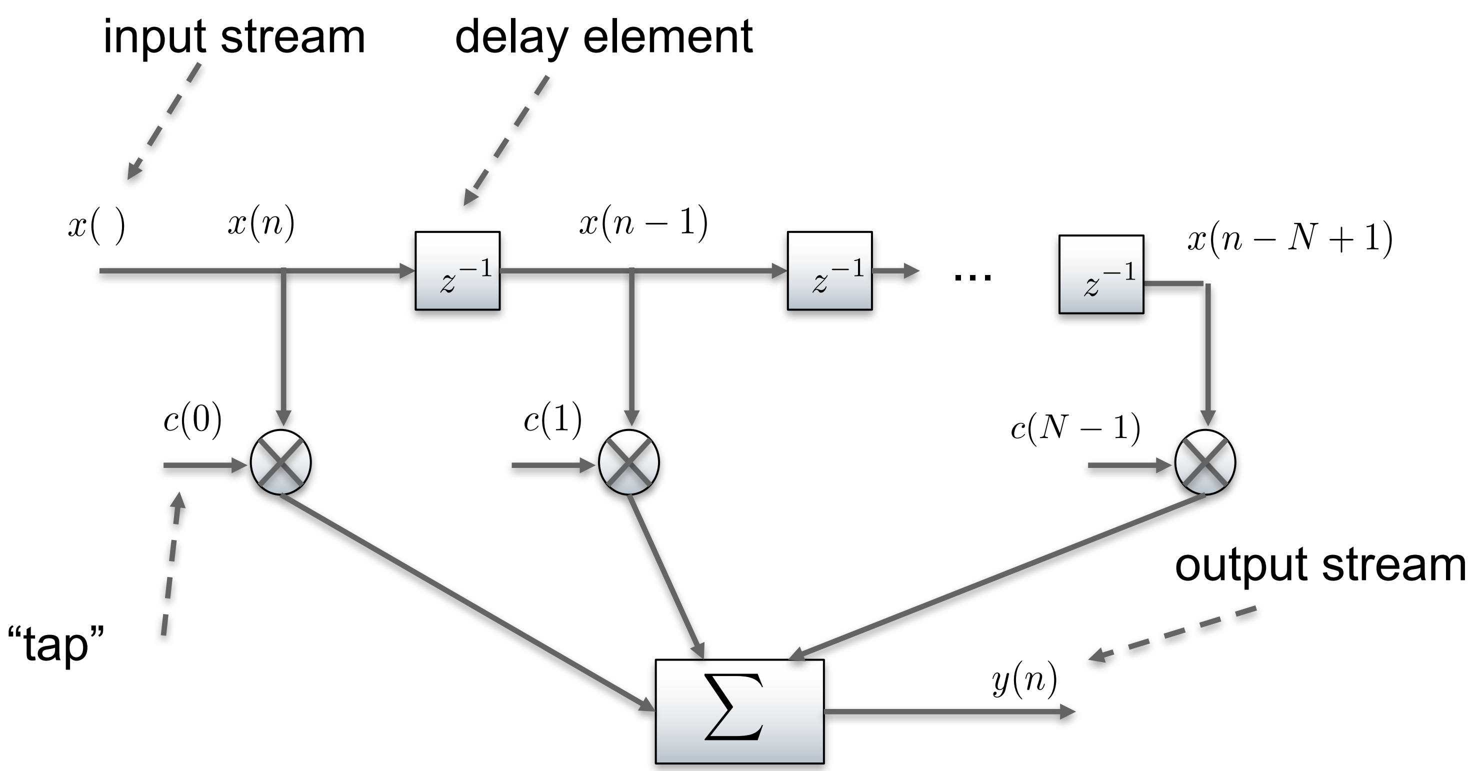
Lab 4
The Intuition Behind FIR Filters

FIR filter computational structure

- **The FIR Filter Structure is the following**
  - This computes the convolution of *x(n)* with *c(n)*

