Multirate Signal Processing* Tutorial using MATLAB**

By, Deborah Goshorn
dgoshorn@cs.ucsd.edu

I. Signal processing background

II. Downsample Example

III. Upsample Example

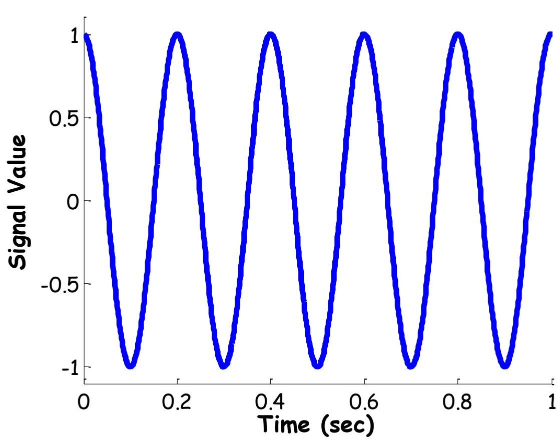
* Multrate signal processing is used for the practical applications in signal processing to save costs, processing time, and many other practical reasons.

** MATLAB is an industry standard software which performed all computations and corresponding figures in this tutorial

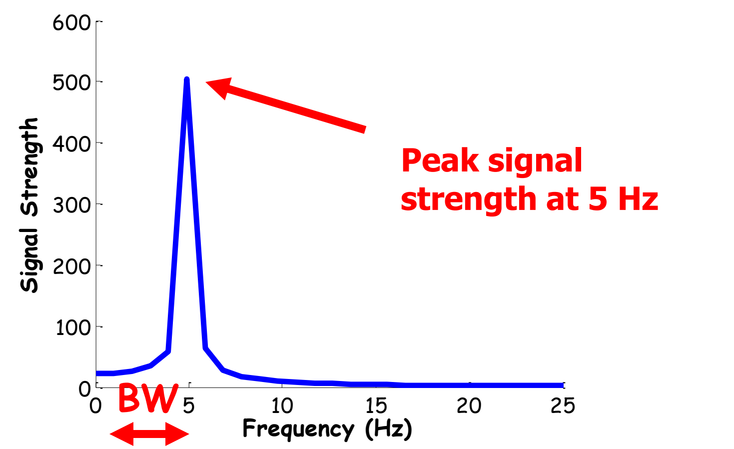
I. Signal processing background

Receive an analog signal

- Receive an analog signal at **5** Hz
  (as pictured below left, there are **5** wave cycles in one second.)
- The highest frequency component (5 Hz) of the signal is called the signal's **<u>bandwidth</u>**, **BW**, **since in the examples in this presentation, the minimum frequency component is 0Hz.**
- This signal can be represented in two ways:



time representation (sec)

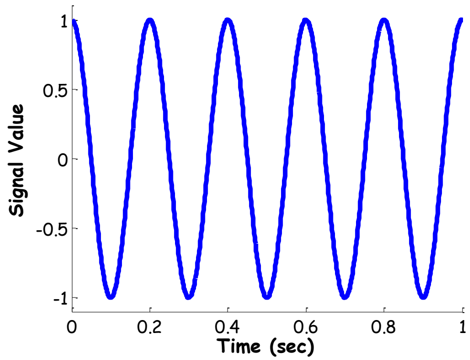


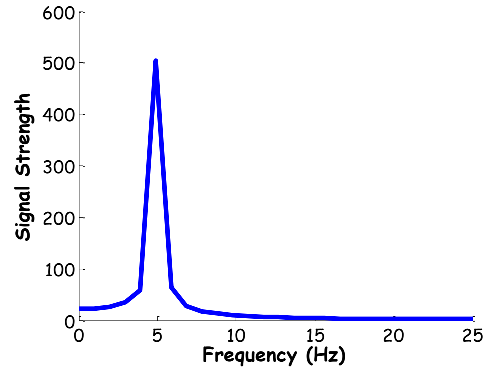
frequency representation (Hz)

Add high frequency components

1. Original 5 Hz signal

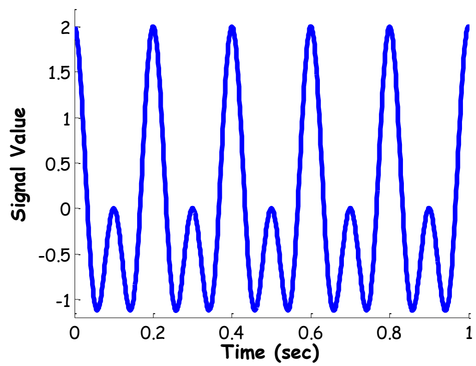
BW = 5 Hz

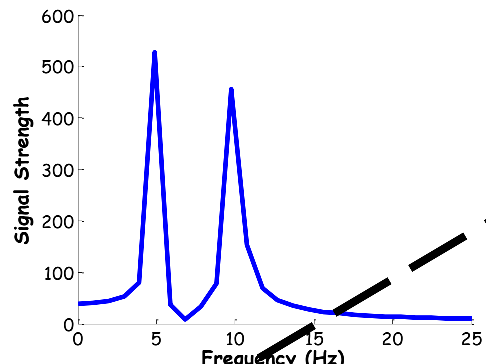




2. Add a 10 Hz component

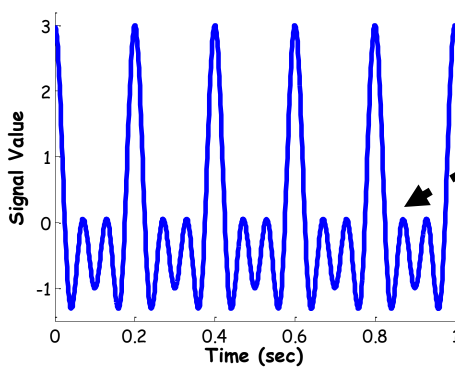
BW = 10 Hz

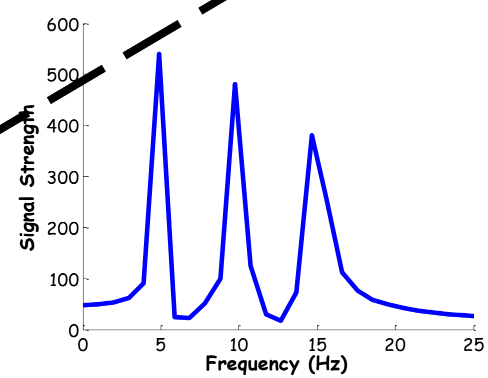




3. Then add a 15 Hz component!

BW = 15 Hz

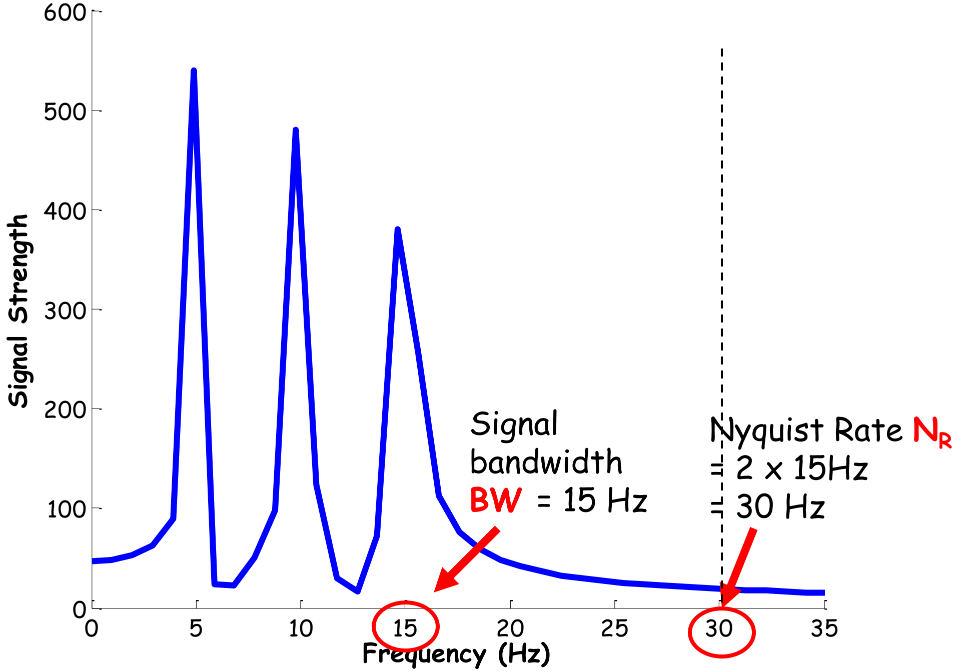




Adding high frequency components creates **jagged edges** in the original 5 Hz signal.

