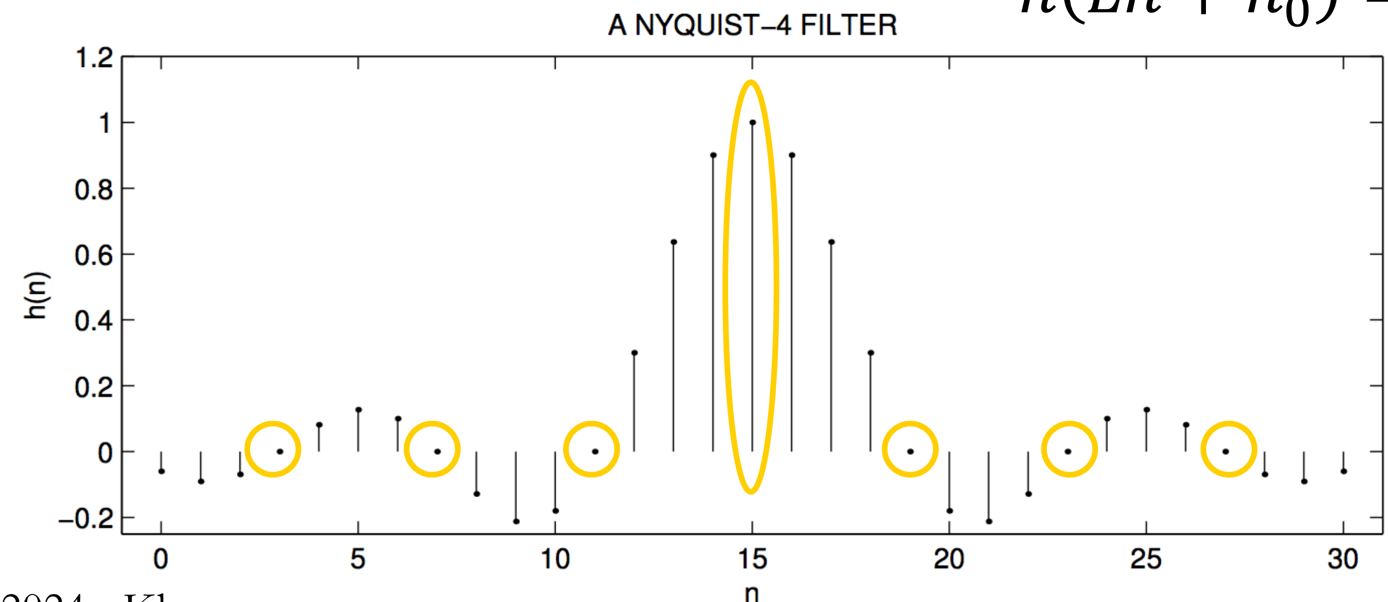
$$\boxed{h(L\,n) = \delta(n).}$$

Interpolation Filter Example 4

- When interpolating a signal x(n) by a factor L, the original samples of x(n) are preserved if h(n) is a Nyquist-L filter.
- A Nyquist-L filter simply generalizes the notion of the halfband filter to L > 2.
- A (0-centered) Nyquist-L filter h(n) is one for which