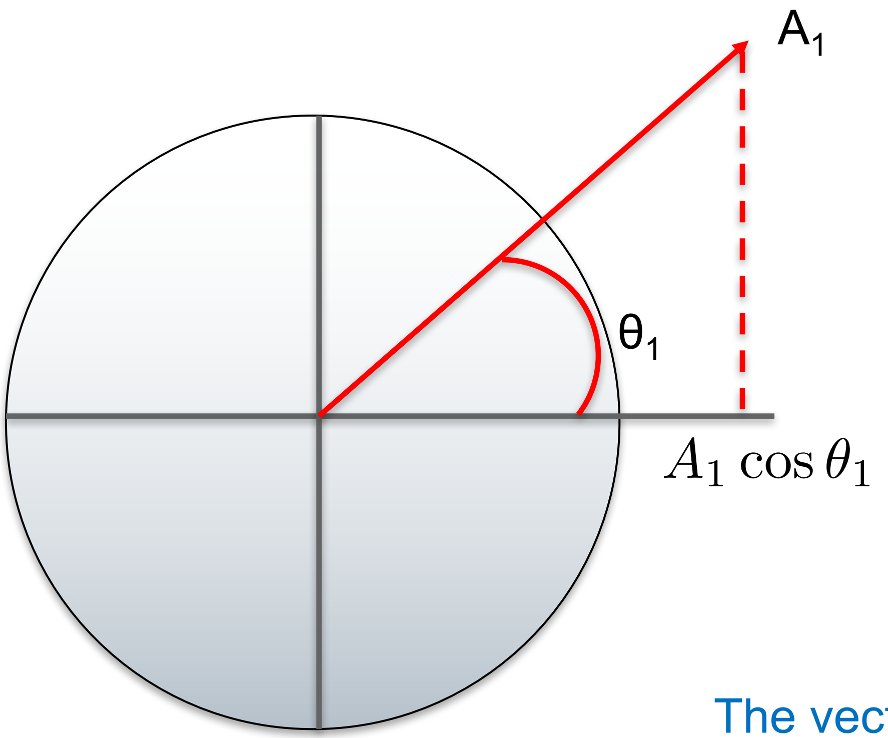


The roadmap to intuitive insight

- **We will ask and answer:**
  - What is the sum of multiple sinusoids, all at the same frequency but with different amplitudes and phases?
    - ✓ Answer: a sinusoid of the same frequency with an amplitude and phase we can compute
  - What is the output of an FIR filter when a sinusoid is input?
    - ✓ Answer: the sum of N sinusoids, all at the same frequency but with different amplitudes and phases—hence: a sinusoid of the same frequency as the input!
  - How can we use the above to design a filter with an arbitrary frequency response?

The projection of a rotating vector is a cosine

- **The projection of any vector rotating around the origin is a sinusoid at the rotation frequency**



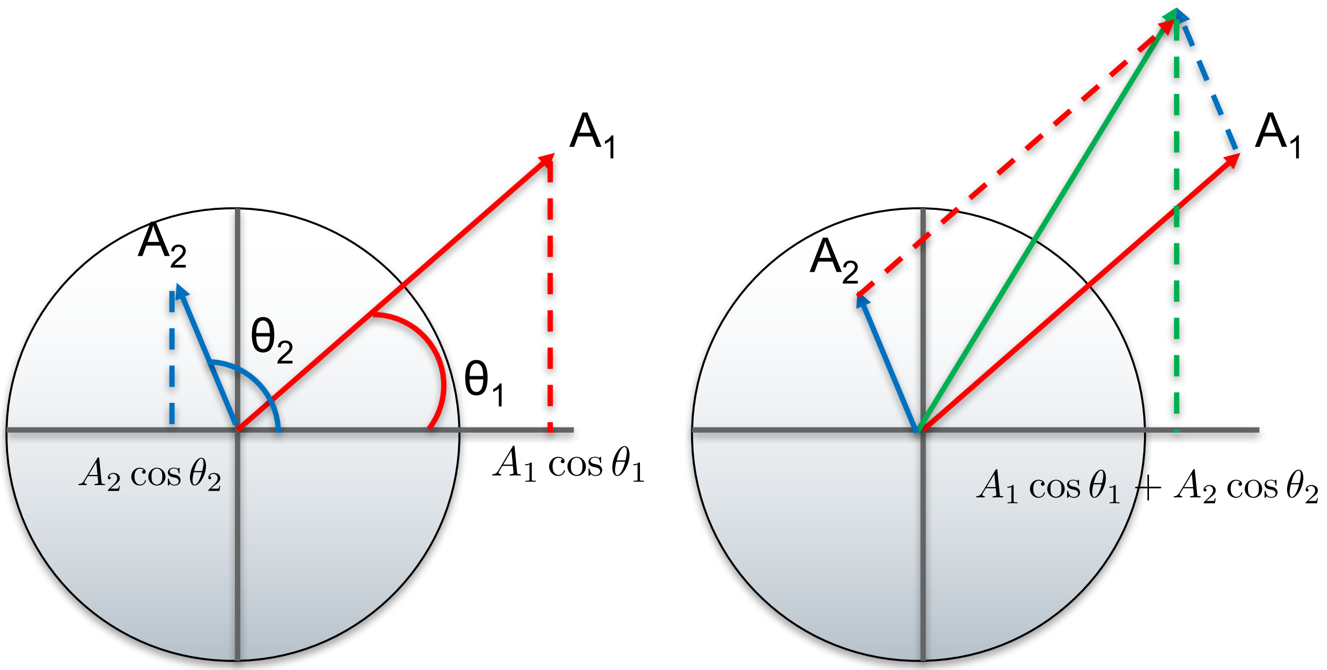
As the red vector rotates counterclockwise at the frequency *f*, its projection onto the x-axis at time *t* is:

$$A_1 \cos(2\pi f t + \theta_1)$$

The vector is drawn for *t = 0*

Summing sinusoids at the same frequency

- **Any sum of two sinusoids, both at the <u>same</u> frequency, yields a sinusoid at that frequency**
  - As the red and blue vectors rotate, their vector sum (green arrow) rotates at the same rate. It's projection is therefore a sinusoid at the same frequency