Sampling the signal: Nyquist Rate

- In order to sample the signal without losing information, use a sampling rate (S_R) of at least the Nyquist Rate (N_R), which is 2 x BW of the received analog signal.



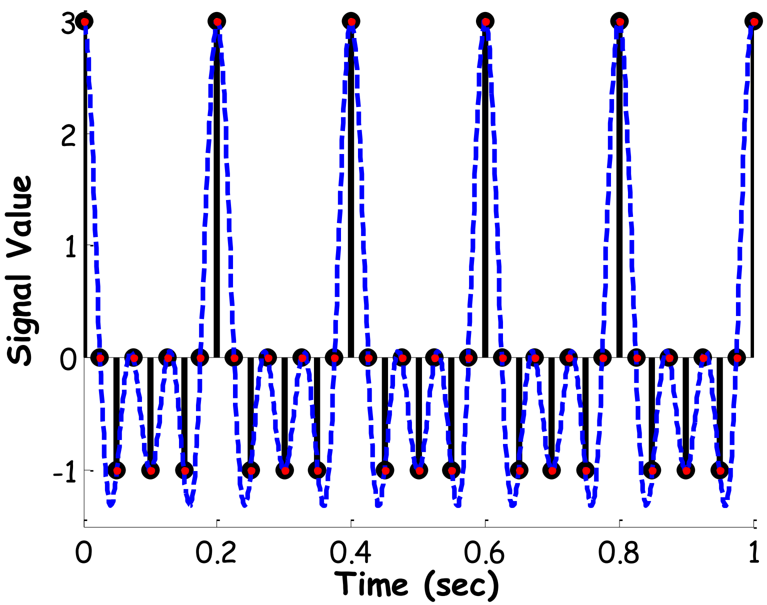
RULE: Sampling Rate S_R ≥ Nyquist Rate N_R

Sampling the signal: Nyquist Rate

Since Bandwidth **BW** = 15 Hz,

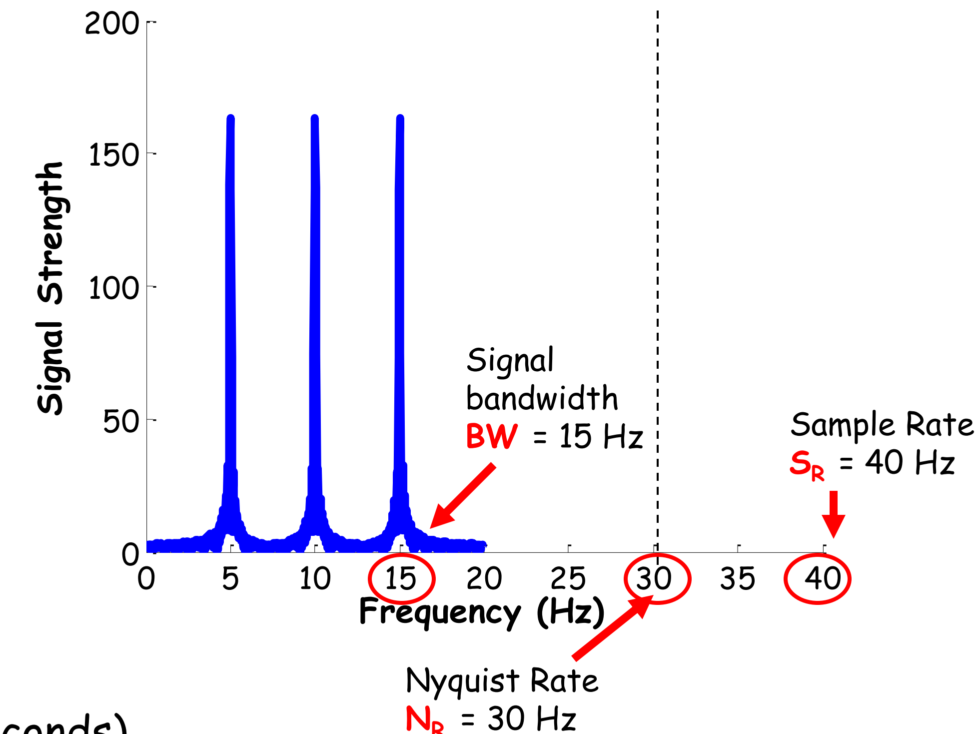
the Nyquist Rate N_R = 2 x 15Hz = 30Hz.

RULE #1: Sampling Rate S_R ≥ Nyquist Rate N_R



Let Sample Rate S_R = 40 Hz,

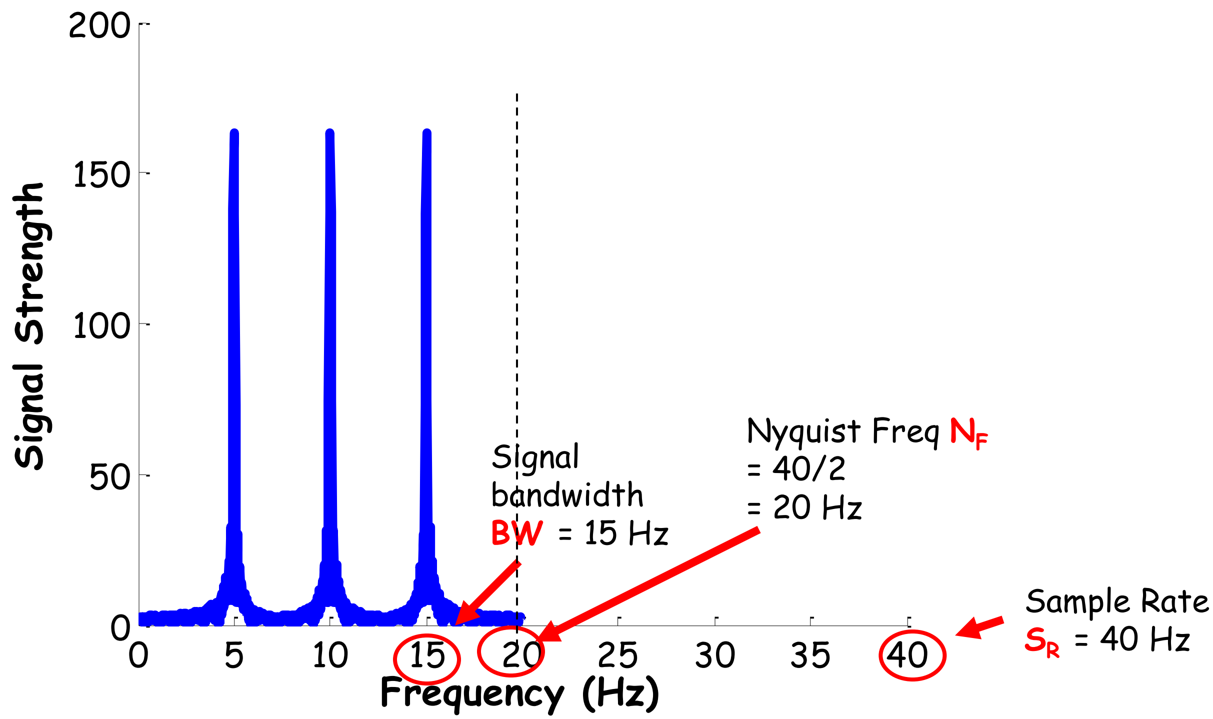
so sample signal every 0.025 sec (25 milliseconds).