$$\boxed{h(L\,n) = \delta(n).}$$

$$h(Ln + n_0) = \delta(n)$$



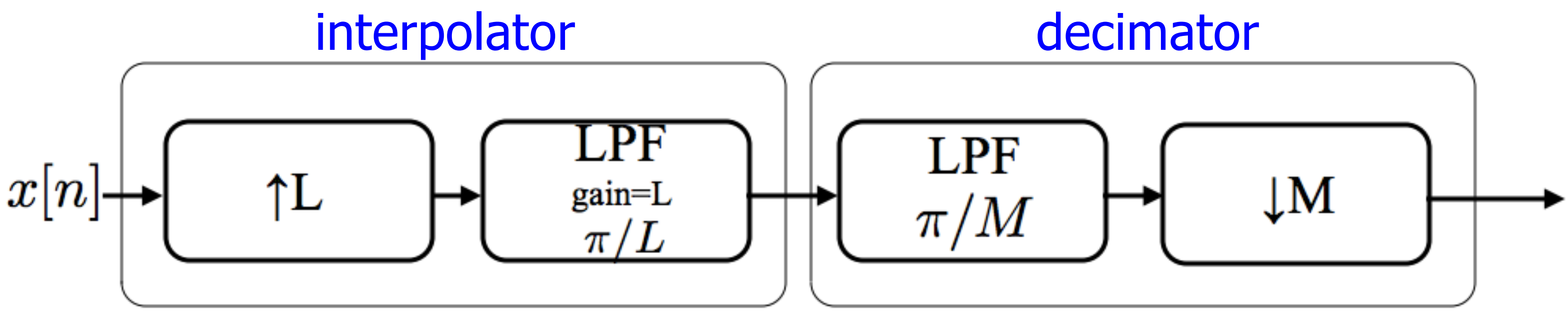
Non-integer Resampling

Non-integer Resampling

- $T' = TM/L$

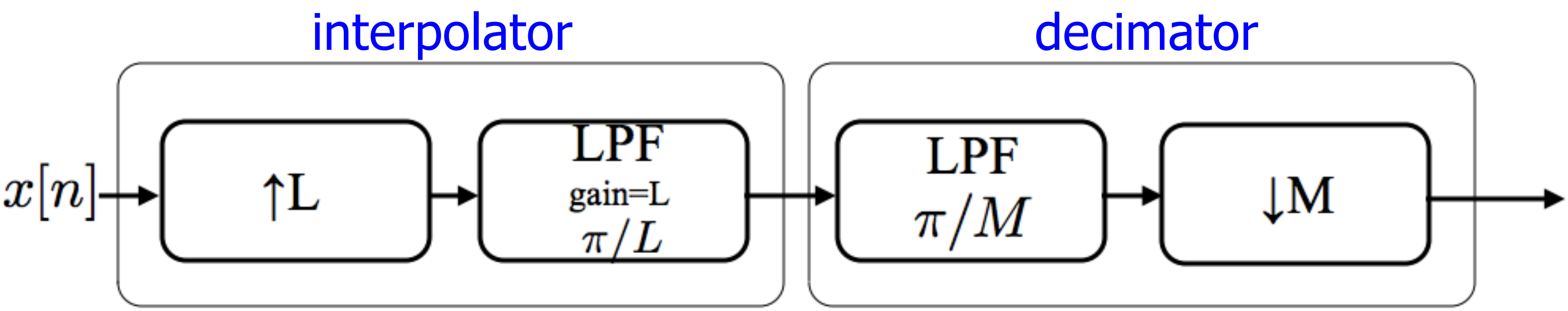
Non-integer Resampling

- T'=TM/L
  - Upsample by L, then downsample by M

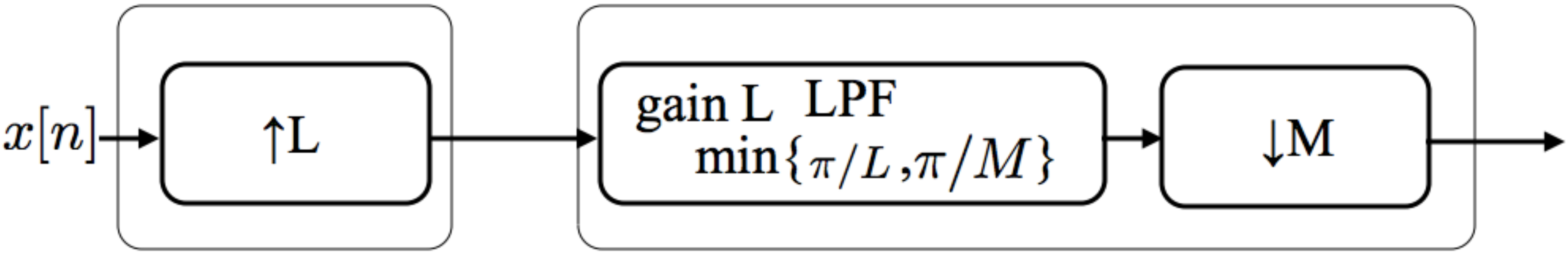


Non-integer Resampling

- T'=TM/L