Extension to arbitrary sum of sinusoids

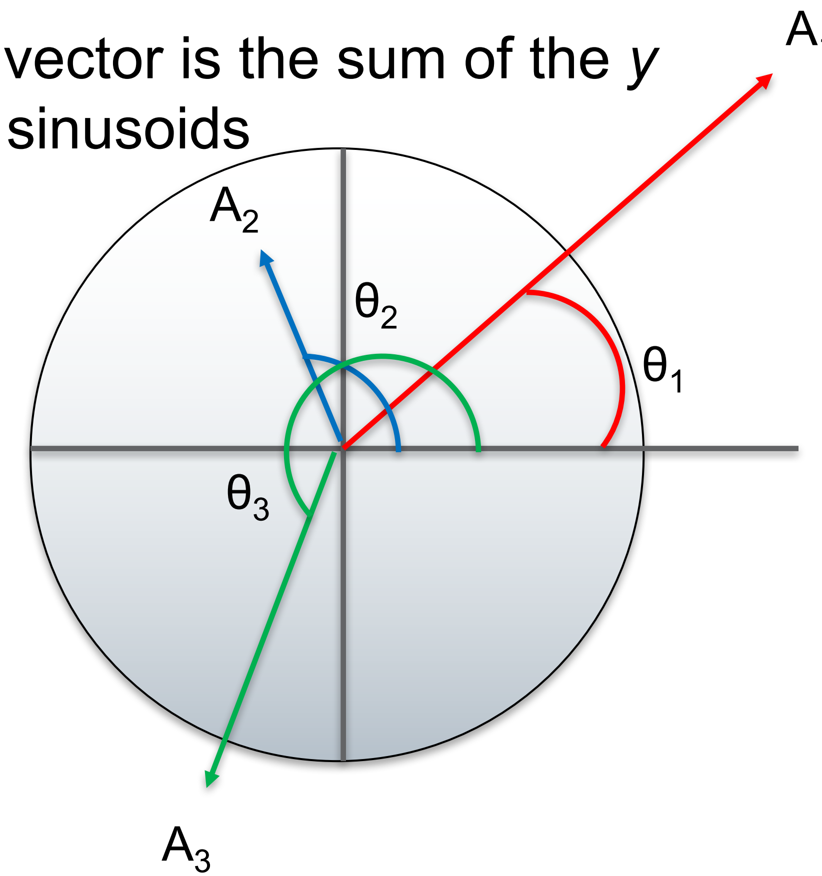
- **Any sum of sinusoids all at the same frequency is a sinusoid at that frequency**
  - Proof: suppose you are adding up 4 sinusoids
    - ✓ Add the first 2 sinusoids; then add that sum to the third sinusoid; then add that sum to the last sinusoid
      - – At each step you are adding two sinusoids at the same frequency, so you get another sinusoid at that same frequency

Formula for adding multiple sinusoids at the same frequency

- **Determine the final vector with vector addition**
  - The *x* component of the final vector is the sum of the *x* components of the individual sinusoids
  - The *y* component of the final vector is the sum of the *y* components of the individual sinusoids