Sampling the signal: **Nyquist Freq**

- The **Nyquist Frequency** (**N_F**) is equal to half of the sampling rate (**S_R**). The **N_F** must be equal to or greater than the bandwidth **BW** of the desired signal to reconstruct.

Rule #2: Nyquist Frequency **N_F** ≥ Bandwidth **BW**

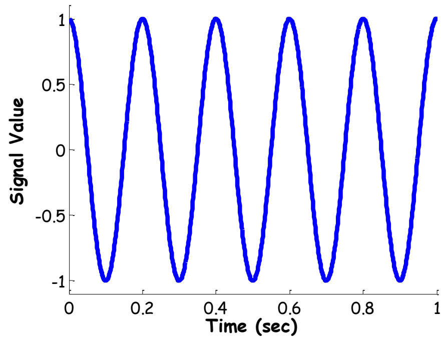


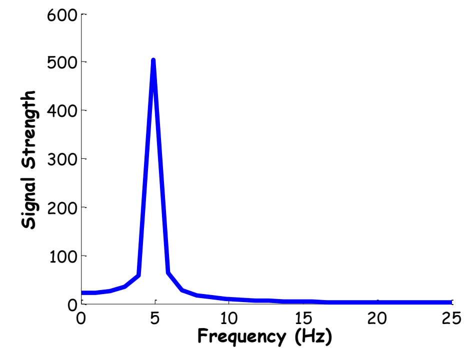
II. Downsample Example

Recall, our original signal at 5Hz...

1. Original 5 Hz signal

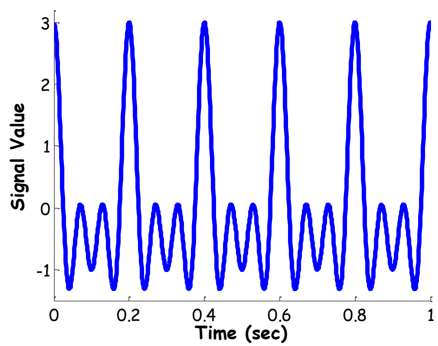
BW = 5 Hz

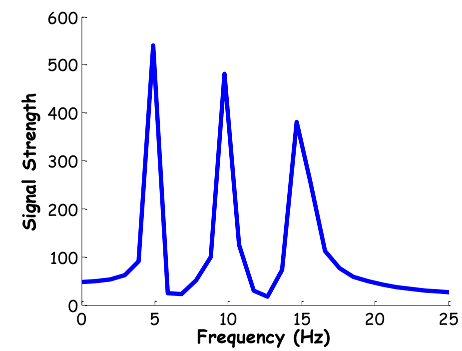




2. We added 10 & 15 Hz components!

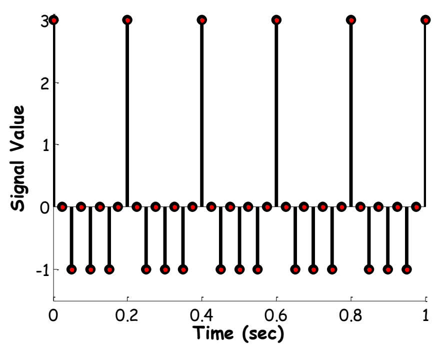
BW = 15 Hz

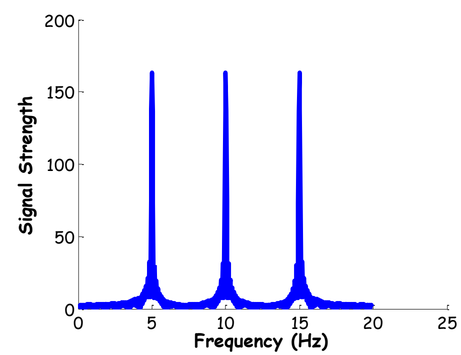




3. Then we sampled at S_{R1} = 40Hz

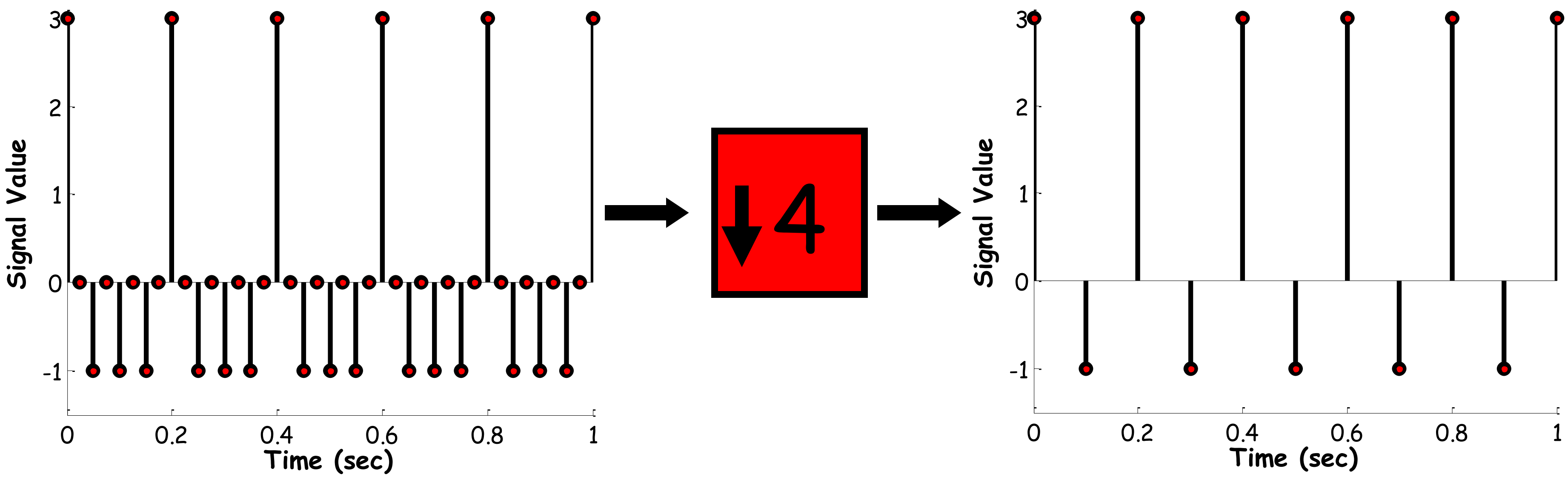
BW = 15 Hz





Resample the sampled signal: downsampling

Downsample by 4 means to retain only every 4th sample



Sample Rate 1 S_{R1} = 40Hz

N_{F1} = 20Hz > 15Hz = BW

GOOD!

