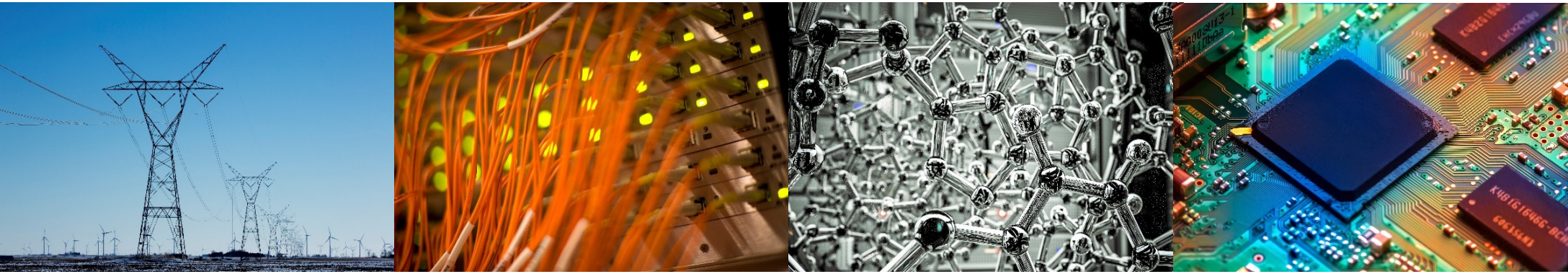
ECE 420- Embedded DSP Laboratory

Lecture 5 – Pitch Synthesis

Thomas Moon

September 19, 2022



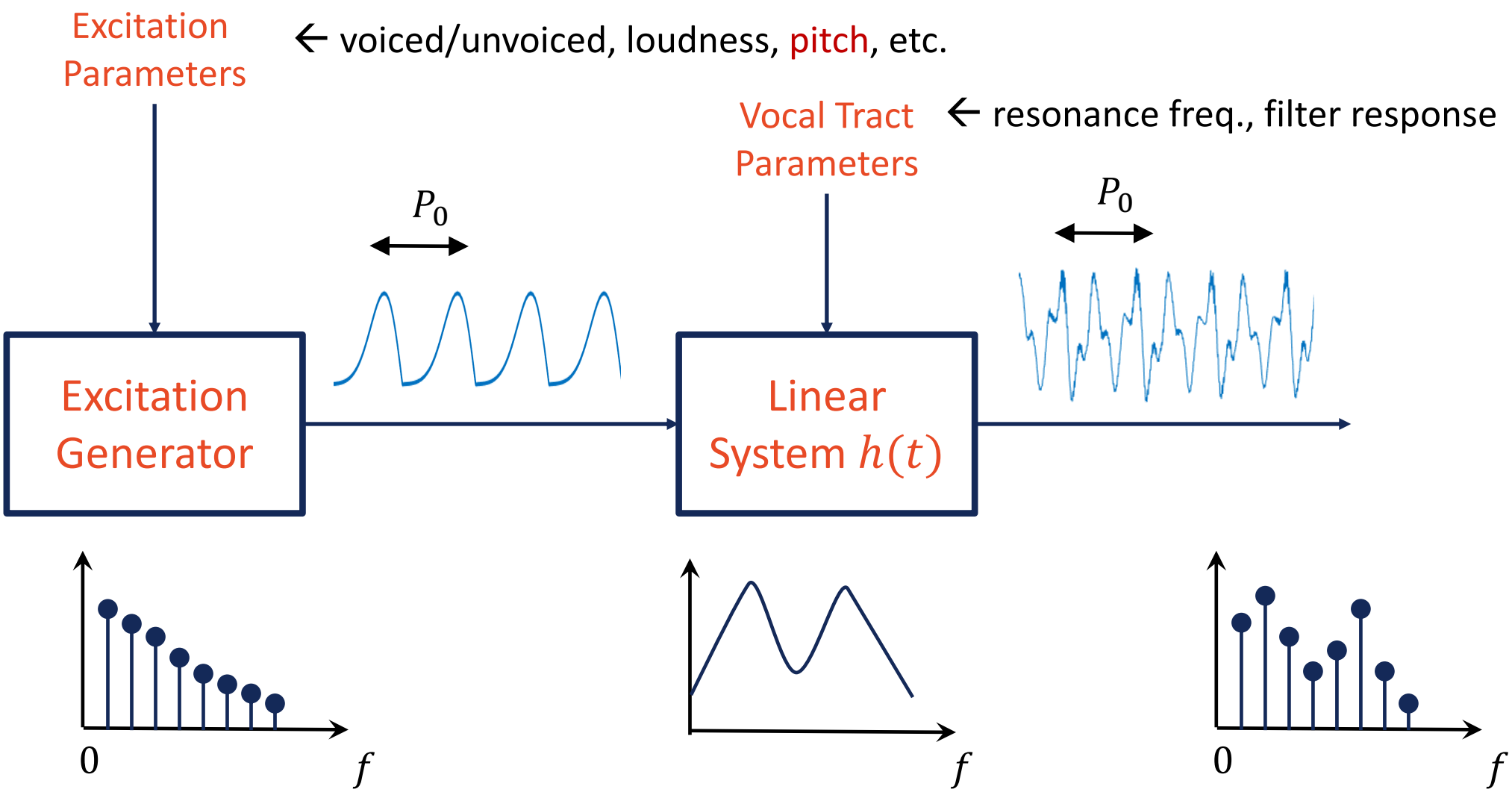
ILLINOIS
Electrical & Computer Engineering
GRAINGER COLLEGE OF ENGINEERING

Lab Summary So Far

- Lab4 (demo this week)
  - Speech signal → Pitch detection
    1. Energy
    2. Autocorrelation

- Lab5 (demo next week)
  - Spech signal → Pitch modification
    1. Resampling (Upsampling+Downsampling)
    2. TD-PSOLA

Source-filter Model



source spectrum

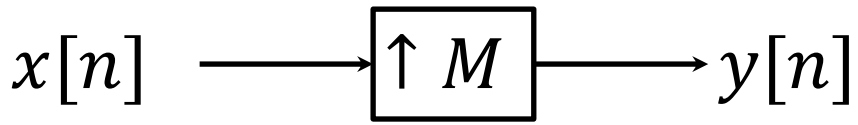
filter response
(one or more resonances)

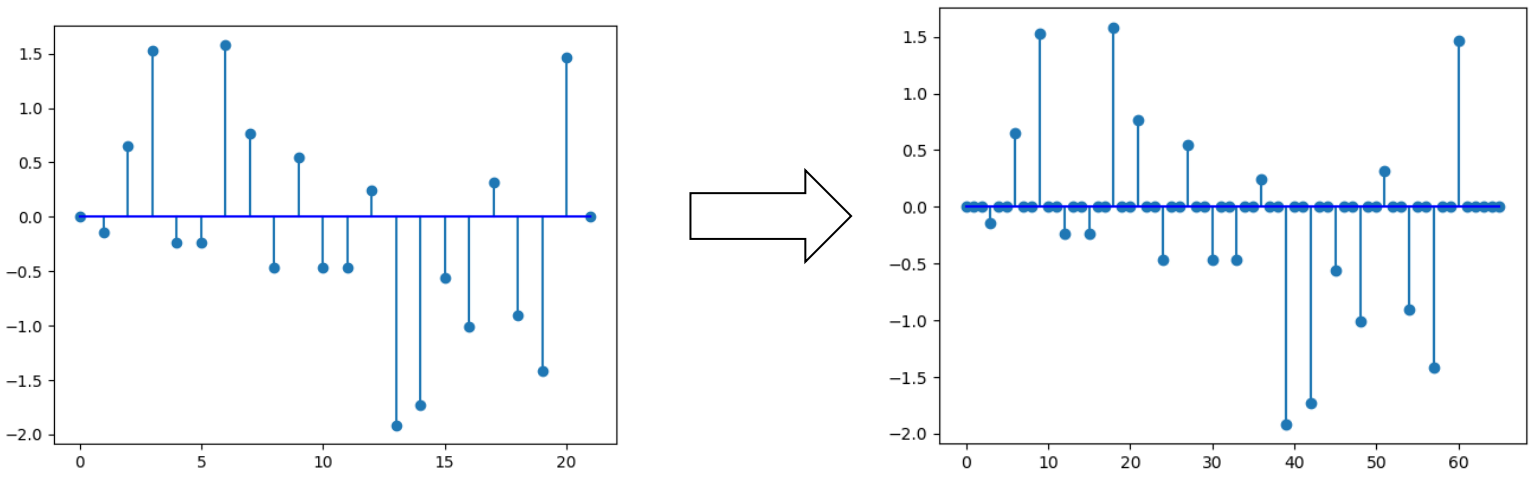
output spectrum

How can we modify P_0?