$$S_x = \sum_{n=1}^{N} A_n \cos \theta_n$$

$$S_y = \sum_{n=1}^{N} A_n \sin \theta_n$$

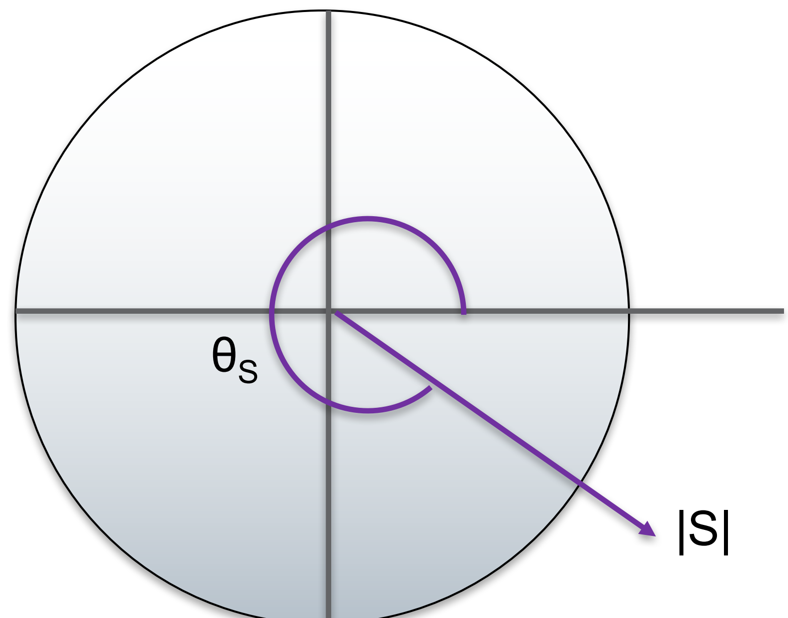


Magnitude and angle of final vector

- **The magnitude and angle of the final rotating vector**

$$|\underline{\mathbf{S}}| = \sqrt{(S_x, S_y) \cdot (S_x, S_y)}$$

$$\theta_S = \tan^{-1} \frac{S_y}{S_x}$$

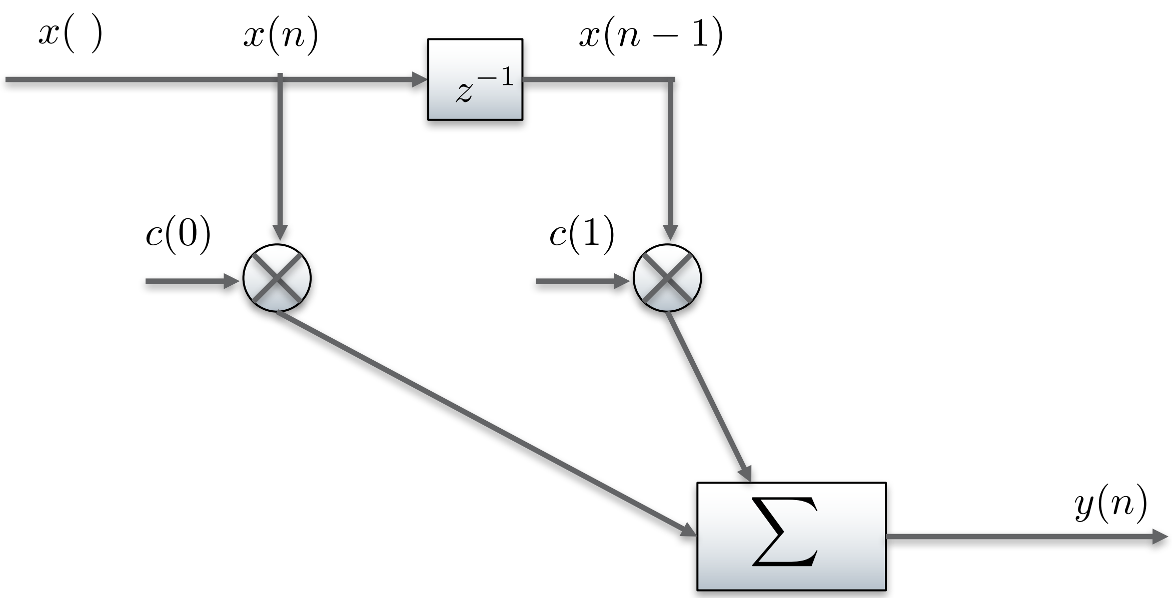


The projection of ***S*** onto the x-axis as a function of *t*

$$s(t) = |\underline{\mathbf{S}}| \cos(2\pi f t + \theta_S)$$

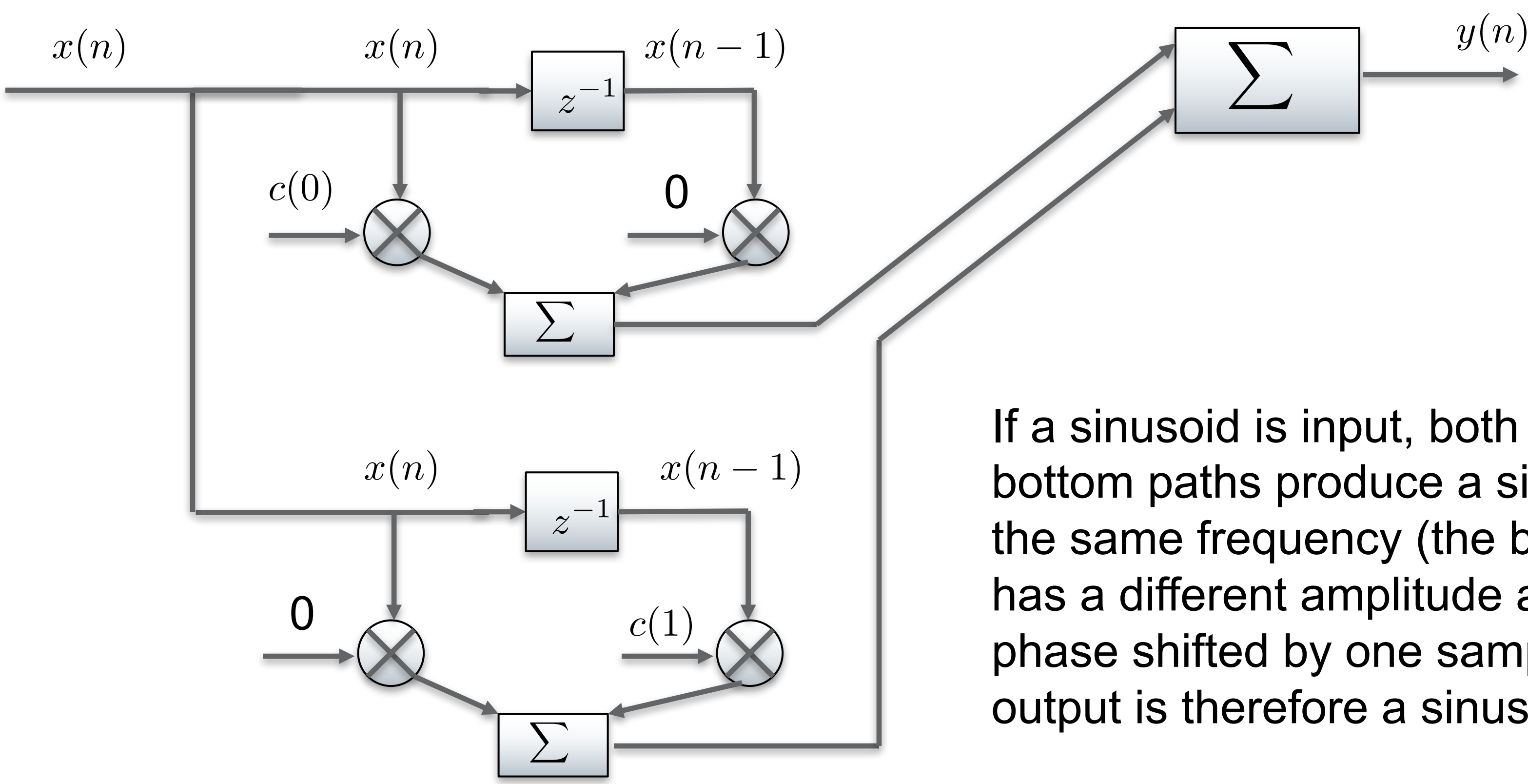
Decomposing an FIR filter

- **Consider a two-tap FIR**



Decomposing an FIR filter cont.

- **We can decompose it into two FIRs as follows**



If a sinusoid is input, both the top and bottom paths produce a sinusoid at the same frequency (the bottom path has a different amplitude and is phase shifted by one sample). The output is therefore a sinusoid!

Extension to more taps

- **The preceding argument obviously extends to an arbitrary number of taps**

A sinusoid applied to an FIR's input produces a sinusoid of the same frequency at its output. The magnitude and phase may be changed.

FIR's output for a sinusoidal input

- **Assume a sinusoid of magnitude 1 is input to an FIR filter**
  - Let F_s be the sampling frequency and *f* be the sinusoid's frequency. The sampling period is $T = 1 / F_S$
  - The input to the correlator is

$$x(n) = \cos(2\pi f n T)$$

- **The *m*'th correlator "tap" produces a scaled and delayed version of *x(n)***

$$\psi_m(n) = c_m \cos(2\pi f n T - \theta_m)$$

The delay angle