Sample Rate 2 S_{R2} = 10 Hz

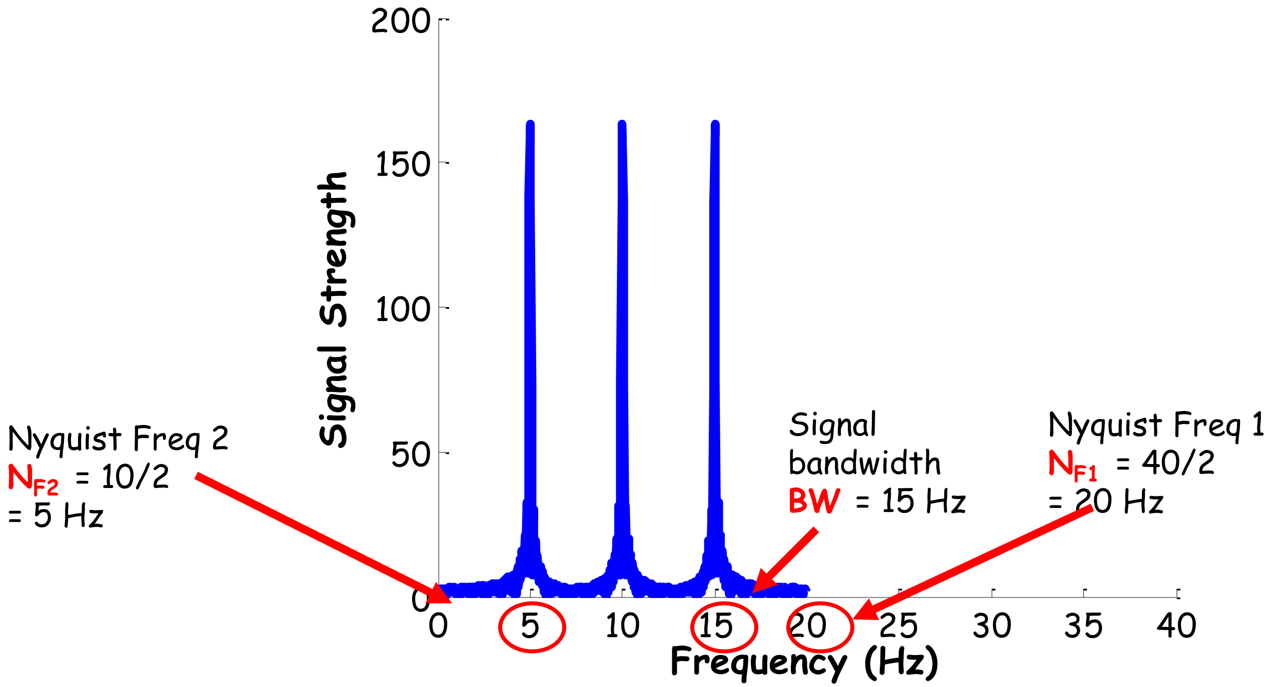
N_{F2} = 5Hz < 15Hz = BW

BAD!

Nyquist Freq < Bandwidth ☹

Cannot recover original signal bandwidth, since new Nyquist Frequency (5Hz) is less than the desired signal bandwdidth BW (15Hz).

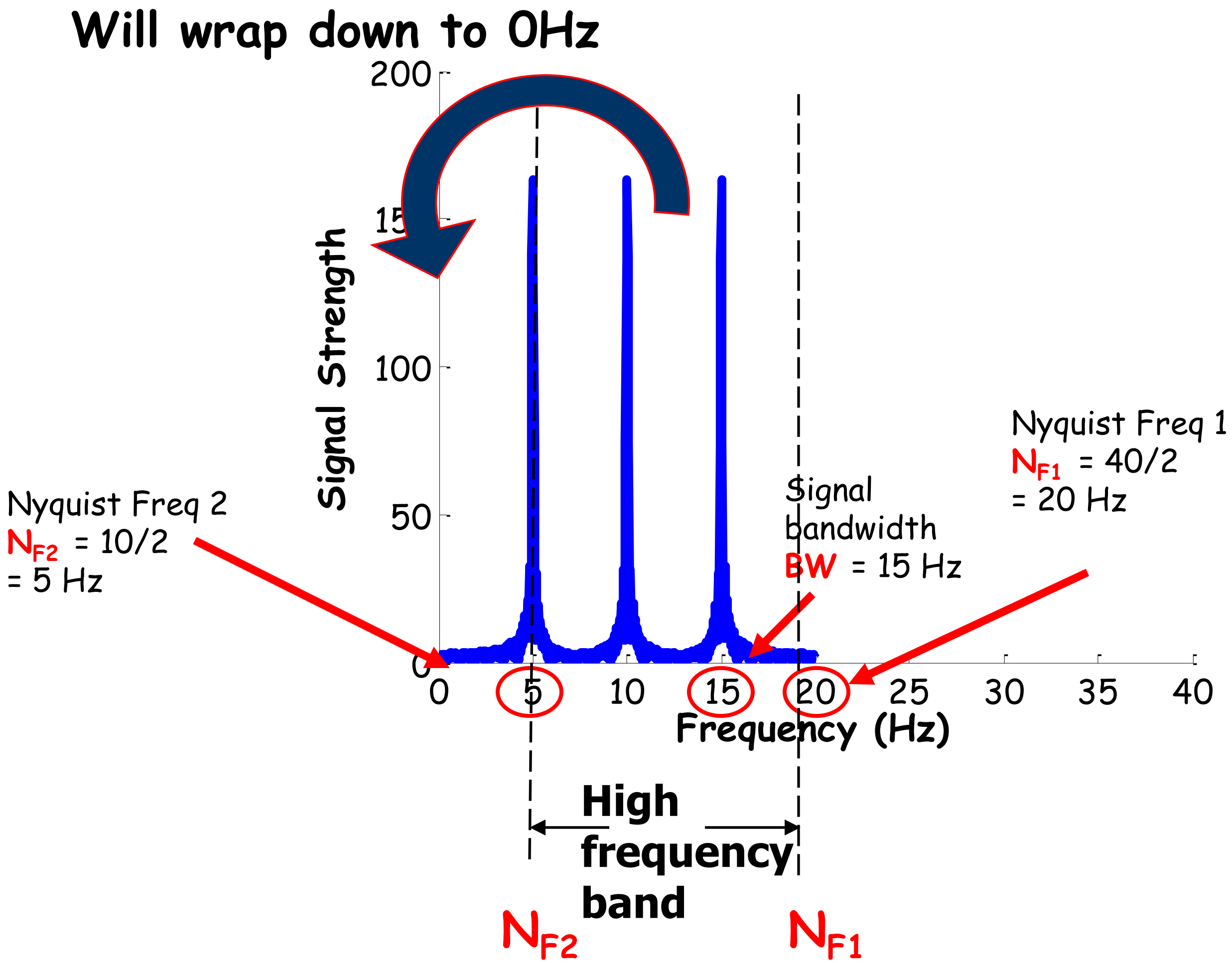
N_{F2} < BW means we cannot recover 15Hz BW signal



Is the original 5Hz signal recoverable? It should be, since $N_{F2} \geq$ BW 5 Hz

Why 5Hz signal not recoverable:

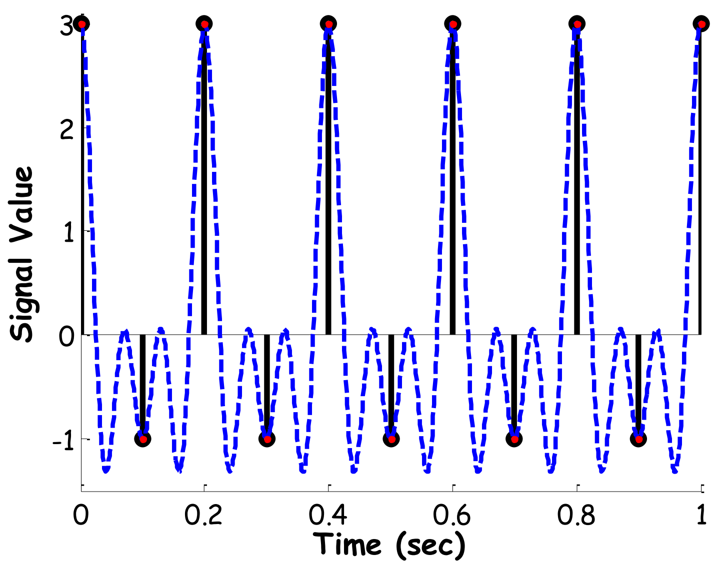
High Frequency band causes aliasing when downsampled