Recap from ECE310: Upsampling

- Performs zero insertion on the signal
  - Add M-1 zeros between each sample
- Always ‘safe’ as we do not lose any data





Upsampling – Frequency Domain

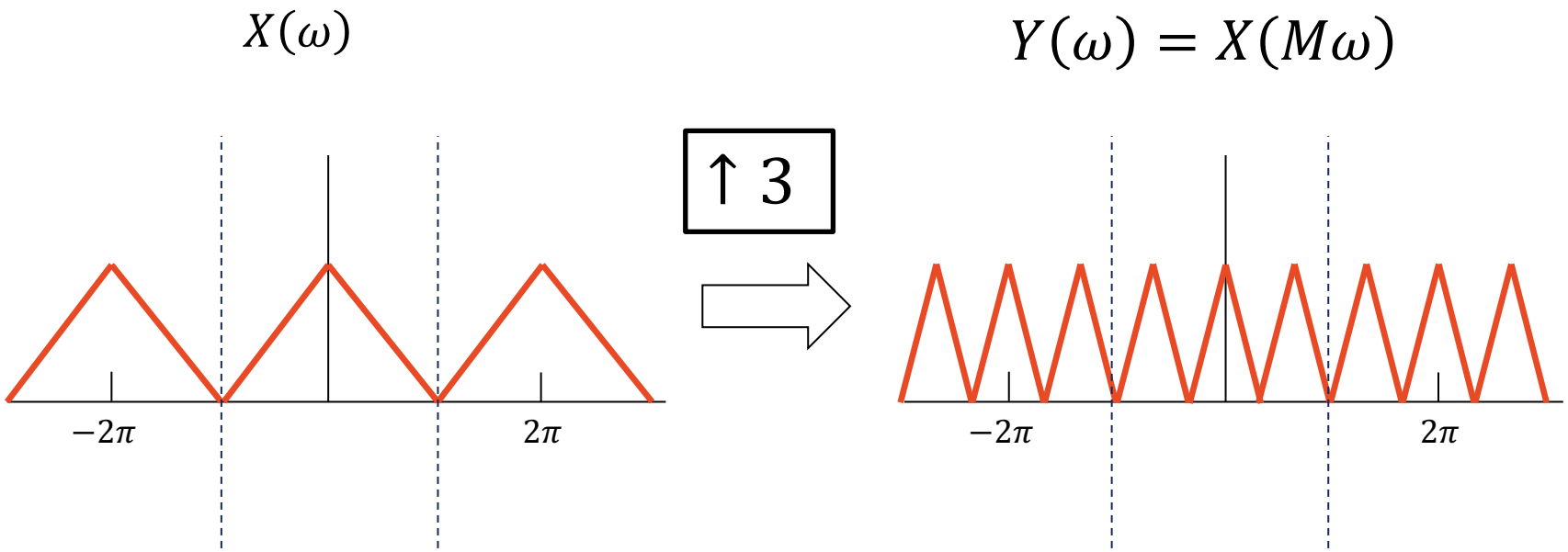
$$y[n] = \begin{cases} x[n/M], & n = 0, \pm M, \pm 2M, \ldots \\ 0\,, & otherwise \end{cases}$$

$$\begin{aligned} Y(\omega) &= \sum_{n=-\infty}^{\infty} y[n] e^{-j\omega n} \\ &= \sum_{n=-\infty}^{\infty} x[n/M] e^{-j\omega n} \\ &= \sum_{l=-\infty}^{\infty} x[l] e^{-j\omega M l} \\ &= X(M\omega) \end{aligned}$$

$$X(\omega) = \sum_{l=-\infty}^{\infty} x[l] e^{-j\omega l}$$

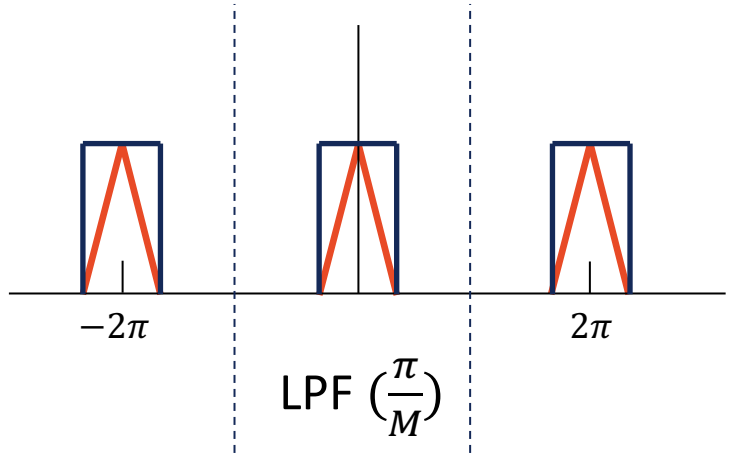
Compress ω by M

Upsampling – Frequency Domain

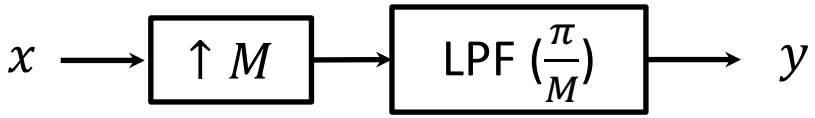


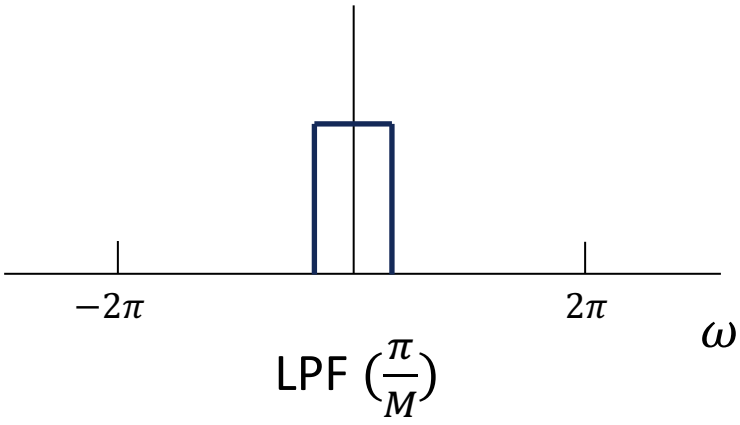
Compress ω by M

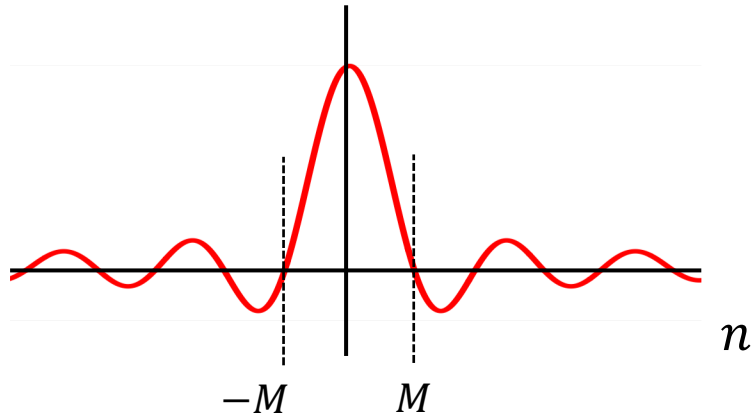
How to eliminate aliased spectra?

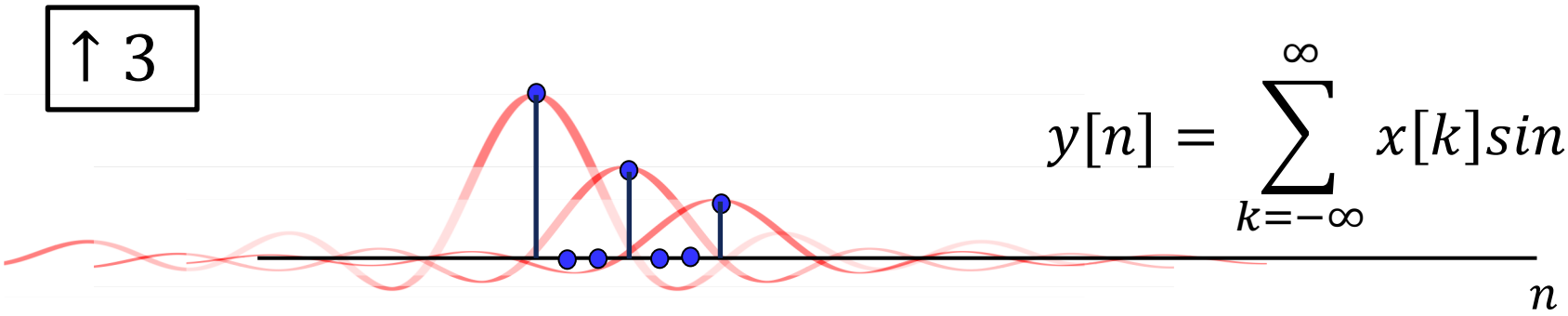


Upsampling with Interpolation







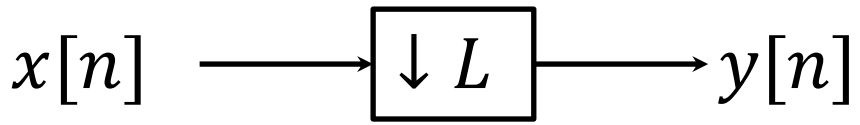


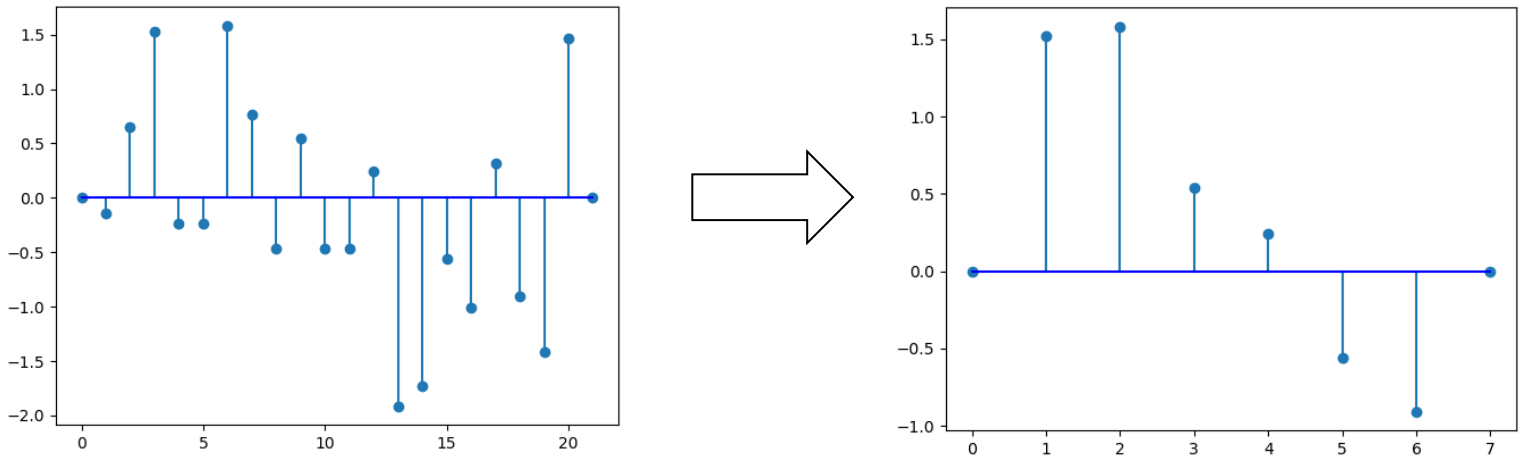
$$y[n] = \sum_{k=-\infty}^{\infty} x[k] sinc\left(\frac{\pi(k - kM)}{M}\right)$$