- **The delay angle of the *m*'th tap's sinusoid depends on *T*, *f*, and *m***
  - A delay of *m* samples is a delay of *mT* seconds
    - ✓ The angle corresponding to this time delay depends on *f*

$$\theta_m = 2\pi f T m$$

We can compute the filter's response to any cosine input!

- **The following steps can be used to compute the response of a correlator to a magnitude 1 cosine of frequency *f***
  - Compute the delay angle for each tap *m*

$$\theta_m = 2\pi f T m$$

  - Then compute the "final" vector

$$\underline{\mathbf{S}} = \left(\sum_{m=0}^{N-1} c_m \cos\theta_m , - \sum_{m=0}^{N-1} c_m \sin\theta_m \right)$$

Note: a minus sign is needed because the sinusoid is delayed, not advanced. (An advance was used in the derivation)

FIR frequency response

- **We can plot the magnitude of the correlator's output as a function of *f***
  - This is the magnitude of the vector $\underline{S}$
    - ✓ We call this plot the "magnitude frequency response"
- **We can plot the angle of the correlator's output as a function of *f***
  - This is the angle of the vector $\underline{S}$
    - ✓ We call this plot the "phase response"

Example: The Moving Average (MA)

- **One of the most common operations for processing a series of data is the "moving average"**
  - It smooths out fluctuations
- **Assume you have a series of data of some sort**
  - The price of milk each day, the rainfall each month, the number of widgets produced each hour, etc.
- **Compute an average inside a window of some width**
- **Slide the window over 1 data point and repeat**

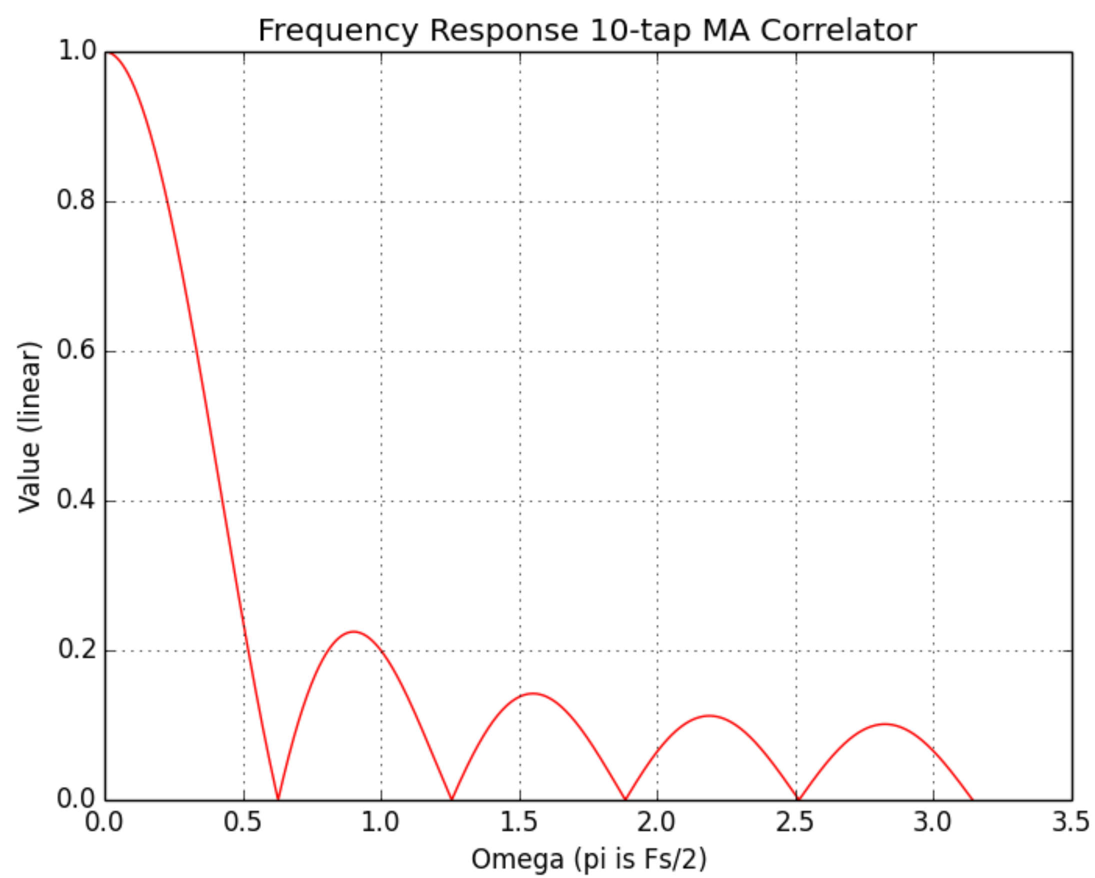
This is an FIR Filter!