  - Upsample by L, then downsample by M

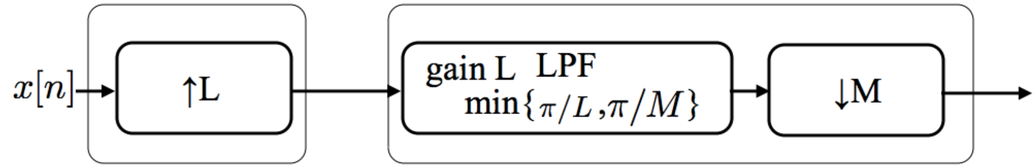


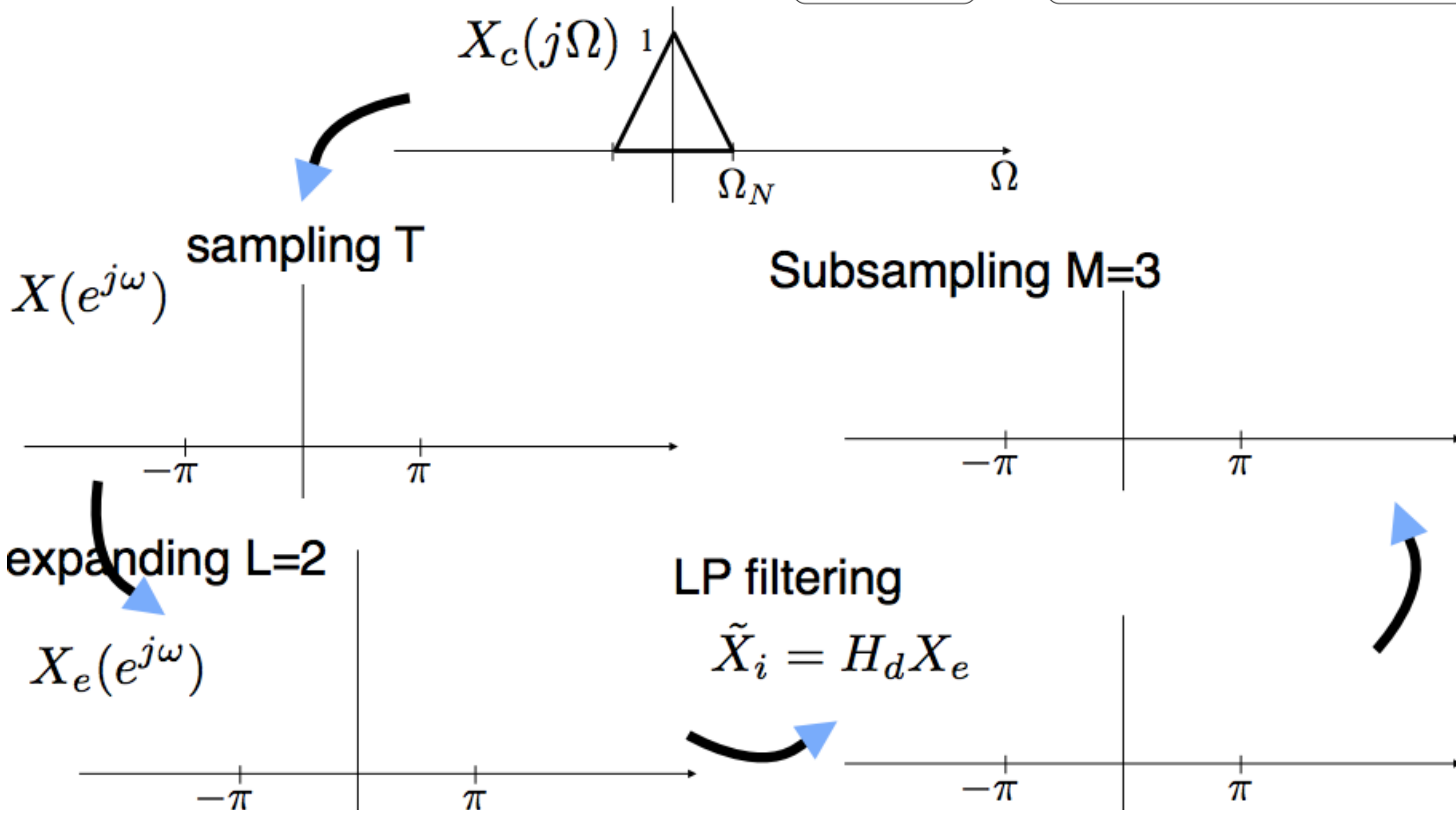
Or,



Example

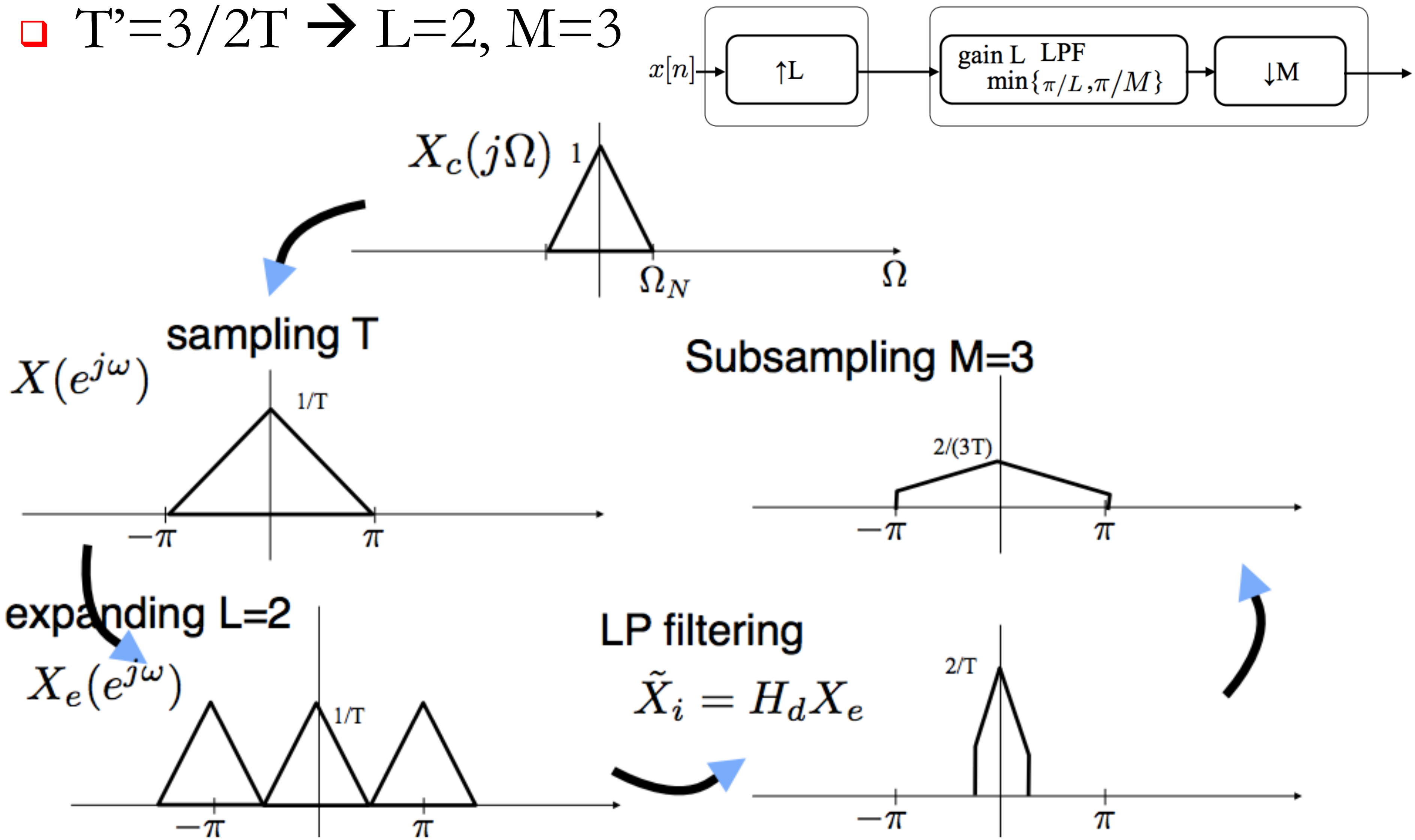
- T'=3/2T → L=2, M=3





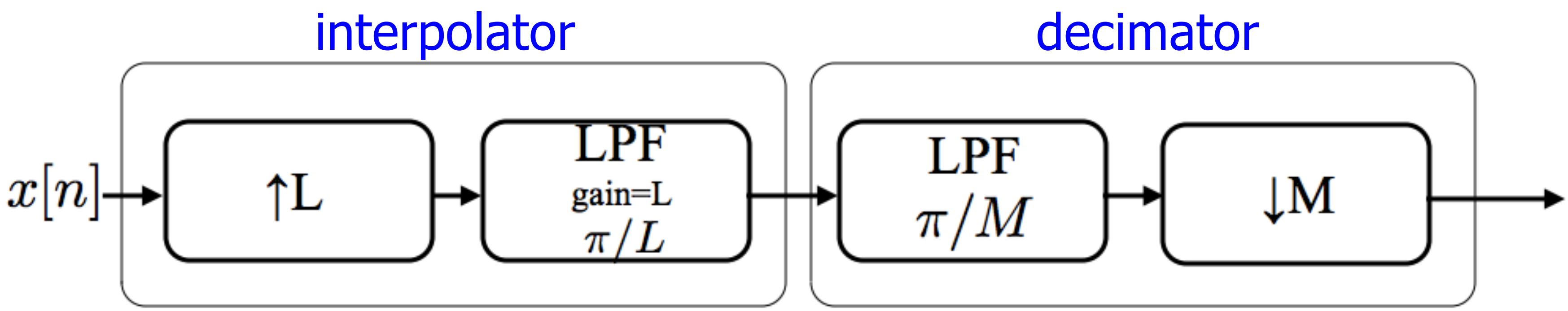
Example

- T'=3/2T → L=2, M=3

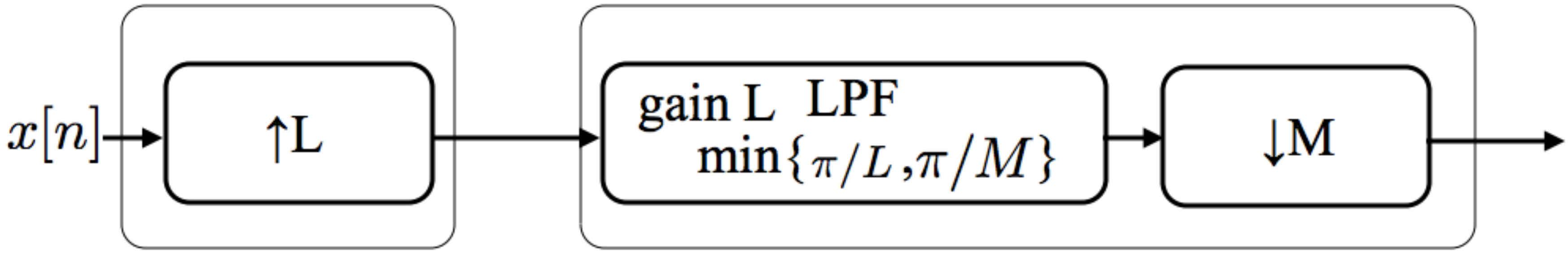


Non-integer Sampling

- T'=TM/L
  - Downsample by M, then upsample by L?