Fill in the missing samples by interpolation kernel

Recap from ECE310: Downsampling

- Reduce the number of samples in the signal
  - Keep first sample out of every batch of L samples
- Potentially unsafe as we are discarding samples





Downsampling – Frequency Domain

$$y[n] = x[Ln]$$

$$Y(\omega) = \sum_{n=-\infty}^{\infty} y[n]e^{-j\omega n}$$

$$= \sum_{n=-\infty}^{\infty} x[Ln]e^{-j\omega n} \qquad \text{(Let } l = Ln\text{)}$$

$$= \sum_{\substack{l=-\infty \\ l \text{ is } L-multiple}}^{\infty} x[l]e^{-j\omega l/L}$$

$$= \sum_{l=-\infty}^{\infty} x[l]r[l]e^{-j\omega l/L}$$

$$= \sum_{l=-\infty}^{\infty} x[l]\left(\frac{1}{L}\sum_{n=0}^{L-1} e^{j2\pi ln/L}\right)e^{-j\omega l/L}$$

$$= \frac{1}{L}\sum_{n=0}^{L-1} X\left(\frac{\omega - 2\pi n}{L}\right)$$

Let $r[l] = \begin{cases} 1, & l = Ln \\ 0, & otherwise \end{cases}$

$$= \frac{1}{L}\sum_{n=0}^{L-1} e^{j2\pi ln/L}$$

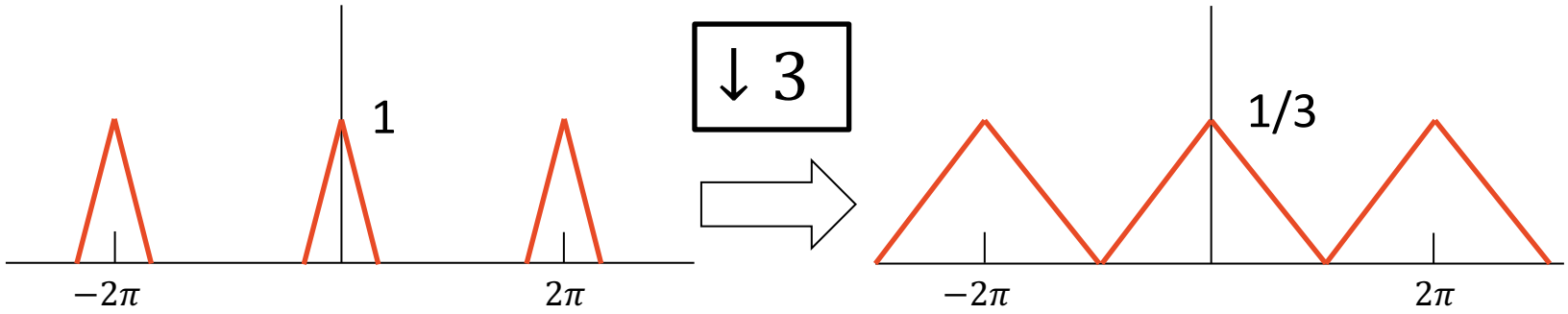
$$= \frac{1}{L}\sum_{n=0}^{L-1}\sum_{l=-\infty}^{\infty} x[l]e^{-jl\left(\frac{\omega-2\pi n}{L}\right)}$$

stretch ω by L and shift by $2\pi n$

Downsampling – Frequency Domain

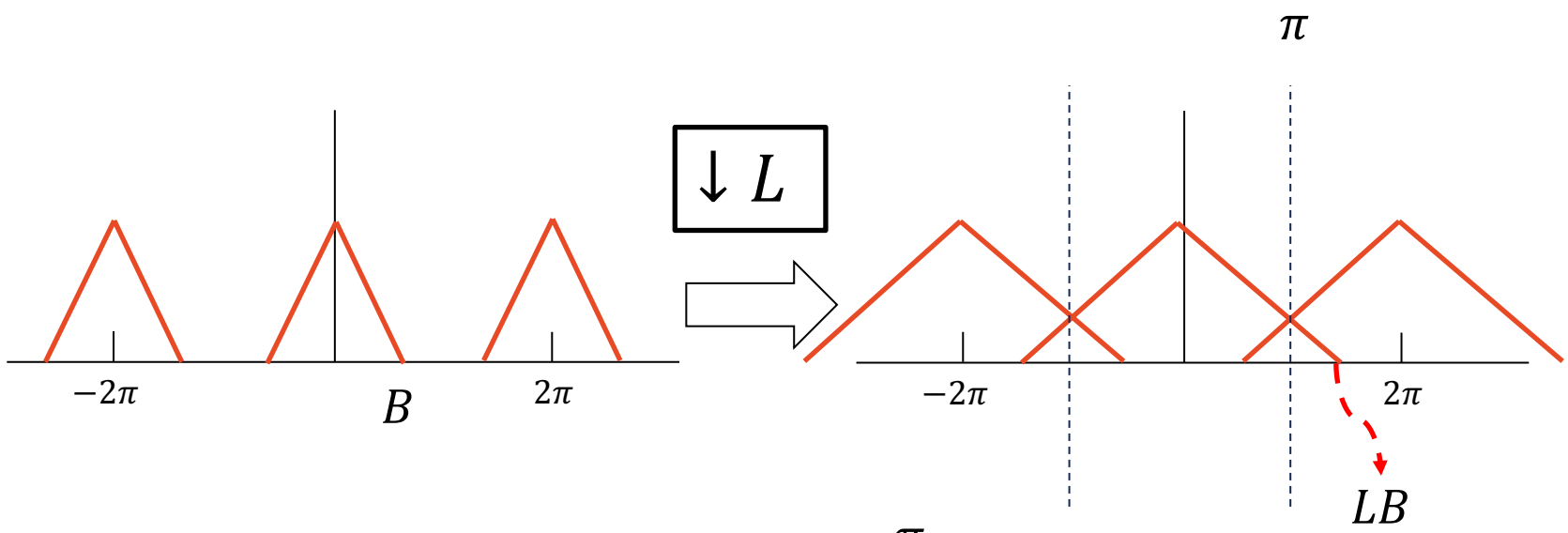
$X(j\omega)$

$$Y(\omega) = \frac{1}{L}\sum_{n=0}^{L-1} X\left(\frac{\omega - 2\pi n}{L}\right)$$