The MA FIR filter

- **Assume the average is computed over a window of length *N***
- **Let each tap have a value of *1/N***
- **We can compute the response of the moving average filter with *N* taps as a function of frequency**

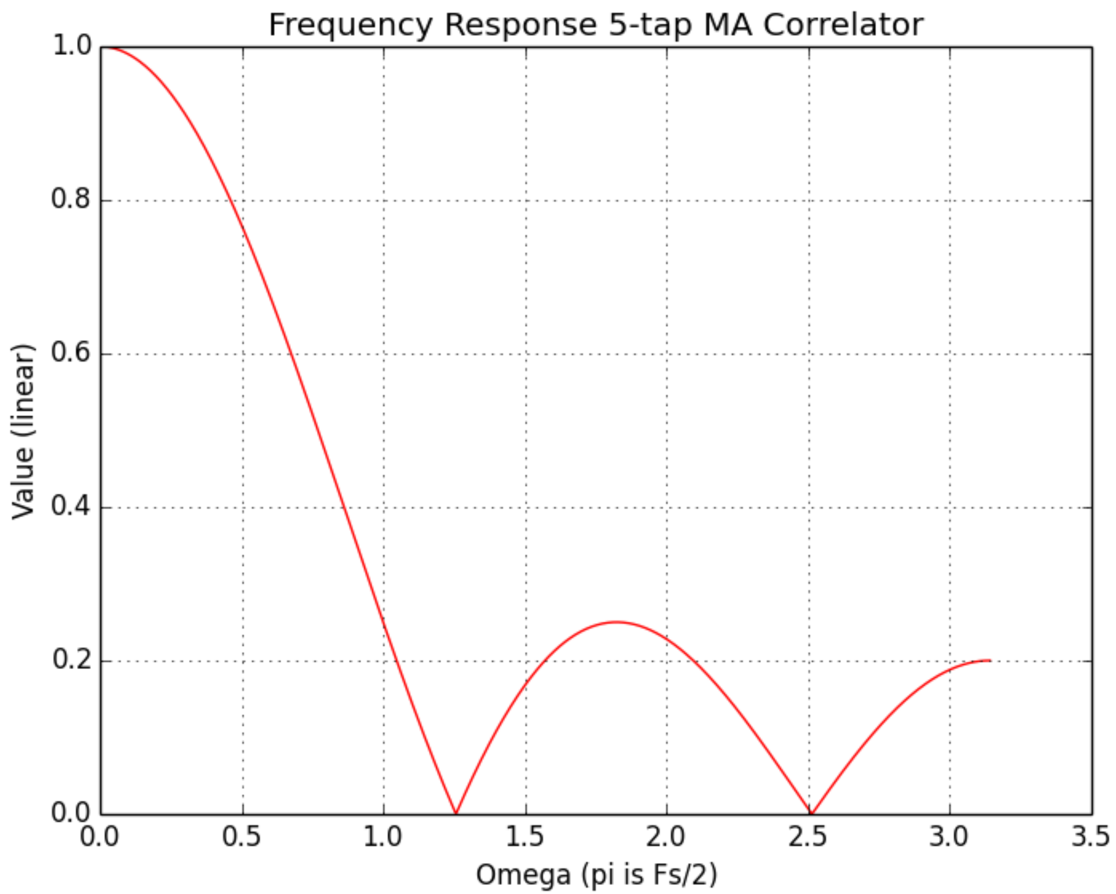
The MA FIR filter cont.