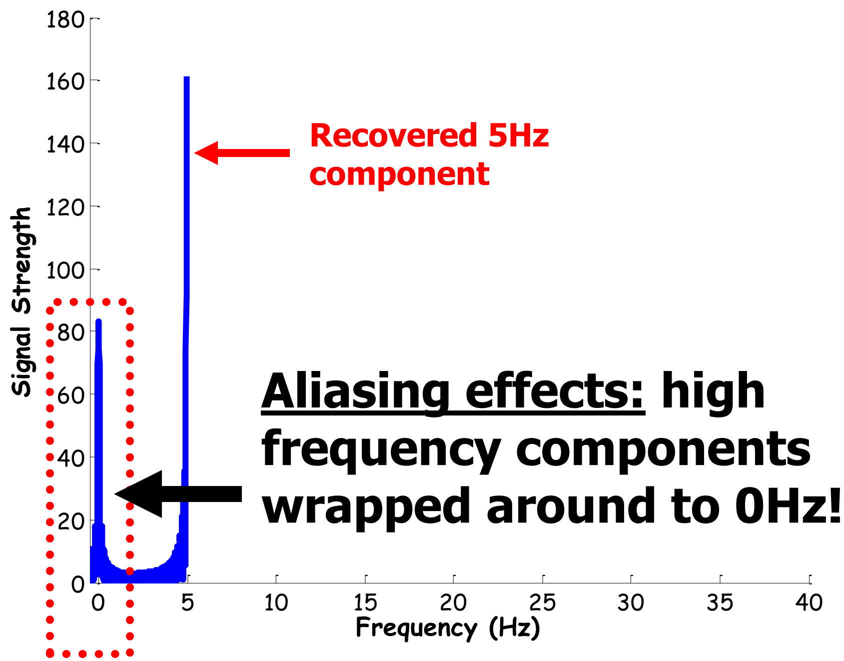
High frequency band will wrap down to 0Hz when downsampled

Why 5Hz signal not recoverable:
Aliasing Effects

Due to the **high frequency** components at 10Hz and 15Hz that show up at 0Hz when the signal is **downsampled**, the 5Hz component is **not recoverable**.



S_{R2} = 10 Hz



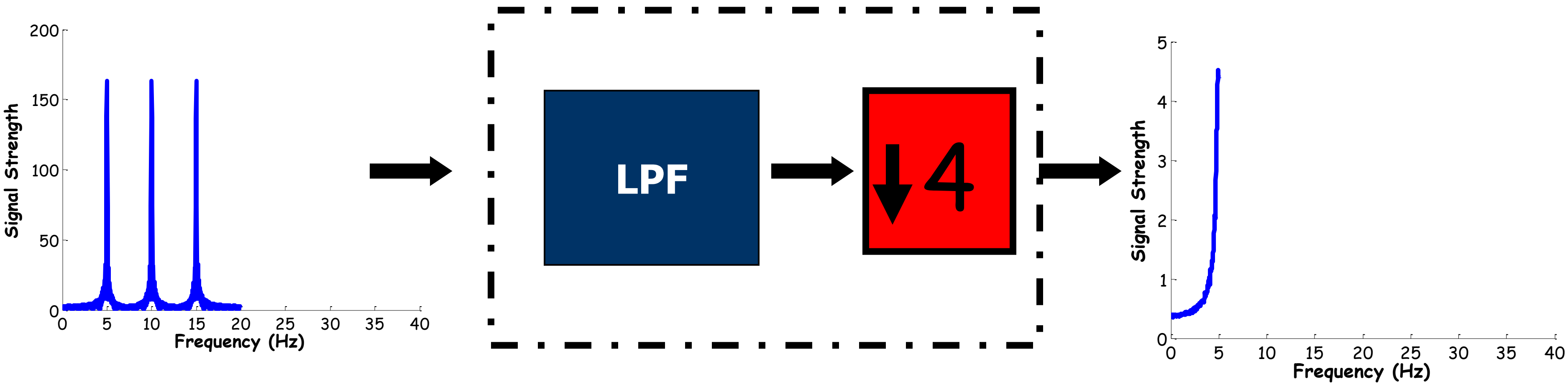
... *unless we **remove the high frequency** components **before downsampling**.*

How to Remove the High Frequency components before downsampling using a low-pass filter

- A **low-pass** filter (LPF) removes high frequency components by only letting **low** frequency components **pass** through.

- It **removes the jagged edges** that were due to high frequencies.

Proof in the pudding: No more aliasing effects when using low pass filter!

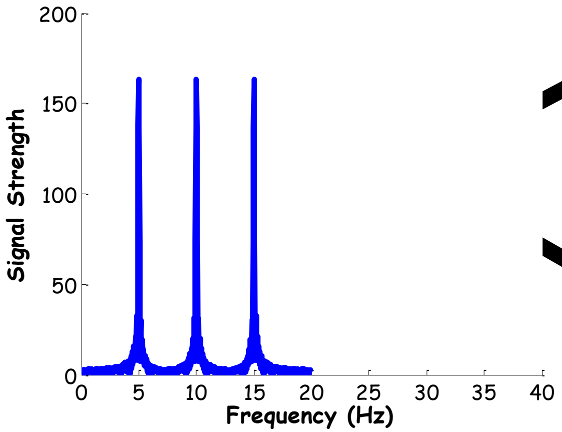


S_{R1} = 40 Hz

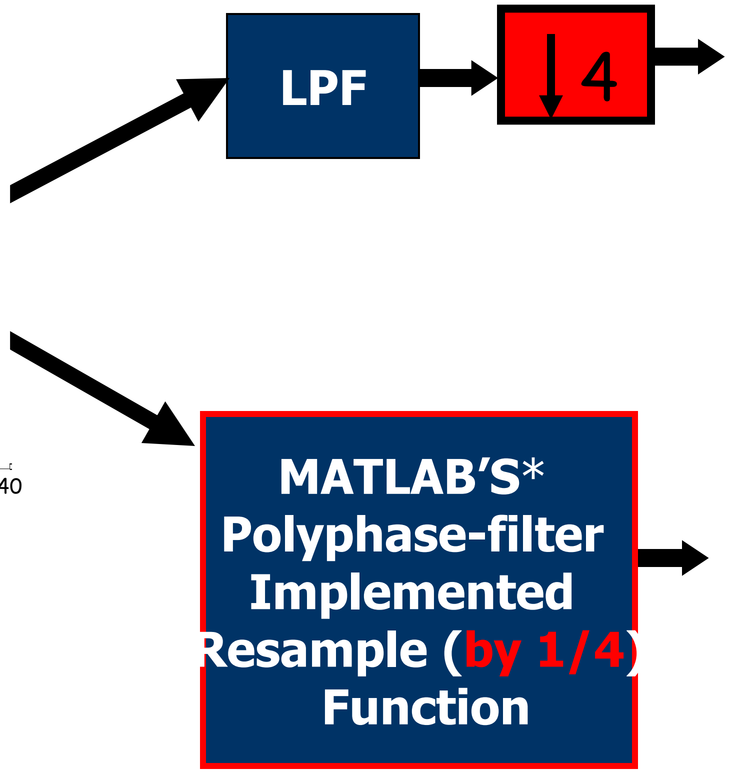
S_{R2} = 10 Hz

The original 5Hz signal is successfully recovered!

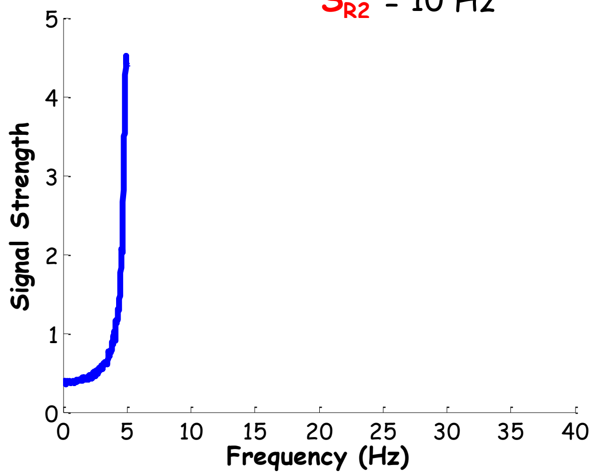
Proof in the pudding: LPF+downsampling <==> multirate polyphase filter resampling