Or,



Example

- What if we want to resample by 1.01T?

Example

- What if we want to resample by 1.01T?
  - Upsample by L=100
  - Filter $\pi/101$
  - Downsample by M=101

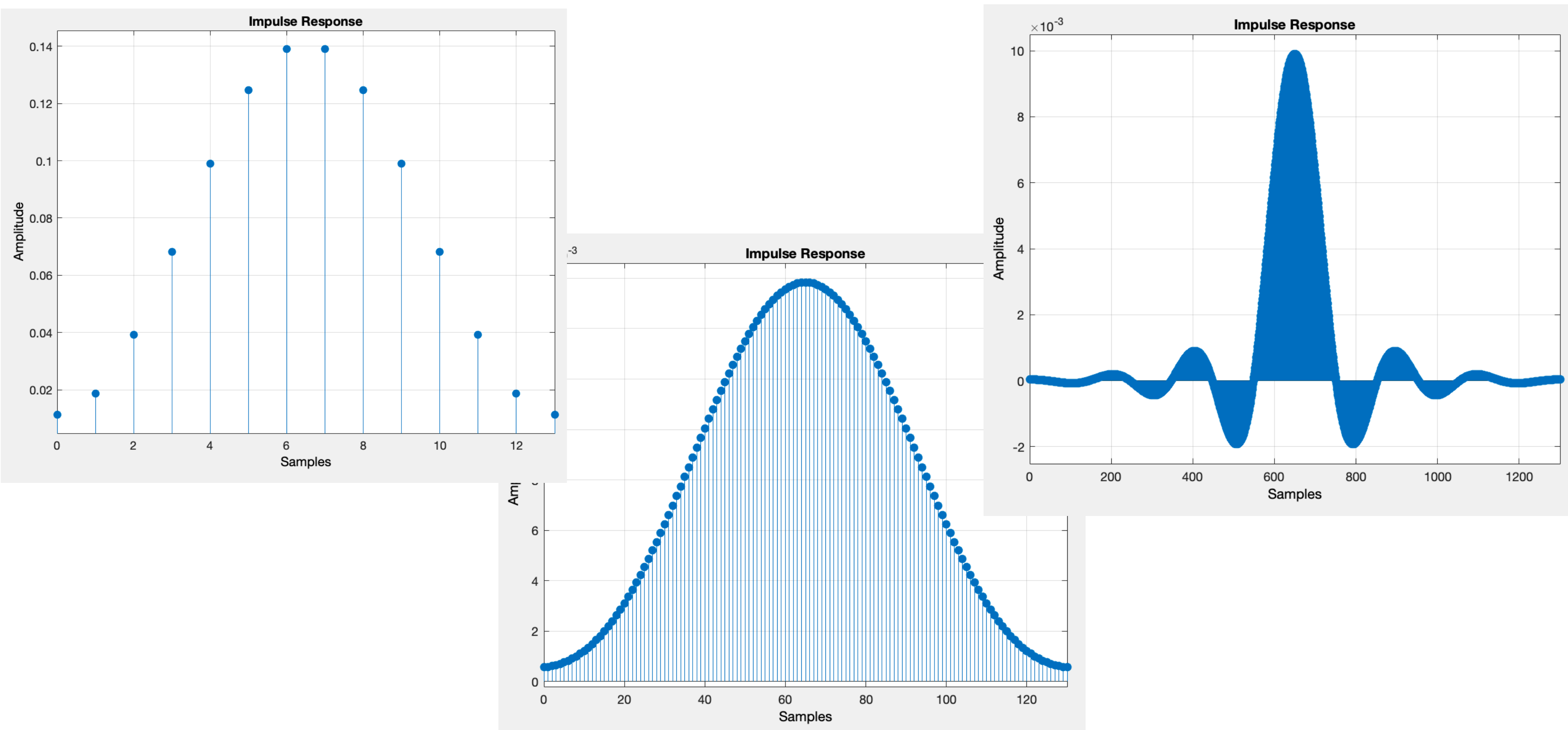
Example

- What if we want to resample by 1.01T?
  - Upsample by L=100
  - Filter $\pi/101$ ($$$$$)
  - Downsample by M=101

- Fortunately there are ways around it!
  - Called multi-rate signal processing
  - Uses compressors, expanders and filtering