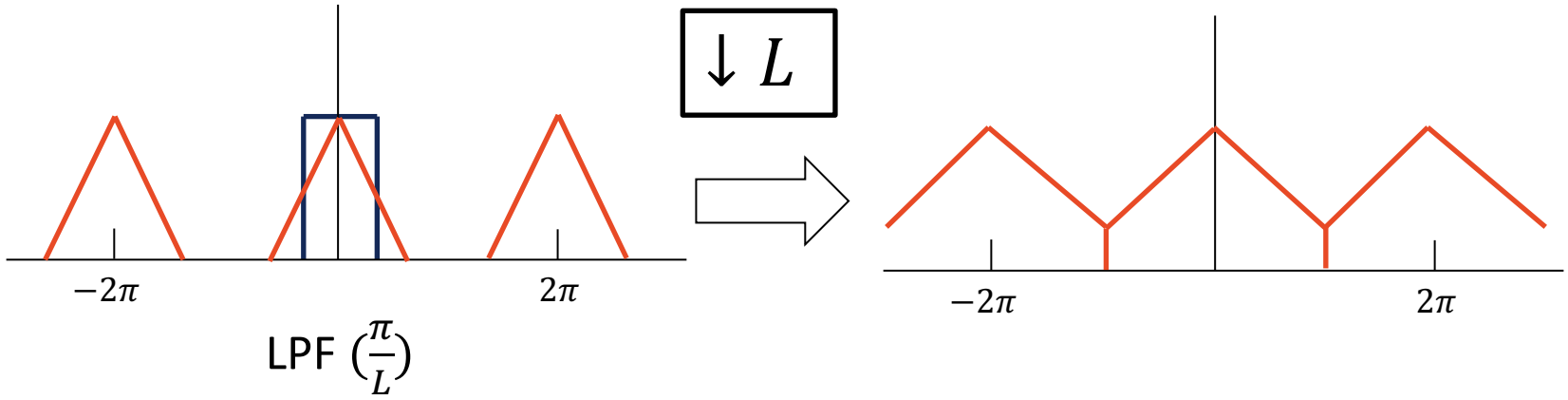


stretch ω by L and shift by $2\pi n$

Downsampling - Prevent Aliased Spectra



$$LB < \pi \rightarrow B < \frac{\pi}{L}$$

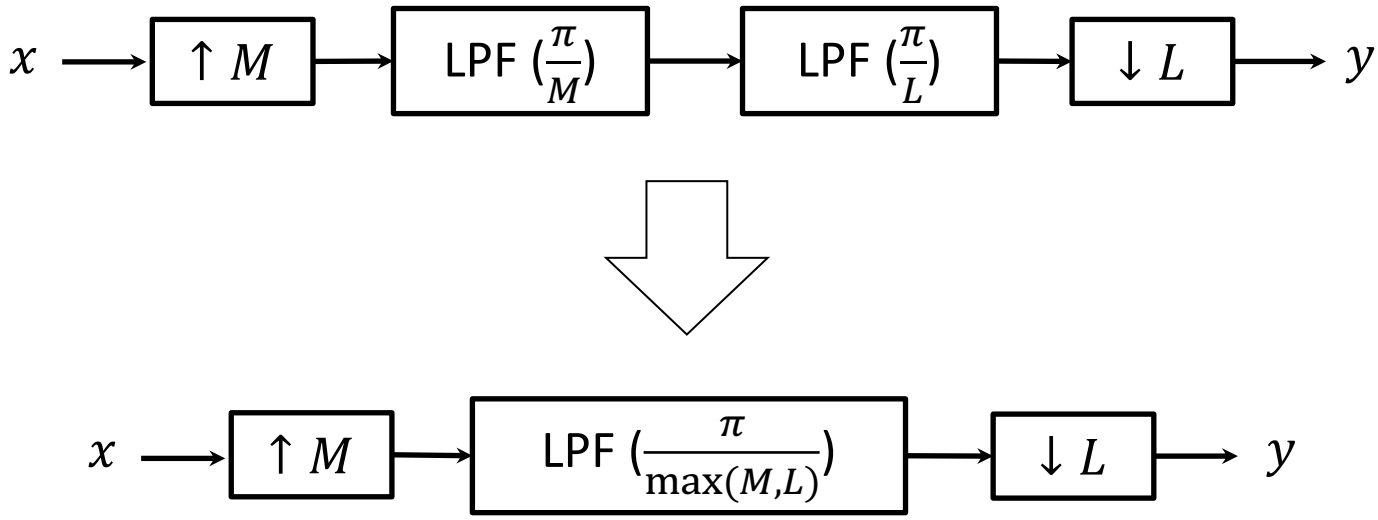


LPF $(\frac{\pi}{L})$

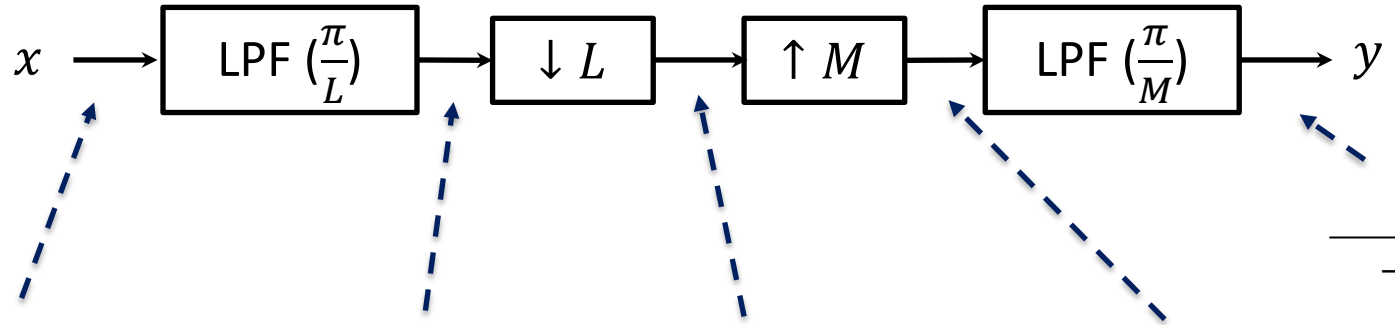
anti-aliasing filter

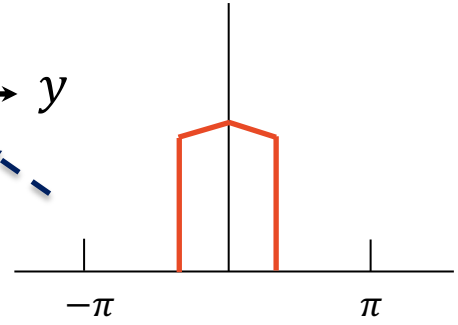
Fractional Rate

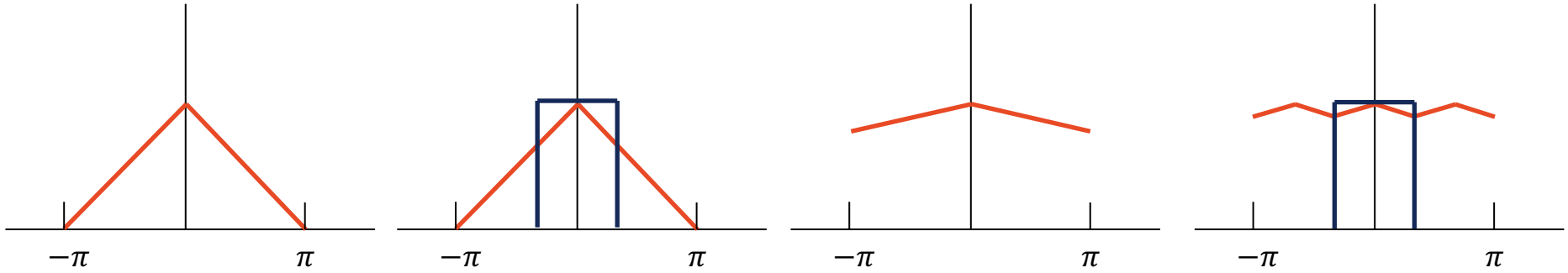
- Upsampling/downsampling operations defined for integer
- How can you implement arbitrary fractional rates?
  - Cascade of Upsampler (rate M) followed by Downsampler (rate L)
  - Effective rate change of M/L
  - Why upsampling first?