Sample Rate 1
S_{R1} = 40 Hz



Sample Rate 2
S_{R2} = 10 Hz



Sample Rate 2
S_{R2} = 10 Hz

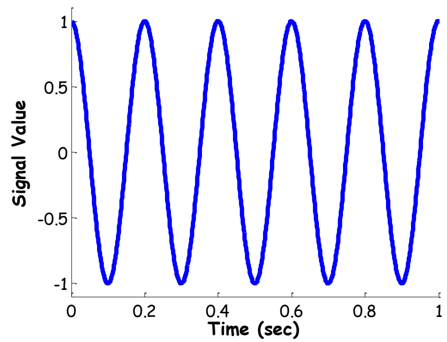
* MATLAB is an industry standard software which performed all computations and corresponding figures in this presentation

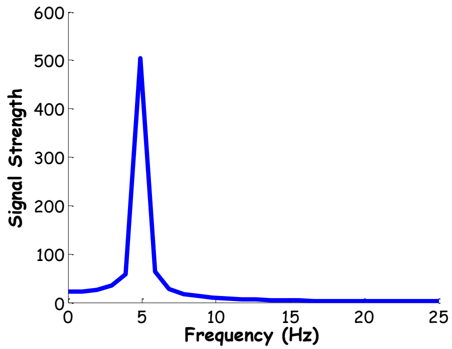
III. Upsampling example

Assume our original signal at 5Hz…

1. Original 5 Hz signal

BW = 5 Hz

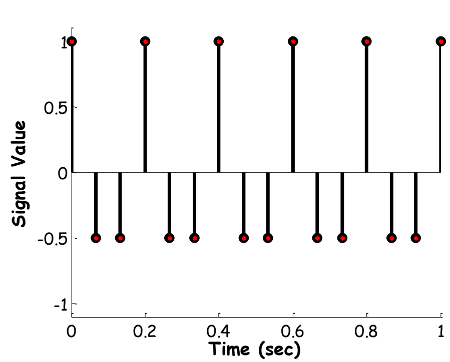


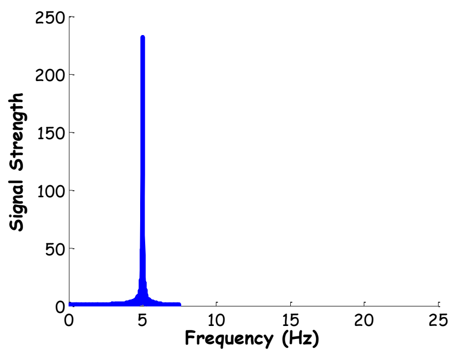


Nyquist Rate N_R = 2 x BW 5Hz = 10Hz, so sample at sampling rate S_R = 15Hz

2. We sample at S_{R1} = 15Hz

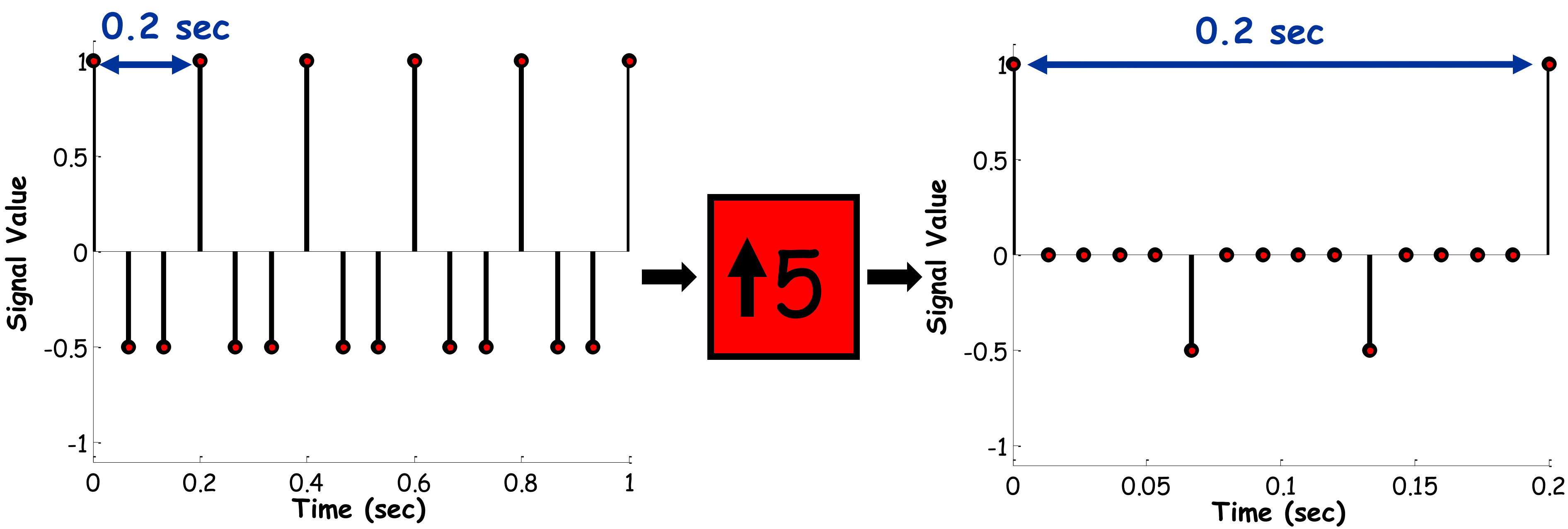
BW = 5Hz





Resample the sampled signal: upsampling

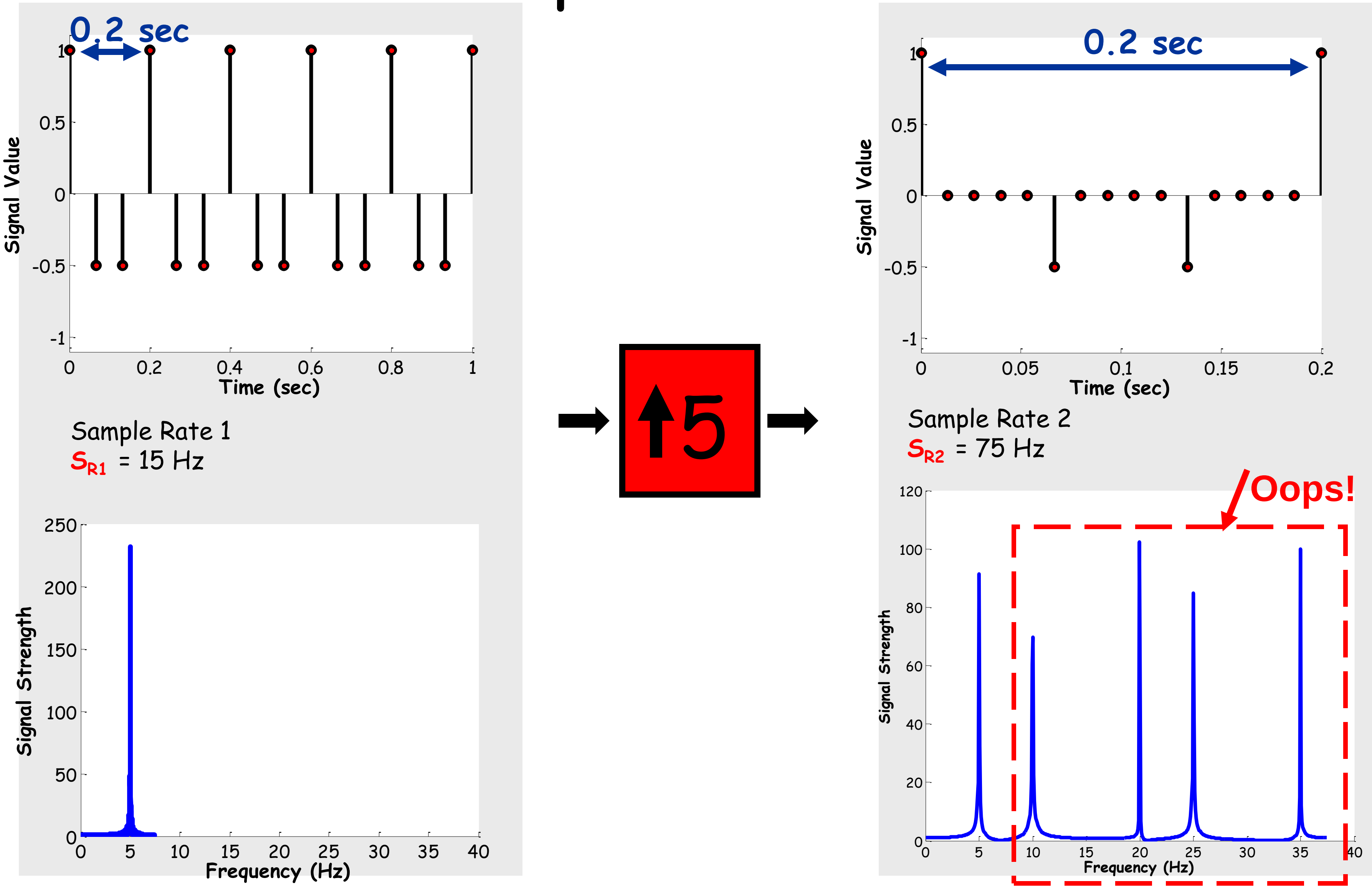
If you need to increase the number of samples in a given time by a factor of 5, you **upsample** by 5 (**insert** 5-1=4 **zeros between each sample**).