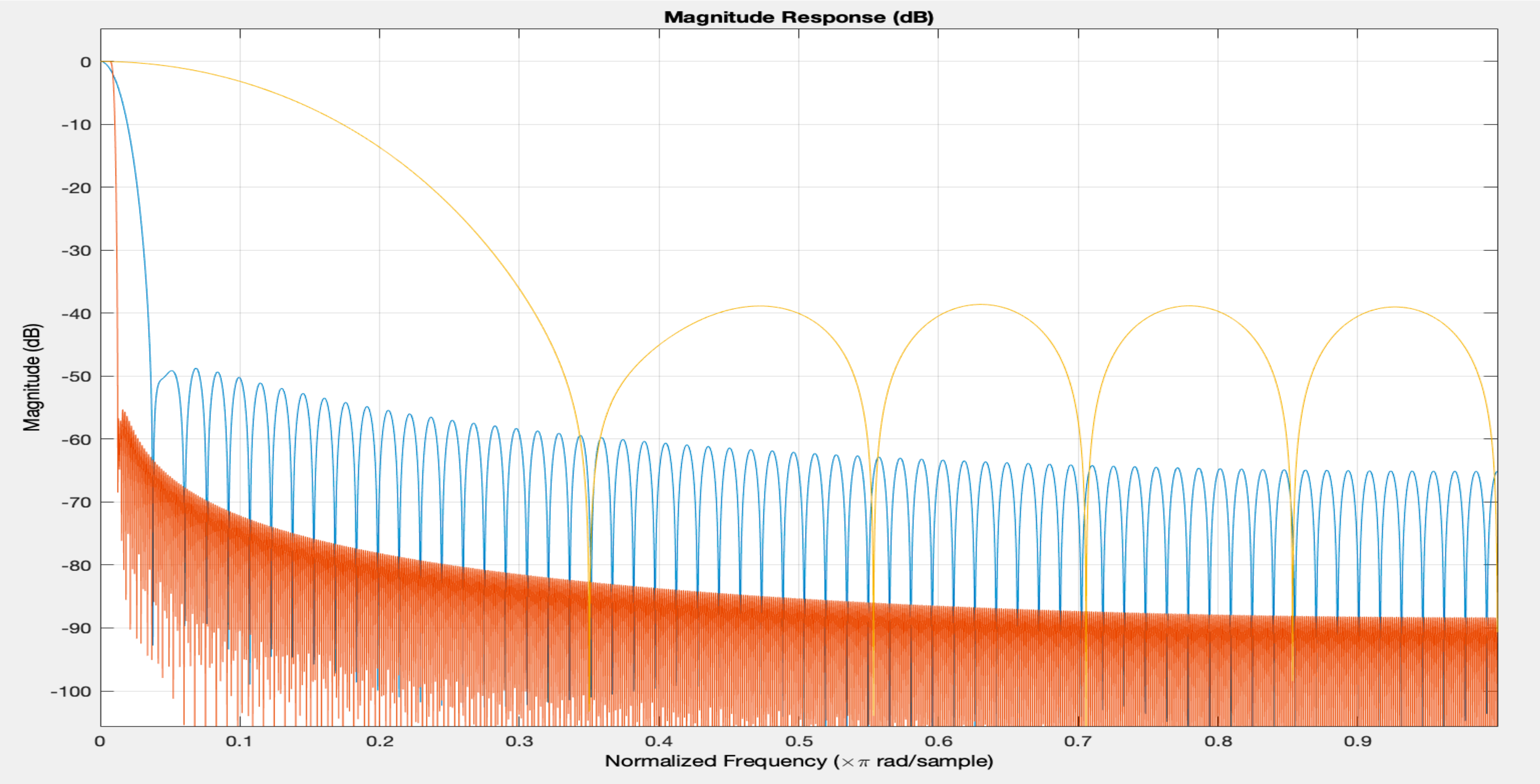
Filter Cost

- Design filter with cutoff frequency of $\pi/101 = 0.01\pi$:
  - Compare 3 filters: length 13, 130, and 1300



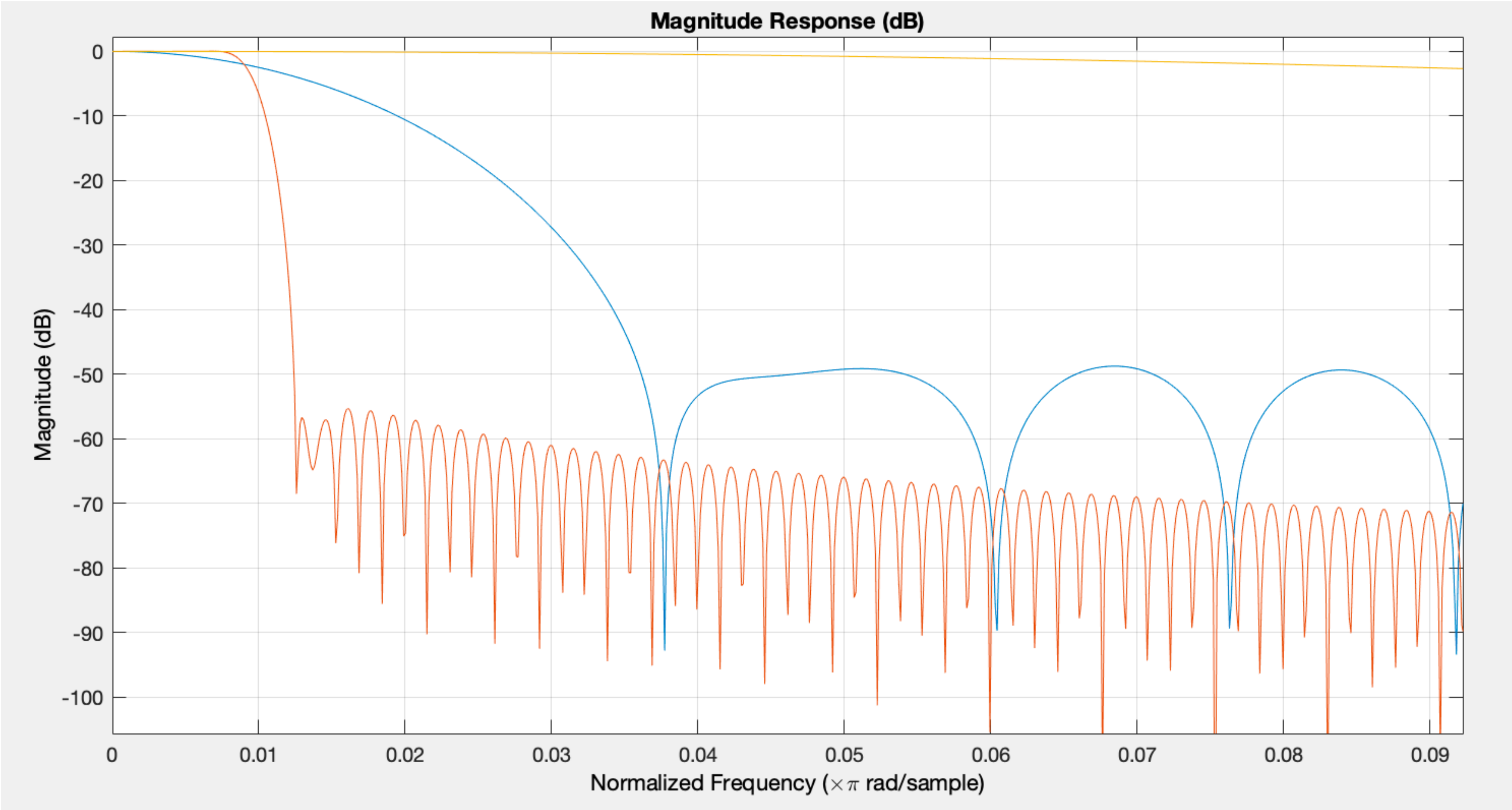
Filter Cost

- Design filter with cutoff frequency of $\pi/101 = 0.01\pi$:
  - Compare 3 filters: length 13, 130, and 1300