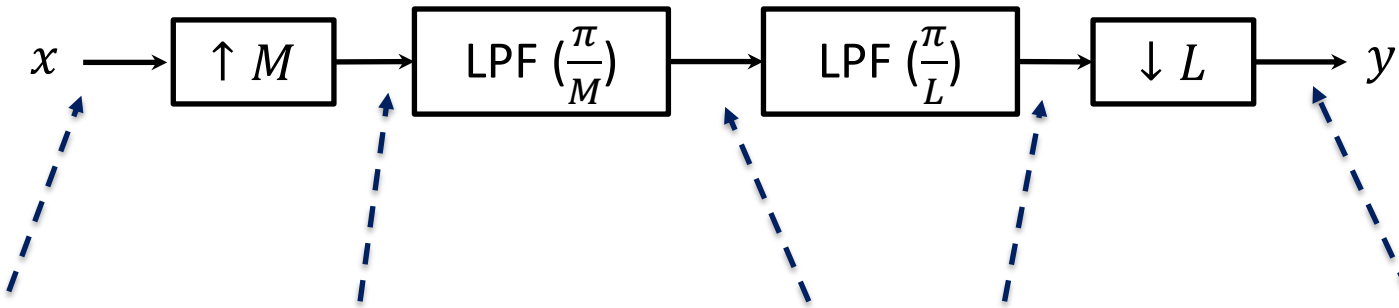
L = 3, M = 3

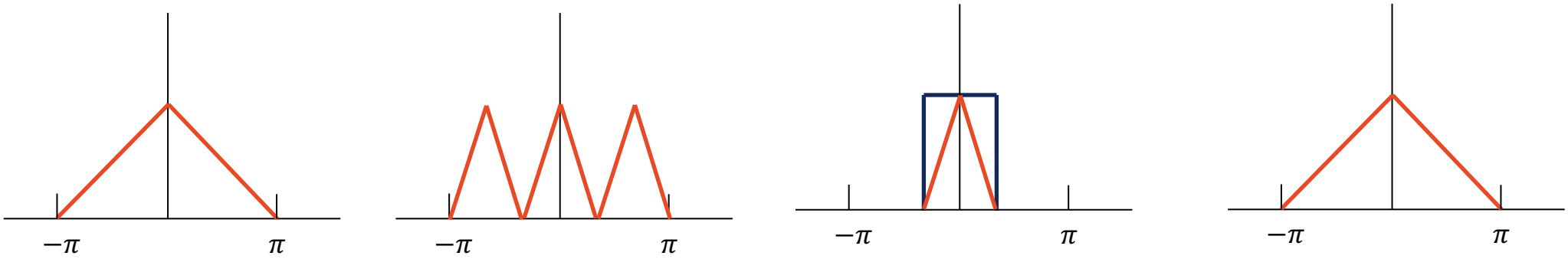




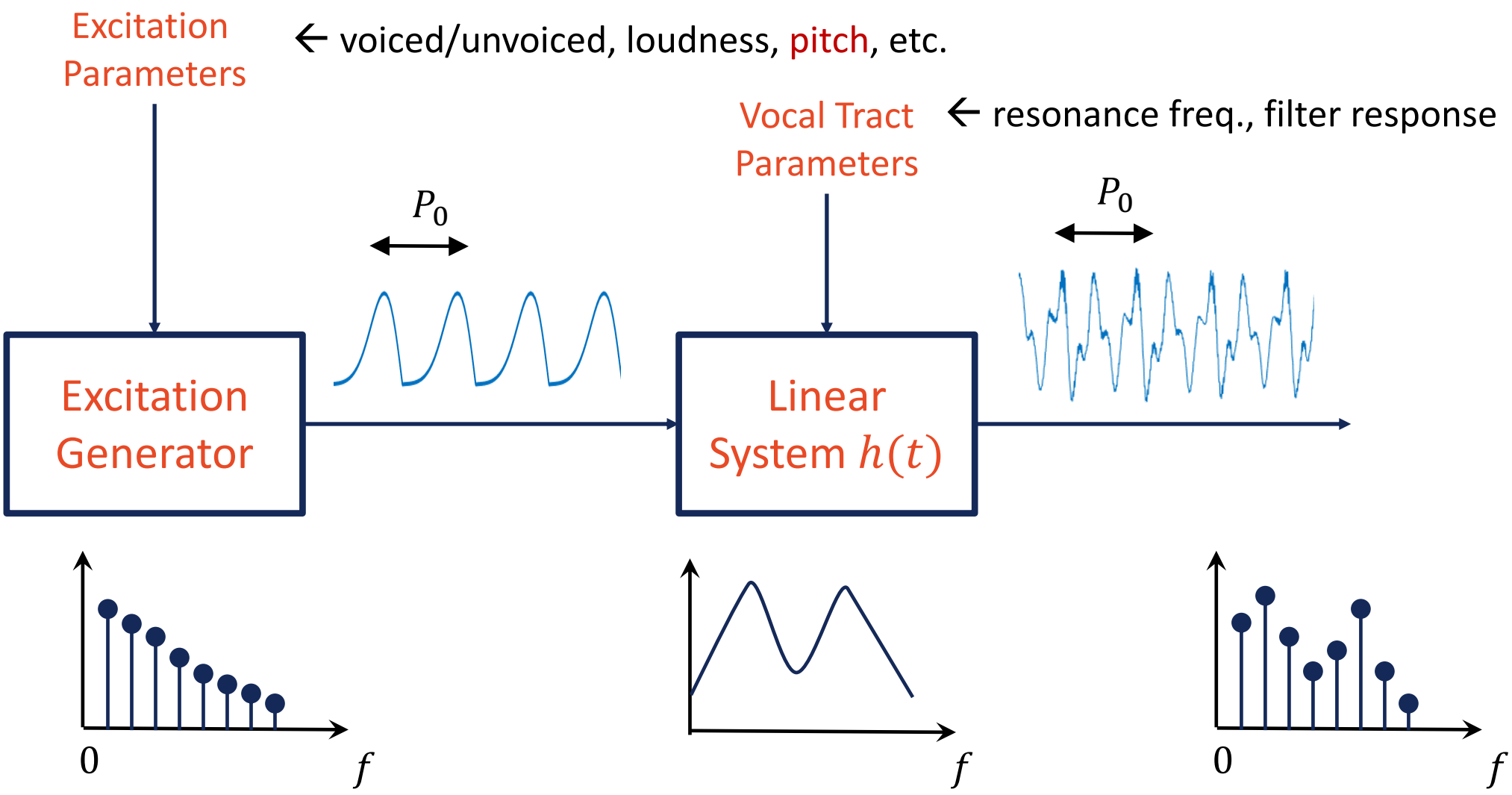


L = 3, M = 3





Source-filter Model



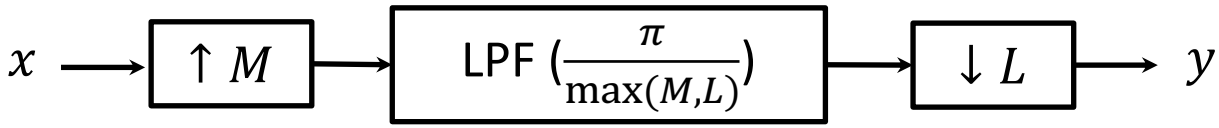
source spectrum

filter response
(one or more resonances)

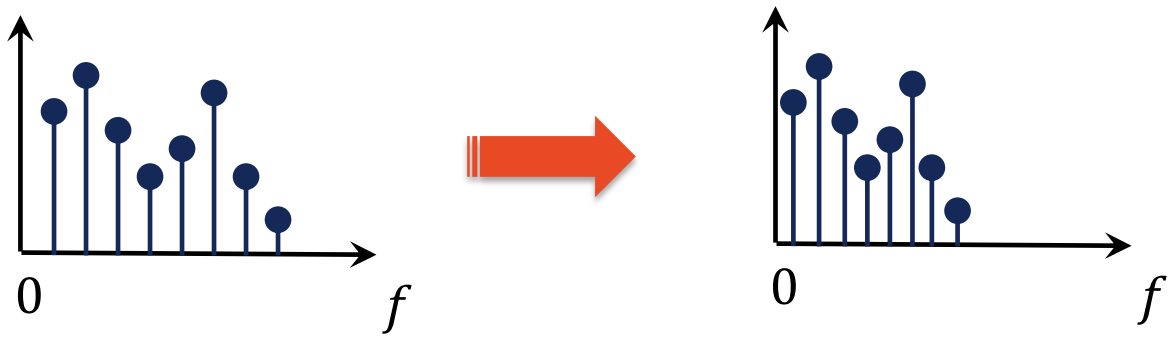
output spectrum

How can we modify P_0?

Modify Pitch by Resampling



$$P_1 = \frac{M}{L} P_0$$



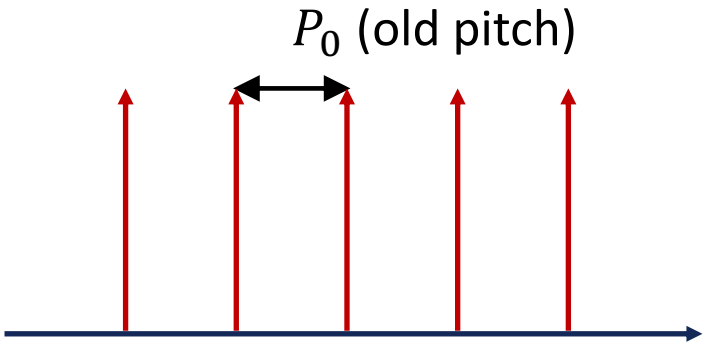
Stretch or compress the entire spectrum.

Both Pitch and Vocal Tract Response changed!

How can we modify only pitch?

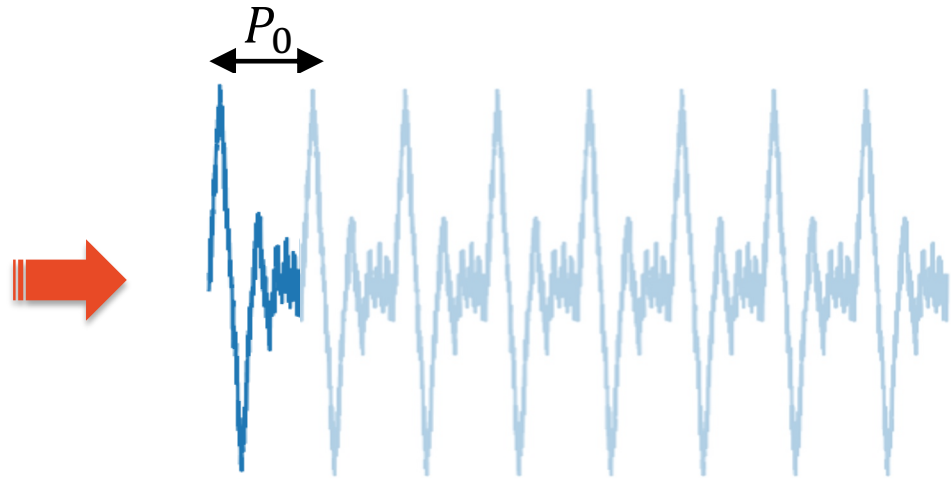
TD-PSOLA

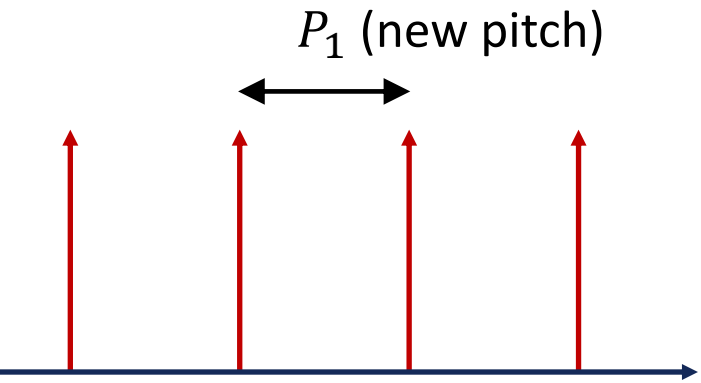
- TD-PSOLA can modify the fundamental pitch without affecting the formants (vocal tract response)
- TD – Time domain
- PS – Pitch Synchronous
  - Operate around reference points (epoch markers or pitch-marks)
- OLA – Overlap-Add
  - The synthesized signal overlaps and are added together to form the final output