- **For 10 taps we have**



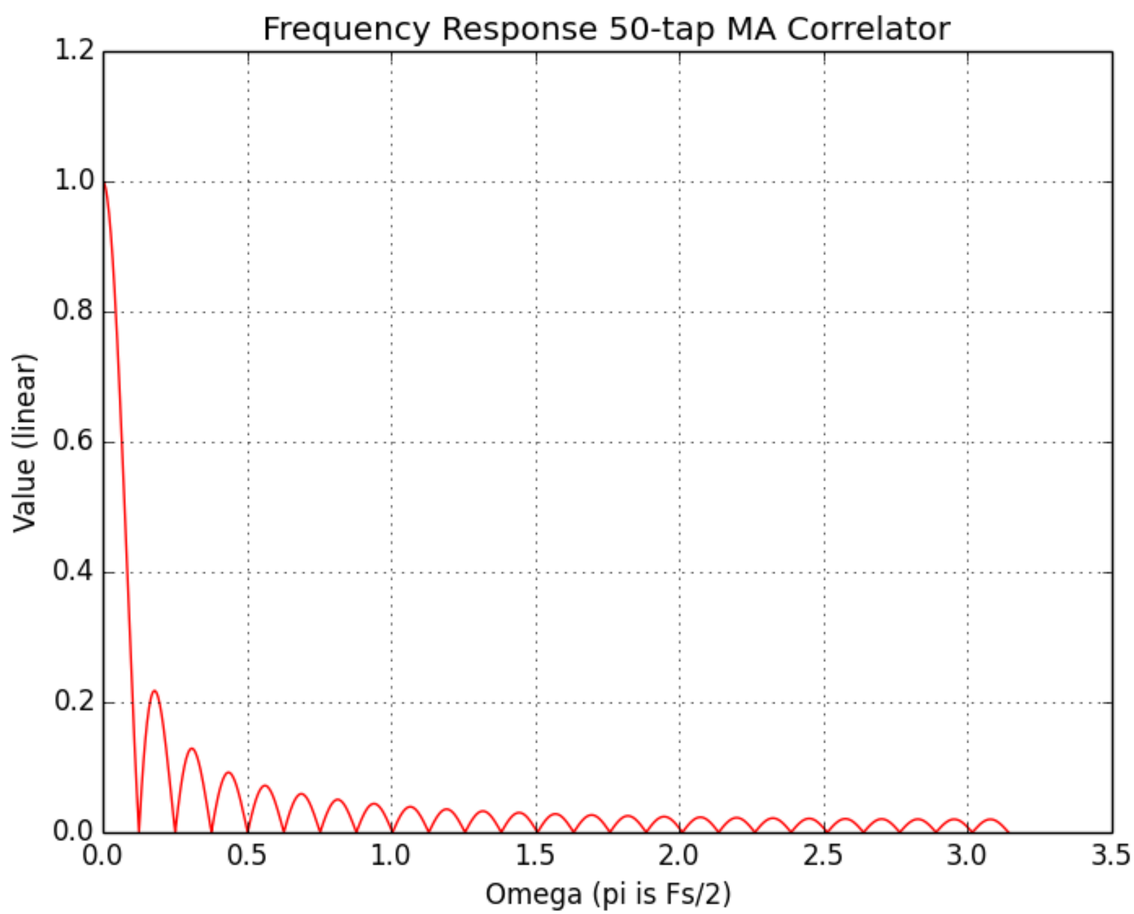
The MA FIR filter cont.

- **For 5 taps we have**



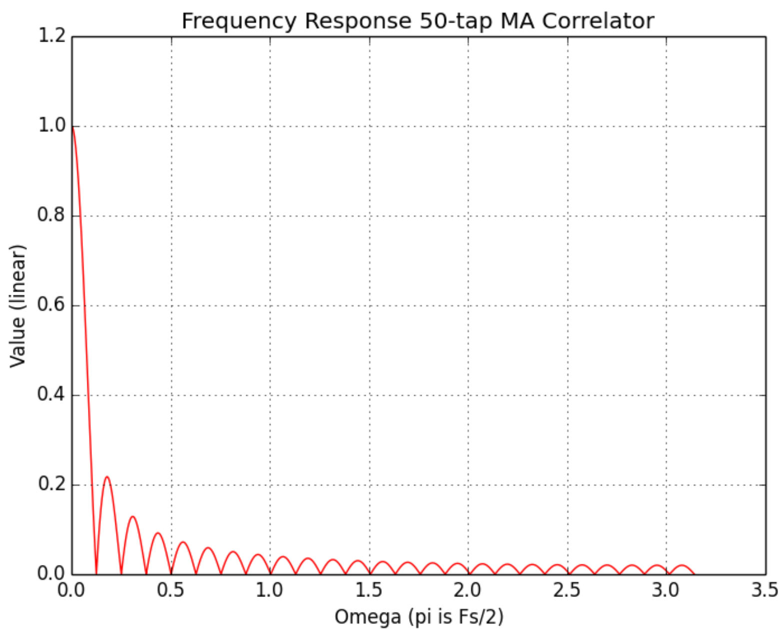
The MA FIR filter cont.