Filter Cost

- Design filter with cutoff frequency of $\pi/101 = 0.01\pi$:
  - Compare 3 filters: length 13, 130, and 1300



Filter Cost

- Design filter with cutoff frequency of $\pi/101 = 0.01\pi$:
  - Compare 3 filters: length 13, 130, and 1300

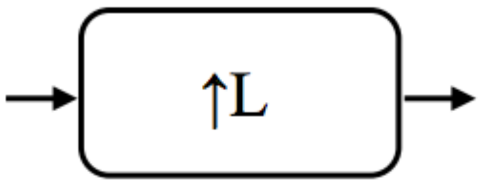
```
>> cost(firFilt3)

ans =

  struct with fields:

              NumCoefficients: 14
                    NumStates: 13
 MultiplicationsPerInputSample: 14
       AdditionsPerInputSample: 13
```

```
>> cost(firFilt1)

ans =

  struct with fields:

              NumCoefficients: 131
                    NumStates: 130
 MultiplicationsPerInputSample: 131
       AdditionsPerInputSample: 130
```

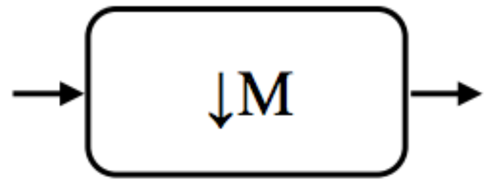
```
>> cost(firFilt2)

ans =

  struct with fields:

              NumCoefficients: 1301
                    NumStates: 1300
 MultiplicationsPerInputSample: 1301
       AdditionsPerInputSample: 1300
```

Interchanging Operations

LTI?



"expander"



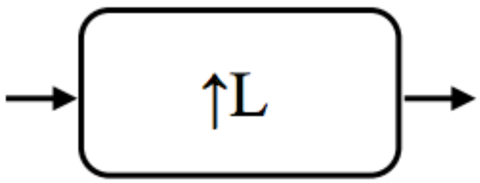
"compressor"

not LTI!

Upsampling
-expanding in time
-compressing in frequency

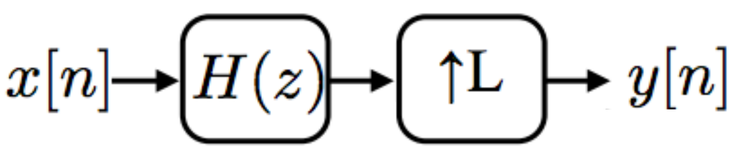
Downsampling
-compressing in time
-expanding in frequency

Interchanging Operations - Expander

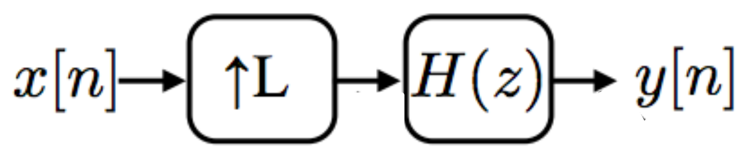


"expander"

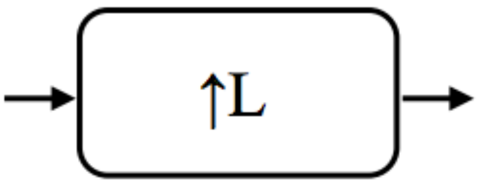
Upsampling
-expanding in time
-compressing in frequency



?

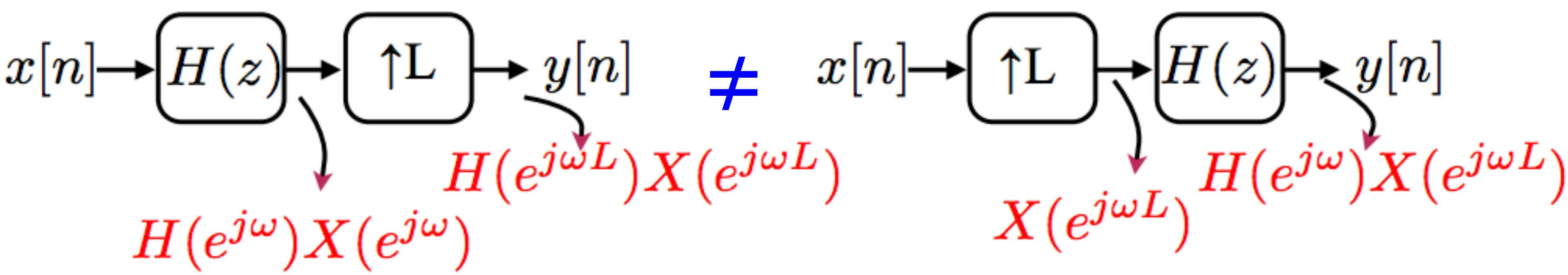


Interchanging Operations - Expander

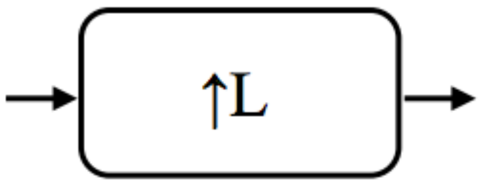


"expander"