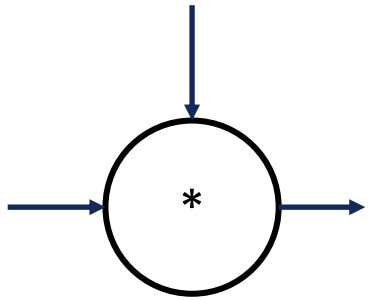
$x[n] = \sum \delta[n - P_0 k]$

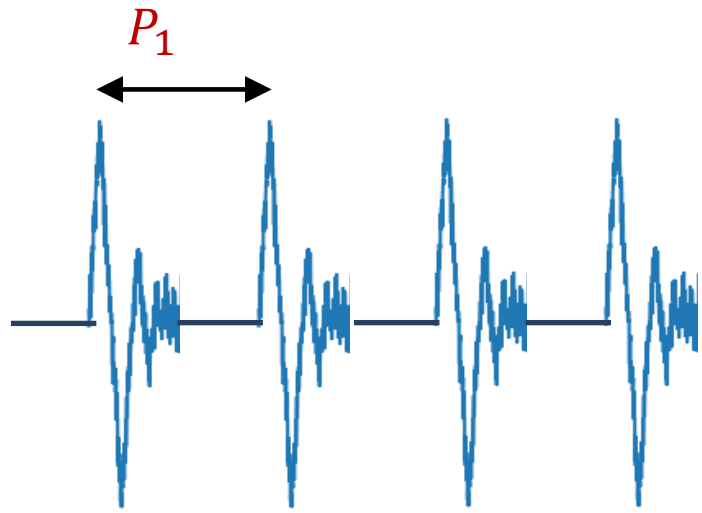
*





$\hat{x}[n] = \sum \delta[n - P_1 k]$





$\hat{y}[n] = \hat{x}[n] * h[n] = \sum h[n - P_1 k]$

Challenges

Signal analysis

1. Pitch changes over time
2. $h[n]$ changes over time

Signal synthesis

1. Discontinuity/distortions while synthesizing the output signal
2. Block processing of the audio frames

TD-PSOLA Algorithm

Find Epochs

↓

Epoch Mapping

↓

↓

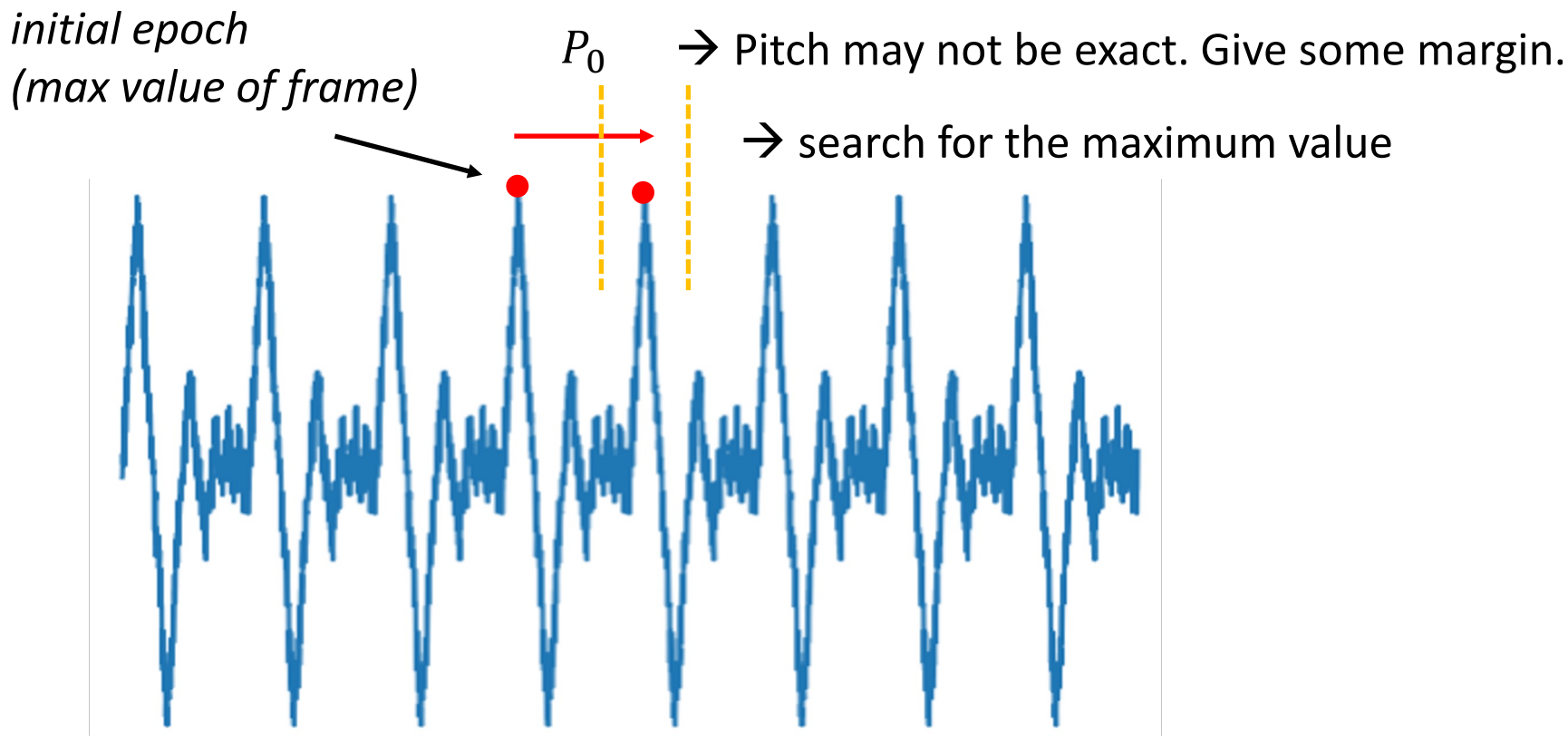
Synthesis by windowing

↓

Past-Present-Future buffering

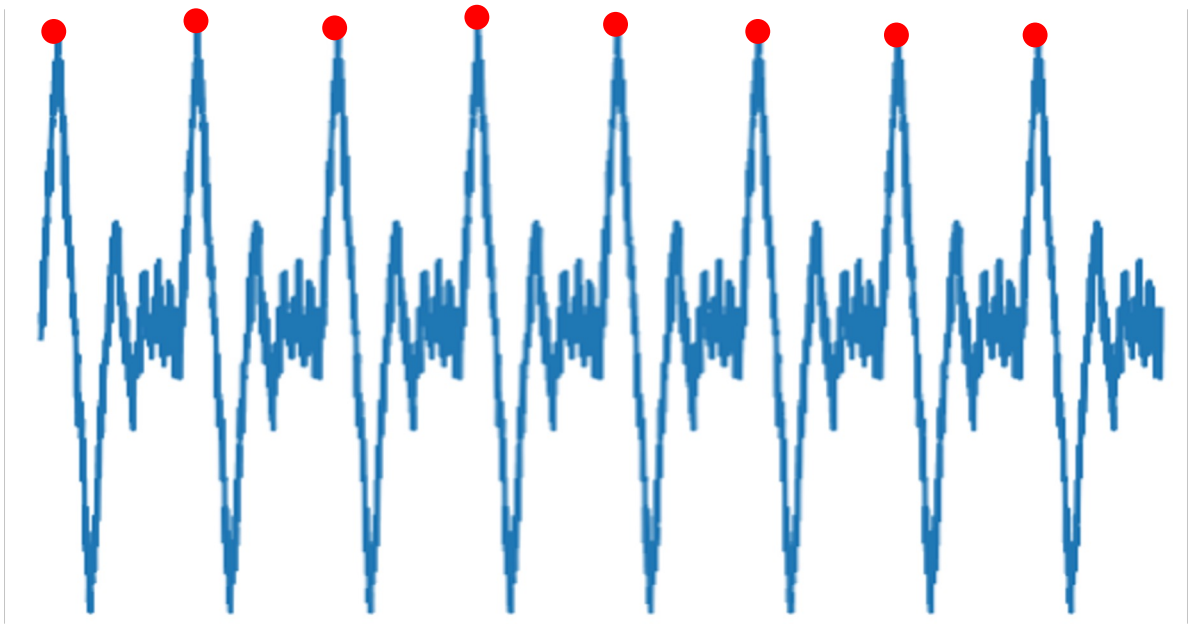
1. Find Epochs

- Peaks of the signal
- Epochs provide the reference point to operate.
- Estimate pitch period from lab4 algorithm



1. Find Epochs

- Peaks of the signal
- Epochs provide the reference point to operate.
- Estimate pitch period from lab4 algorithm



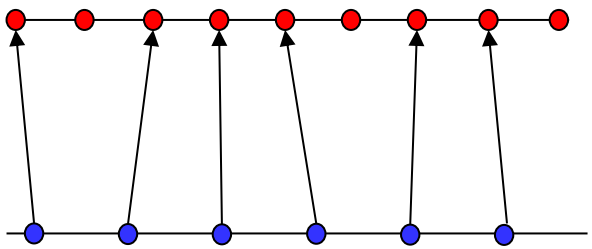
Implemented in `findEpochLocations`

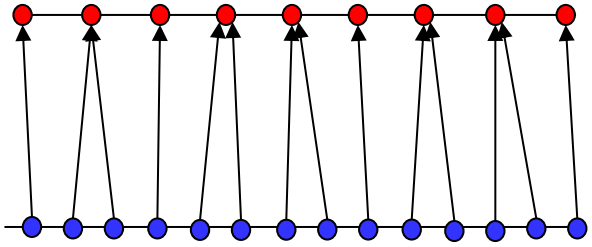
2. Epoch Mapping

- Input epochs: by pitch & waveform analysis
- Output epochs: regularly spaced positions at target pitch
- Algorithm: For each output epoch location, find the nearest input epoch location