Sample Rate 1
S_{R1} = 15 Hz

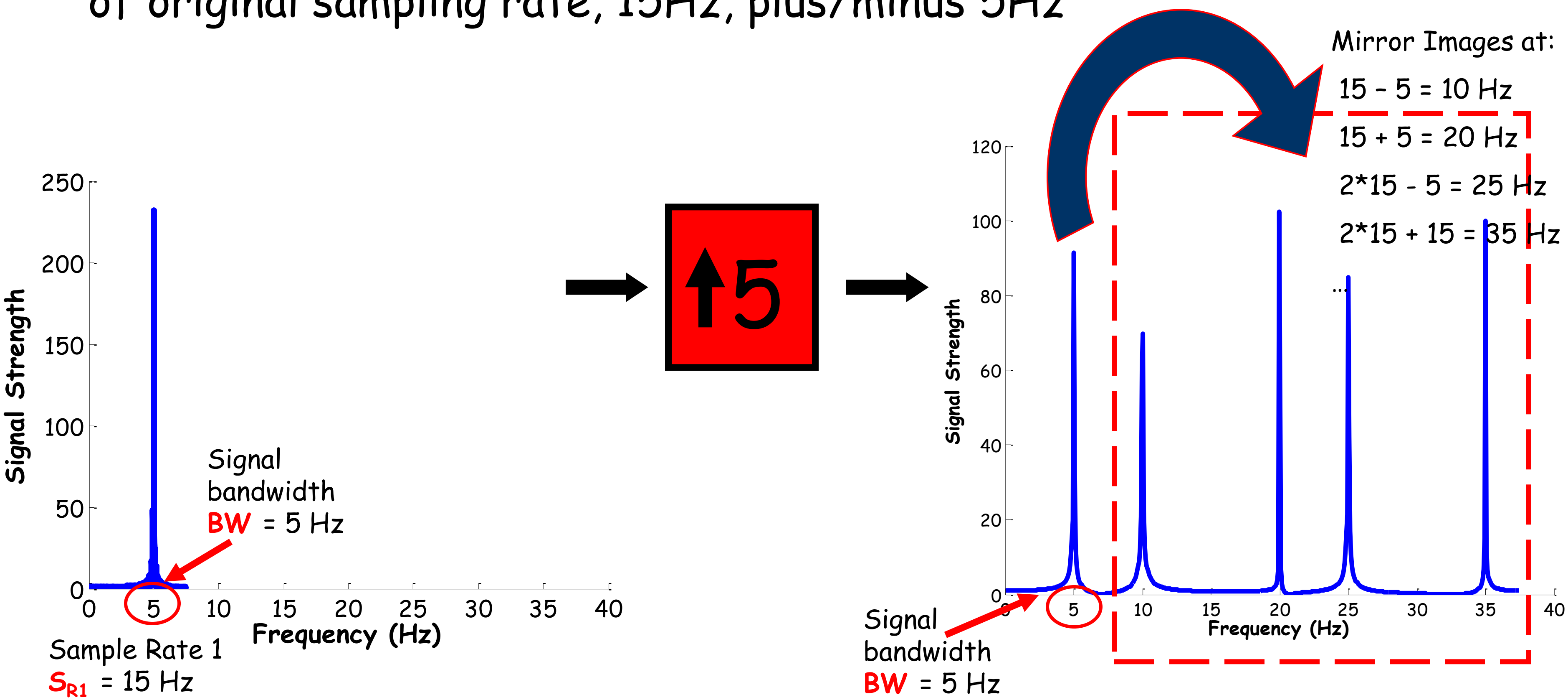
Sample Rate 2
S_{R2} = 75 Hz

Upsampled signal in frequency representation



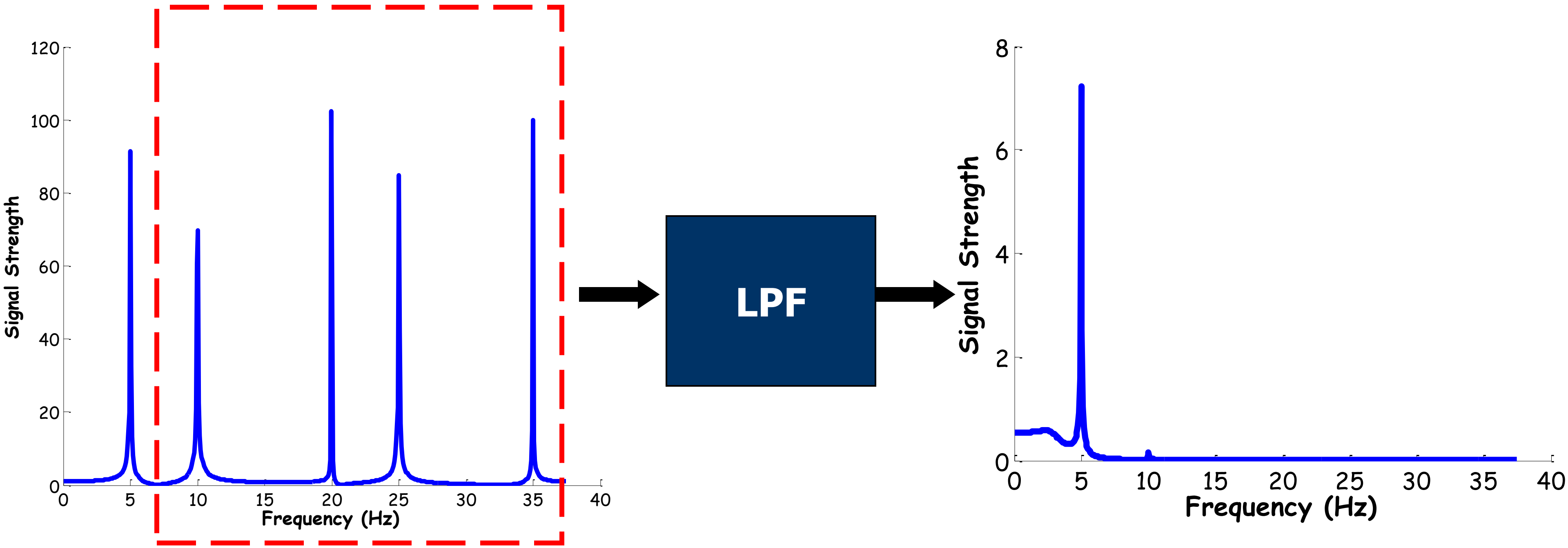
Upsampling causes aliasing in higher frequencies

Upsampling causes copies of the original 5Hz component at multiples of original sampling rate, 15Hz, plus/minus 5Hz