Upsampling
-expanding in time
-compressing in frequency

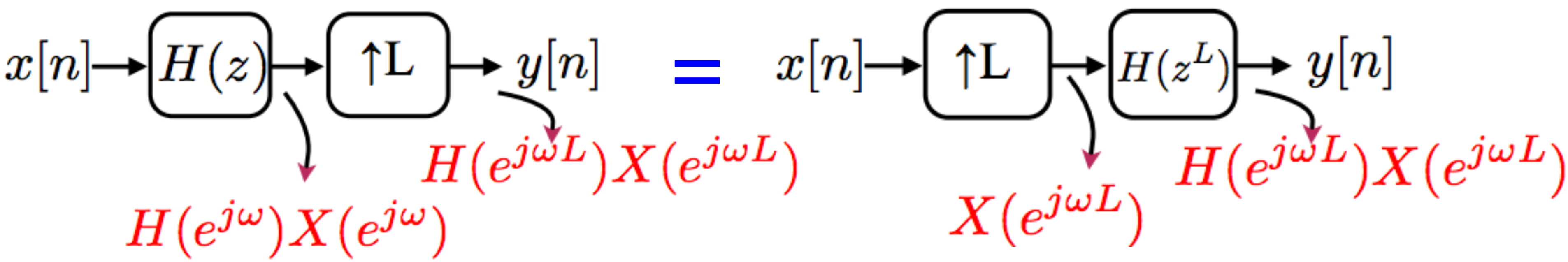


Interchanging Operations - Expander

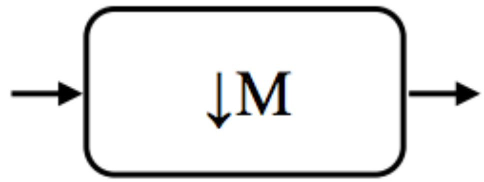


"expander"

Upsampling
-expanding in time
-compressing in frequency



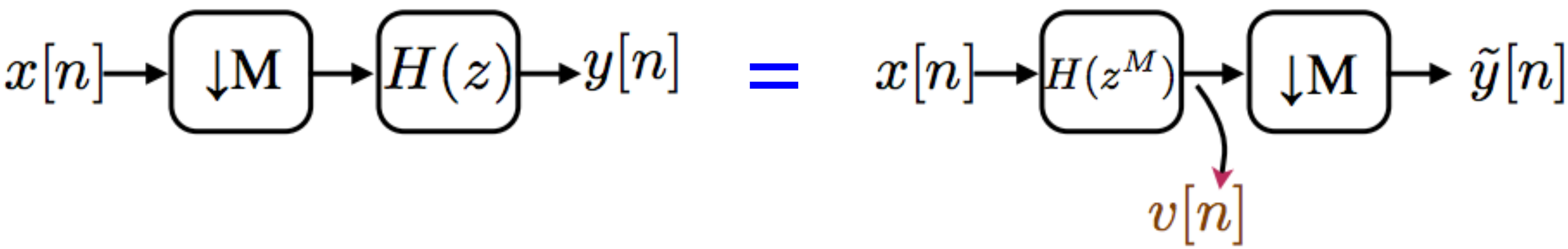
Interchanging Operations - Compressor



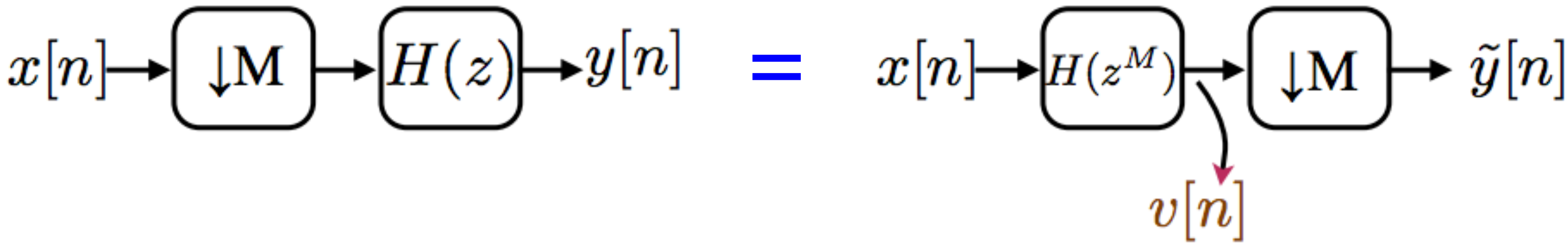
not LTI!

"compressor"

Downsampling
-compressing in time
-expanding in frequency

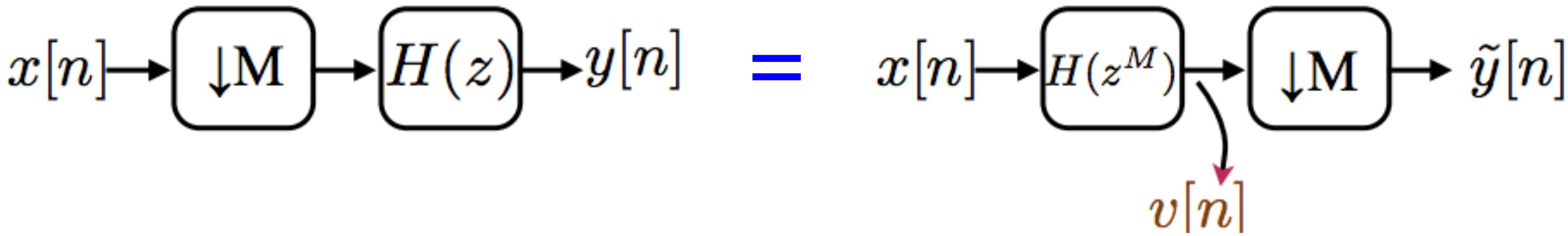


Interchanging Operations - Compressor



$$
\begin{aligned}
Y(e^{j\omega}) &= H(e^{j\omega})\left(\frac{1}{M}\sum_{i=0}^{M-1} X\left(e^{j\left(\frac{\omega}{M}-\frac{2\pi i}{M}\right)}\right)\right) \\
&= \frac{1}{M}\sum_{i=0}^{M-1} \underbrace{H\left(e^{j(\omega-2\pi i)}\right)}_{H(e^{j\omega})} X\left(e^{j\left(\frac{\omega}{M}-\frac{2\pi i}{M}\right)}\right) \\
&= \frac{1}{M}\sum_{i=0}^{M-1} H\left(e^{jM\left(\frac{\omega}{M}-\frac{2\pi i}{M}\right)}\right) X\left(e^{j\left(\frac{\omega}{M}-\frac{2\pi i}{M}\right)}\right)
\end{aligned}
$$

Interchanging Operations - Compressor



$$Y(e^{j\omega}) = \frac{1}{M}\sum_{i=0}^{M-1} H\left(e^{jM\left(\frac{\omega}{M}-\frac{2\pi i}{M}\right)}\right) X\left(e^{j\left(\frac{\omega}{M}-\frac{2\pi i}{M}\right)}\right)$$

$$V(e^{j\omega}) = H(e^{j\omega M})X(e^{j\omega})$$

$$\tilde{Y}(e^{j\omega}) = \frac{1}{M}\sum_{i=0}^{M-1} V\left(e^{j\left(\frac{\omega}{M}-\frac{2\pi i}{M}\right)}\right)$$