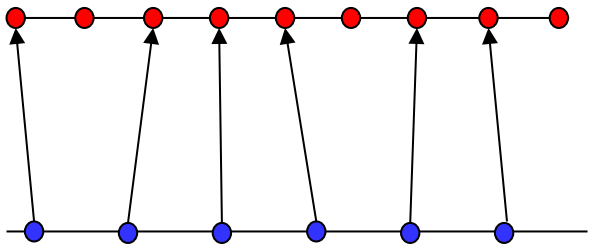
input epochs

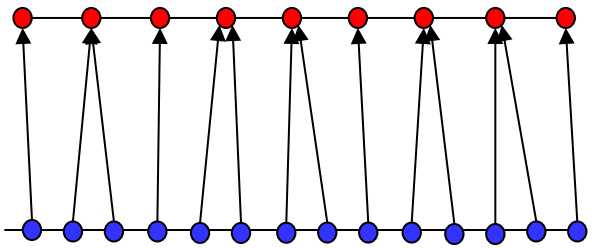
output epochs

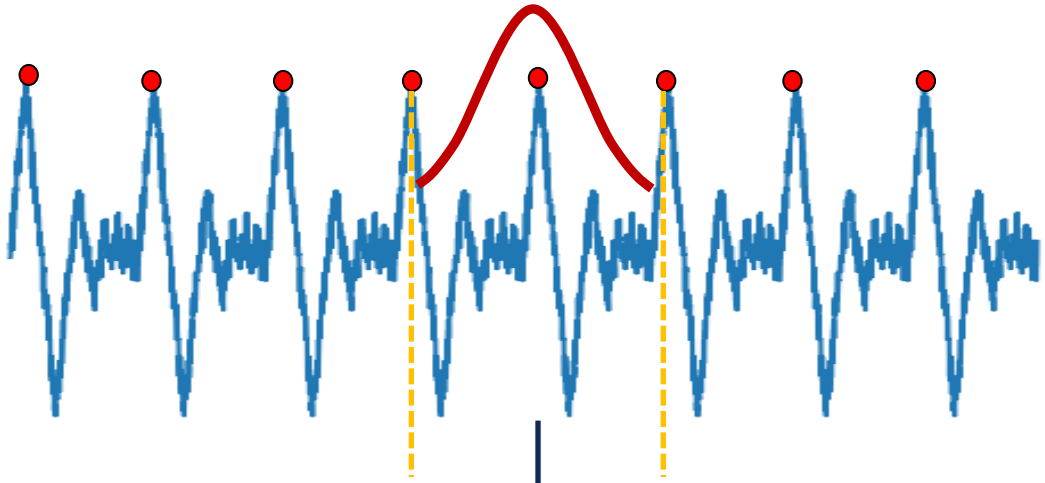




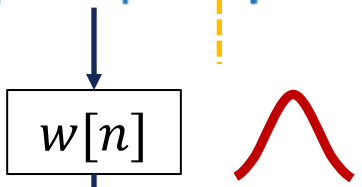
3. Signal Synthesis by Windowing

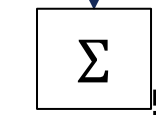


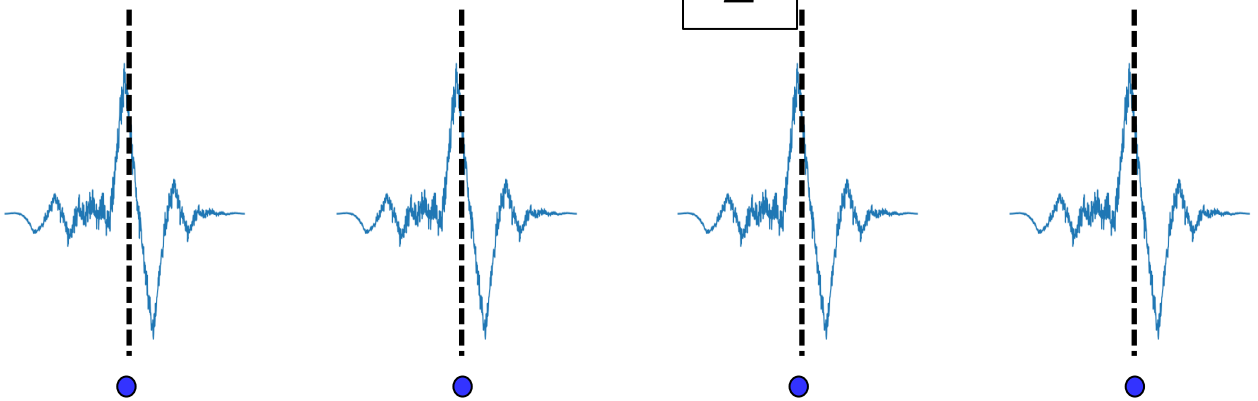




→ Apply window over <u>two adjacent periods</u>







→ Position at output epoch points & Combine all outputs together to form synthesized signal

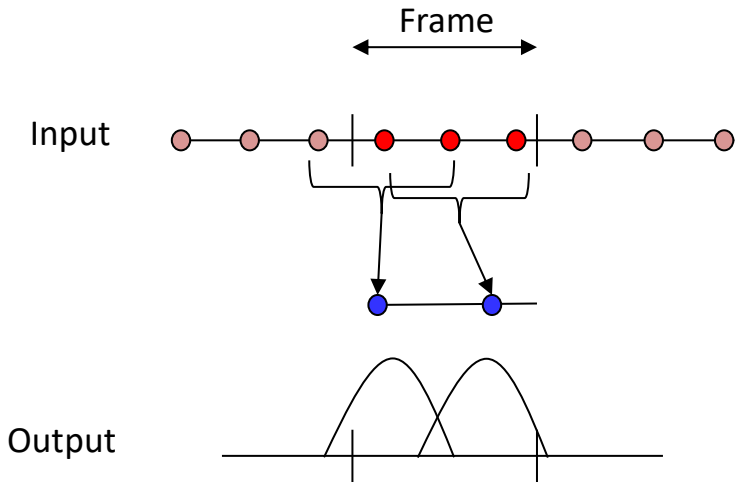
4. Block Processing Challenges

- Data is broken up into blocks/frames of data for processing due to practical reasons
  - Memory
  - Responsiveness
- Depending on the algorithm, there may be dependencies among blocks of data
- How can we address this problem?
  - Buffering!

4. PSOLA Block Processing

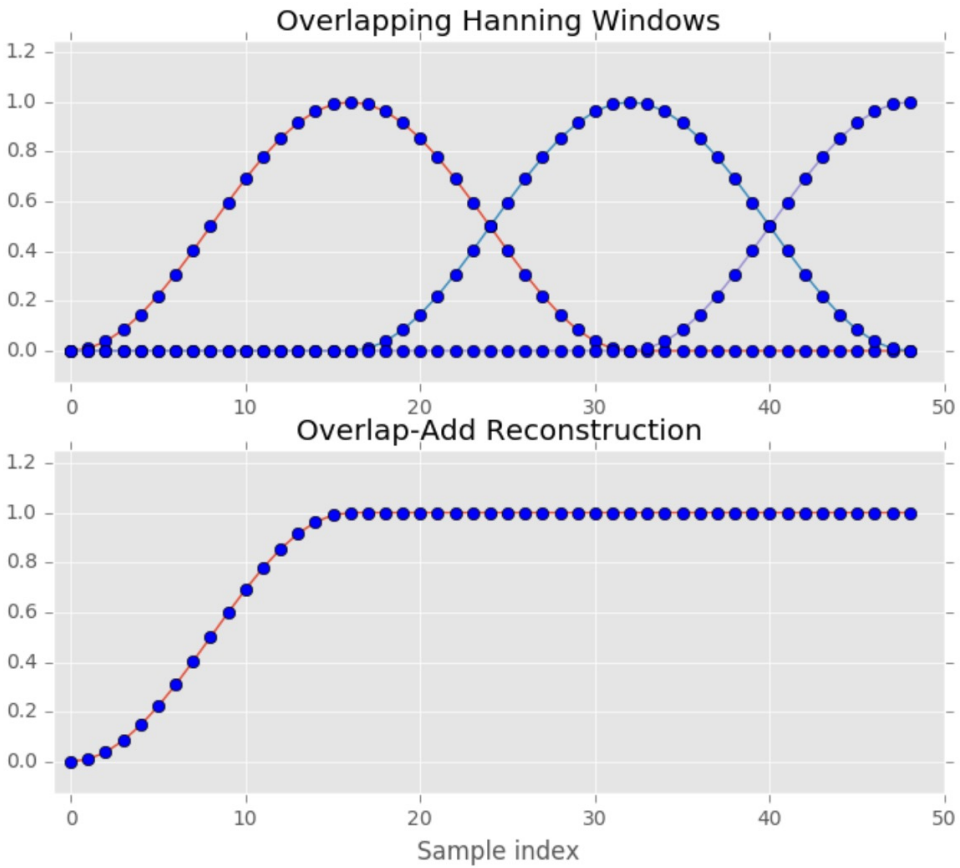
- Two main issues that arise from framing the data
  1. Depending on epochs selected, windowed interval may stretch across multiple input frames
  2. After repositioning on output epoch location, windowed response may stretch across multiple output frames

1. window is out of current **input** frame



2. window is out of current **output** frame

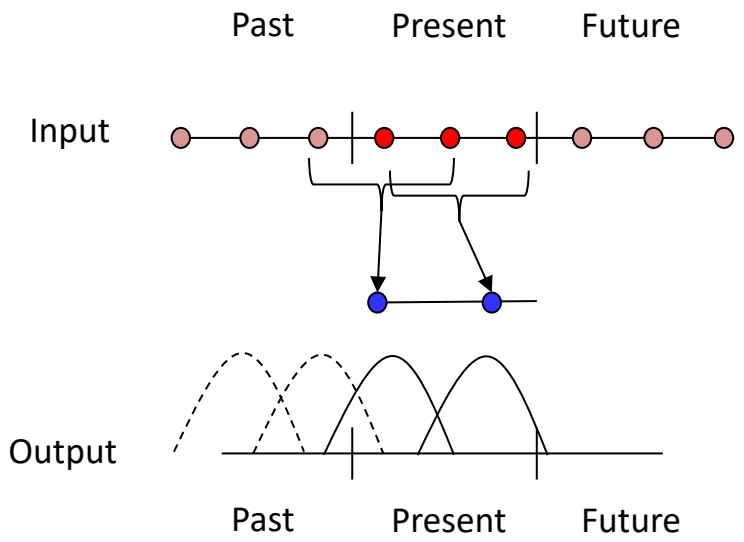
4. PSOLA Block Processing



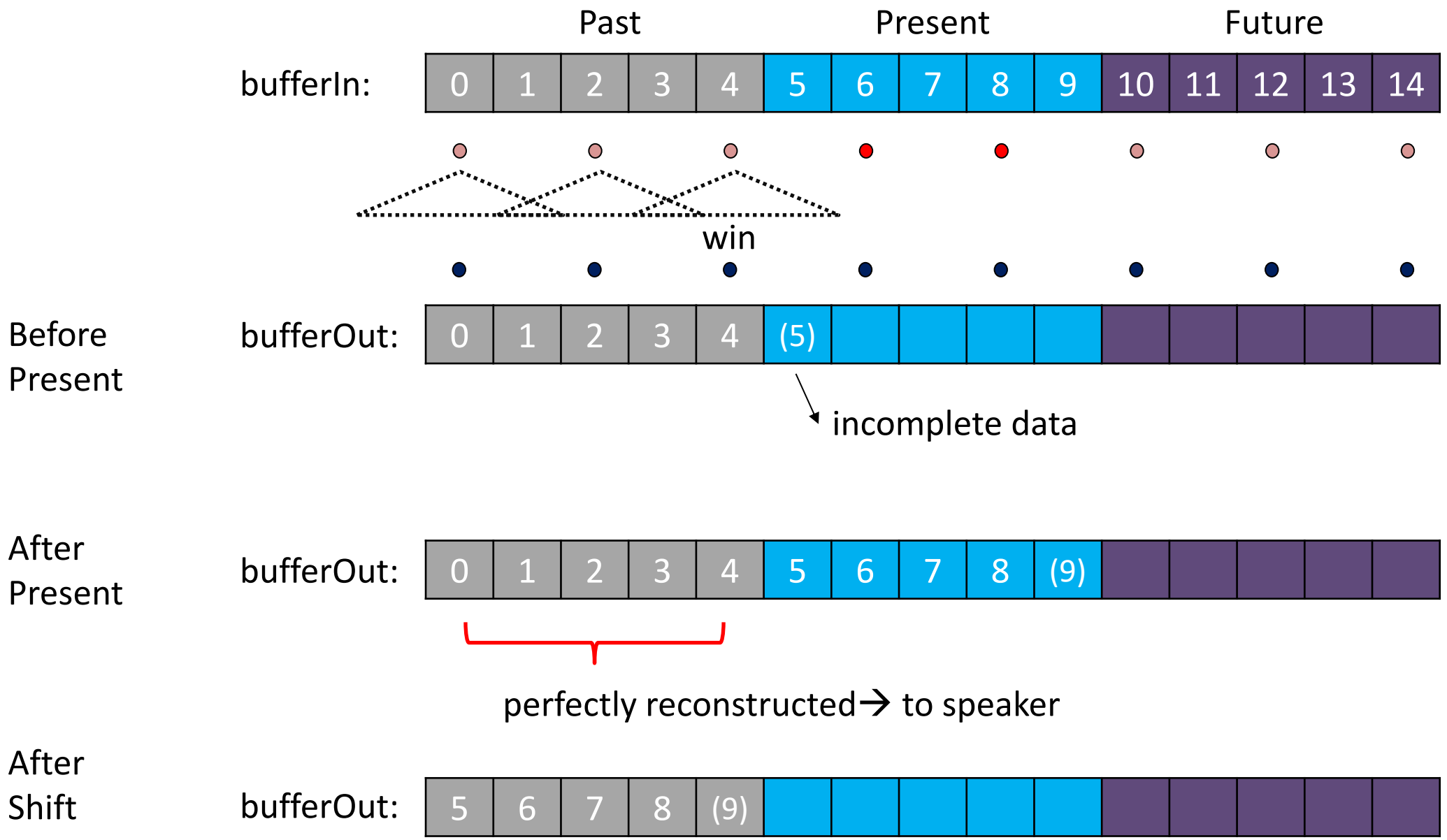
Without fixing the issues
→Discontinuity across the multiple buffers