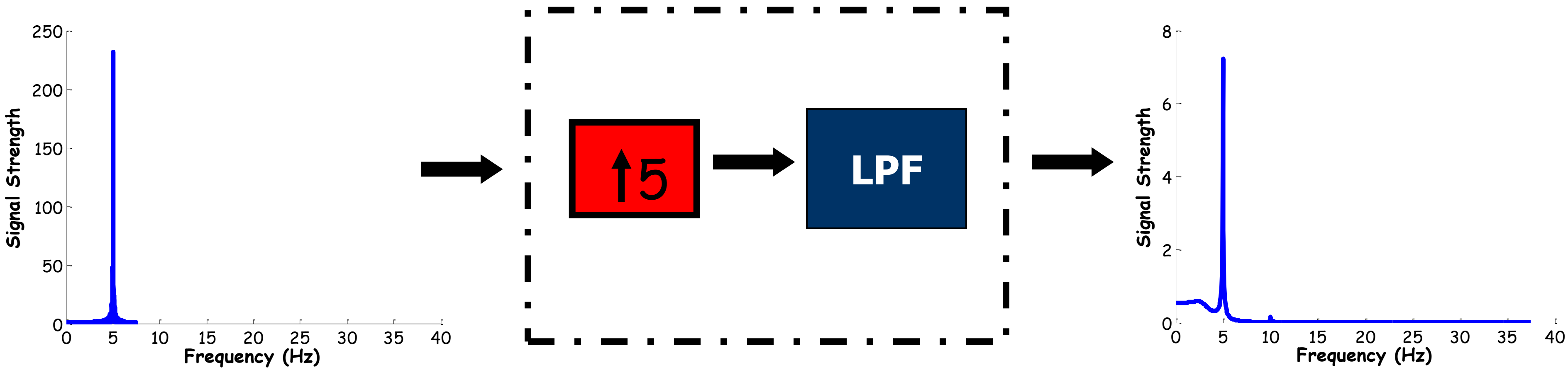
How do we remove these extra high frequency components?

How to remove the extra high frequency components caused by upsampling using a low-pass filter



Low pass filter removes these extra high frequency components

Proof in the pudding: No more aliasing effects when using low pass filter!

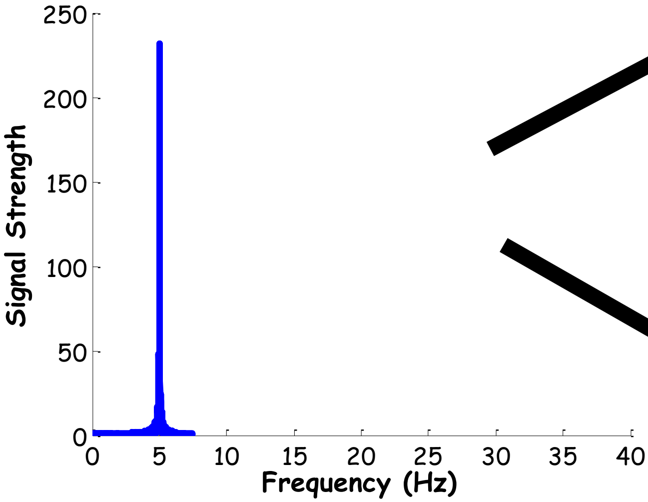


S_{R1} = 40 Hz

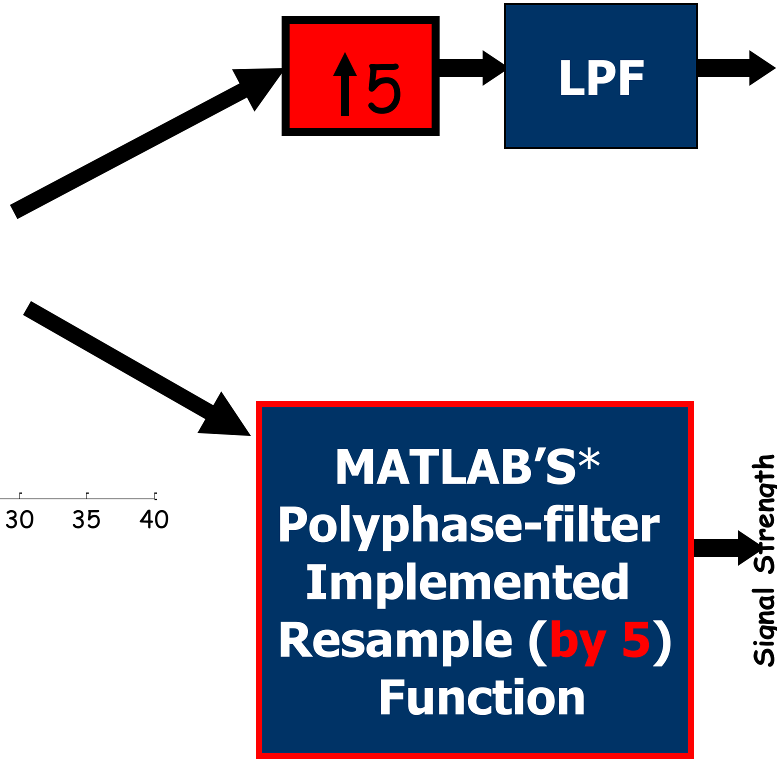
S_{R2} = 10 Hz

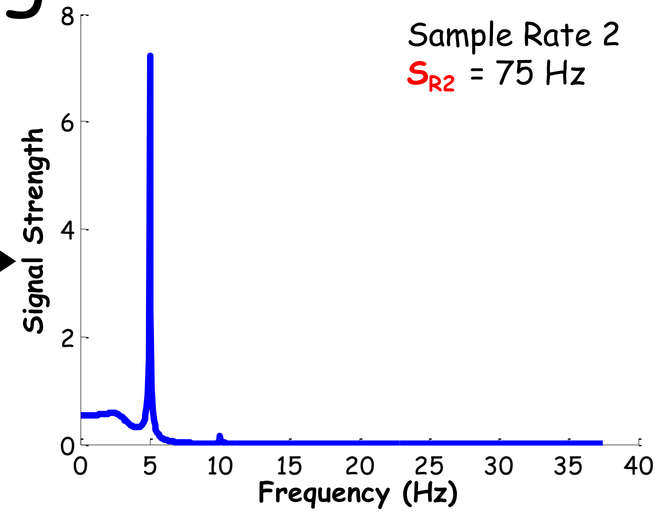
All high frequency copies of the 5Hz signal are removed!

Proof in the pudding: upsampling and lowpass filter <==> multirate polyphase filter resampling

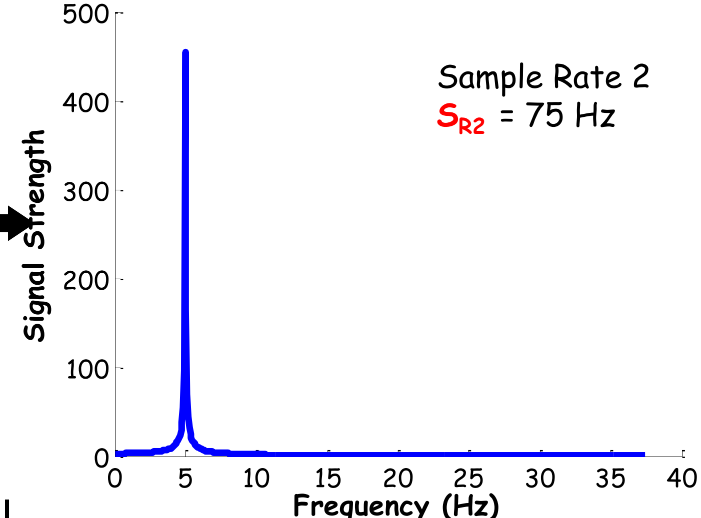


Sample Rate 1
S_{R1} = 15 Hz





Sample Rate 2
S_{R2} = 75 Hz



Sample Rate 2
S_{R2} = 75 Hz

* MATLAB is an industry standard software which performed all computations and corresponding figures in this presentation