Interchanging Operations - Summary

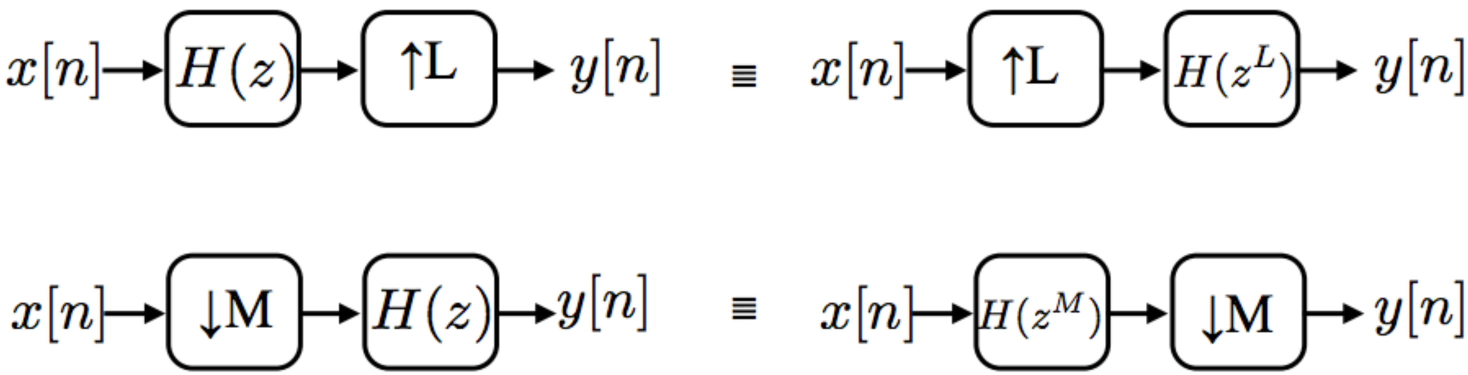
Filter and expander   Expander and expanded filter*

Compressor and filter   Expanded filter* and compressor

*Expanded filter = expanded impulse response, compressed freq response

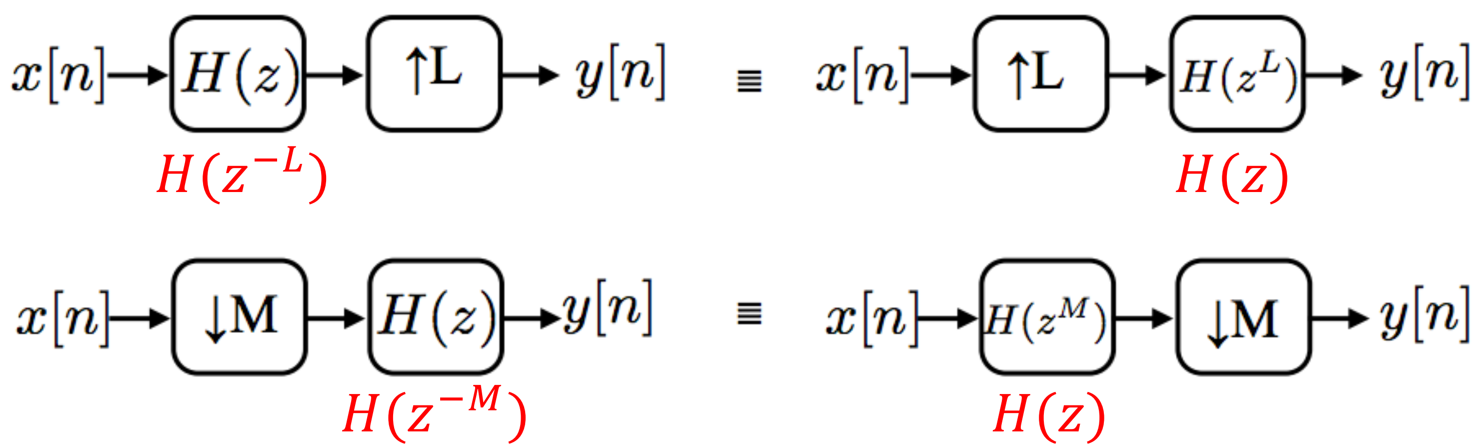
Multi-Rate Signal Processing

- What if we want to resample by 1.01T?
  - Expand by L=100
  - Filter π/101 ($$$$$)
  - Compress by M=101



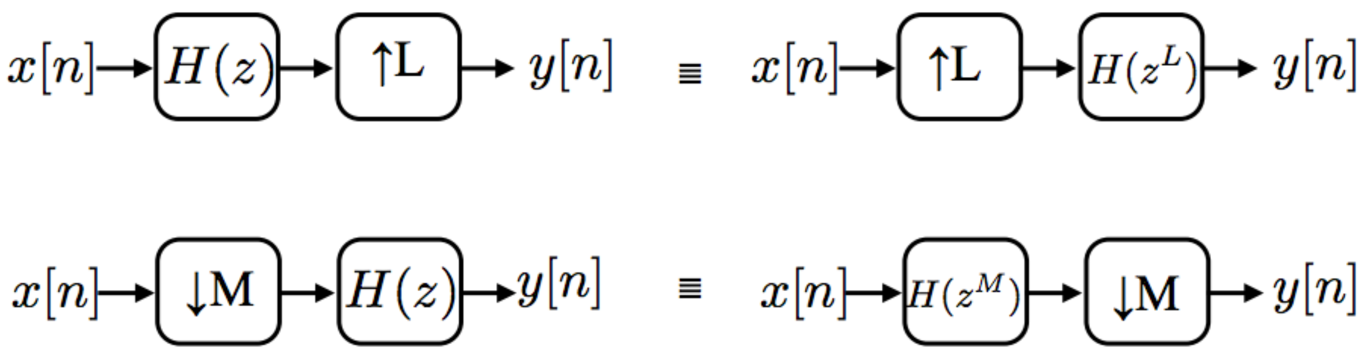
Multi-Rate Signal Processing

- What if we want to resample by 1.01T?
  - Expand by L=100
  - Filter π/101 ($$$$$)
  - Compress by M=101



Big Ideas

- Non-integer Resampling
- Multi-Rate Processing
  - Interchanging Operations



Admin

- HW 4 due Sunday