4. PSOLA Block Processing

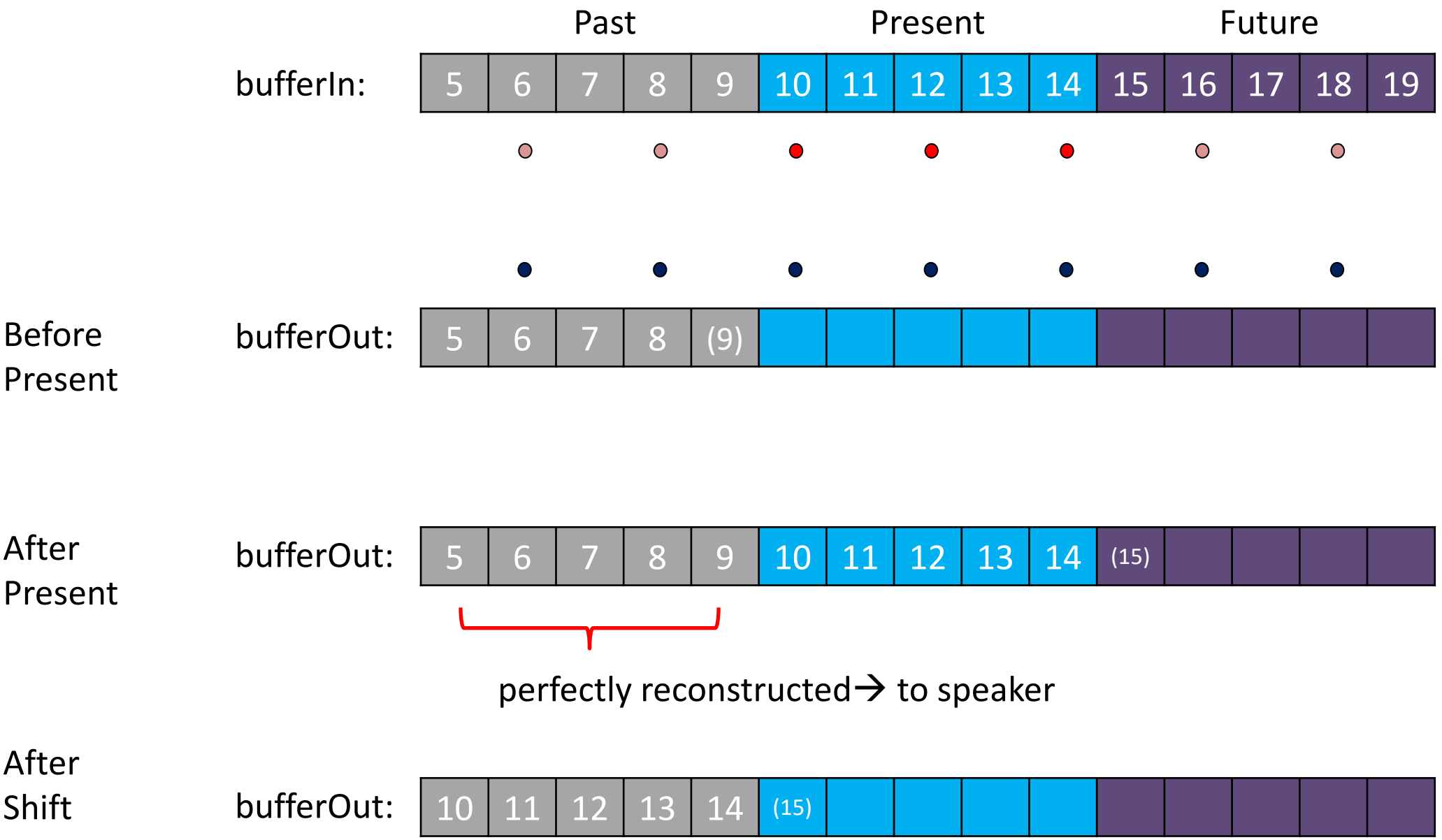
- Approach: Keep buffer of input blocks and output blocks: ‘past’, ‘present’, and ‘future’
- Determine contributions for output epoch points in the ‘Present’
- Allow impulse response to spill over into ‘Past’ and ‘Future’
- After all ‘Present’ points processed ‘Past’ will be complete, ready to emit
- Shift down Present to Past and Future to Present



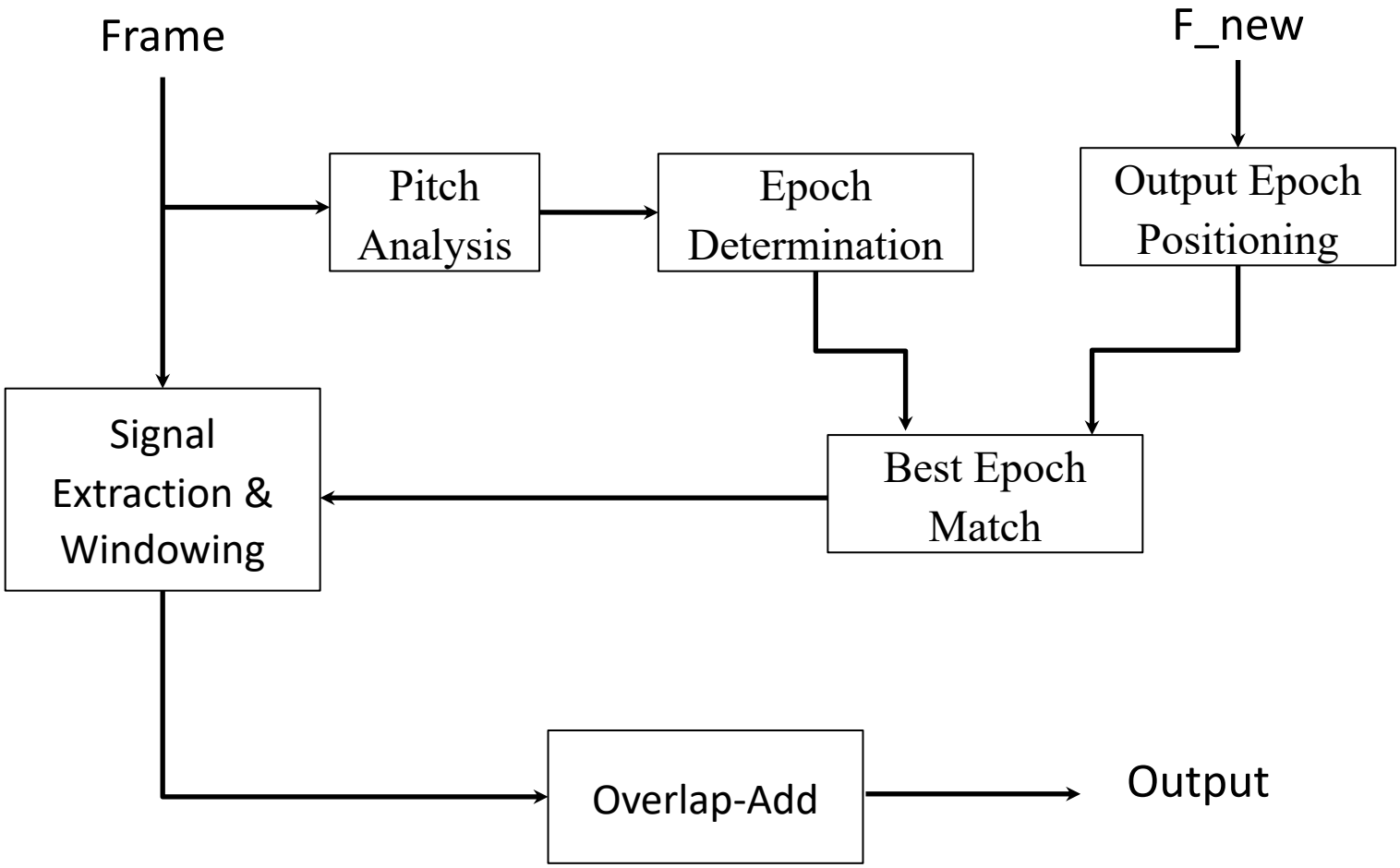
Example : $P_0 = P_1$



Example : $P_0 = P_1$



Pitch Synthesis Algorithm



We learned…

- Upsampling VS Downsampling
  - change both Pitch and Vocal tract response
- TD-PSOLA
  - change only Pitch