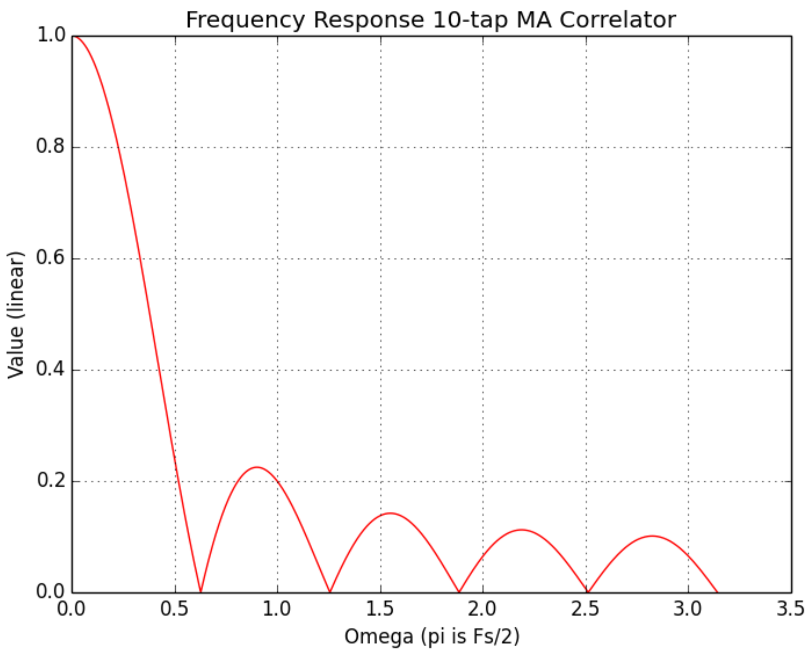
- **For 50 taps we have**



The MA is a Lowpass Filter (LPF)

- **It preferentially allows lower frequencies to pass**
  - The frequency response depends on the number of taps





What have we learned about taps?

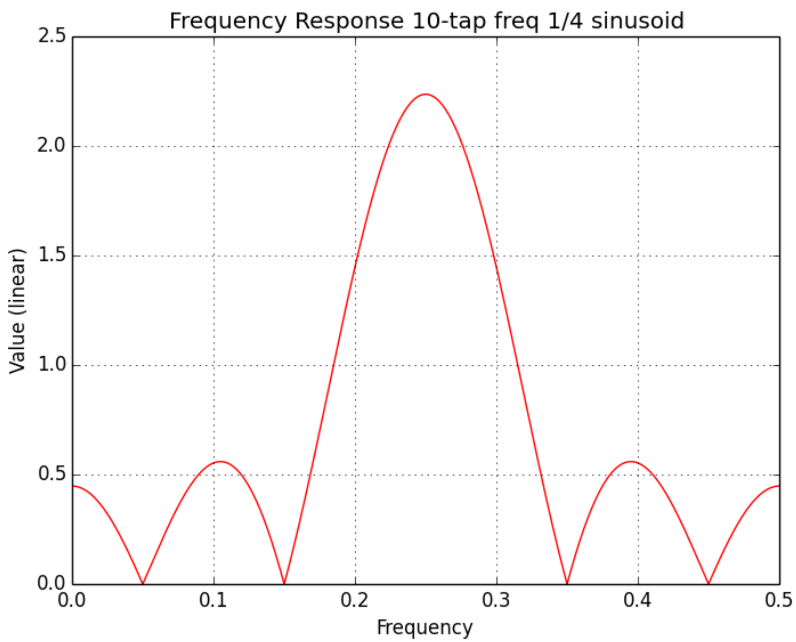
- **Different taps give different frequency responses**

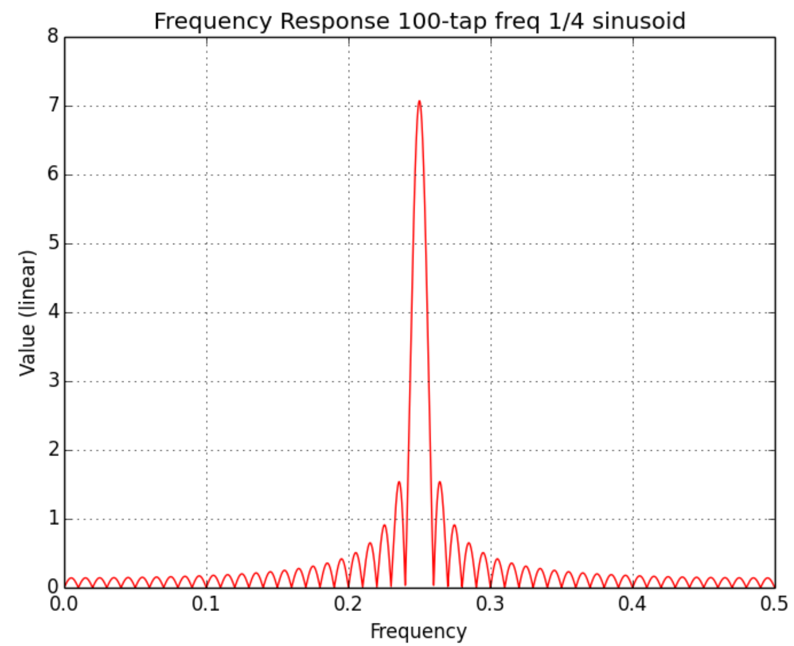
Natural questions:

Can we choose taps to get a desired frequency response?

If so, how?

Frequency response of sinusoidal taps

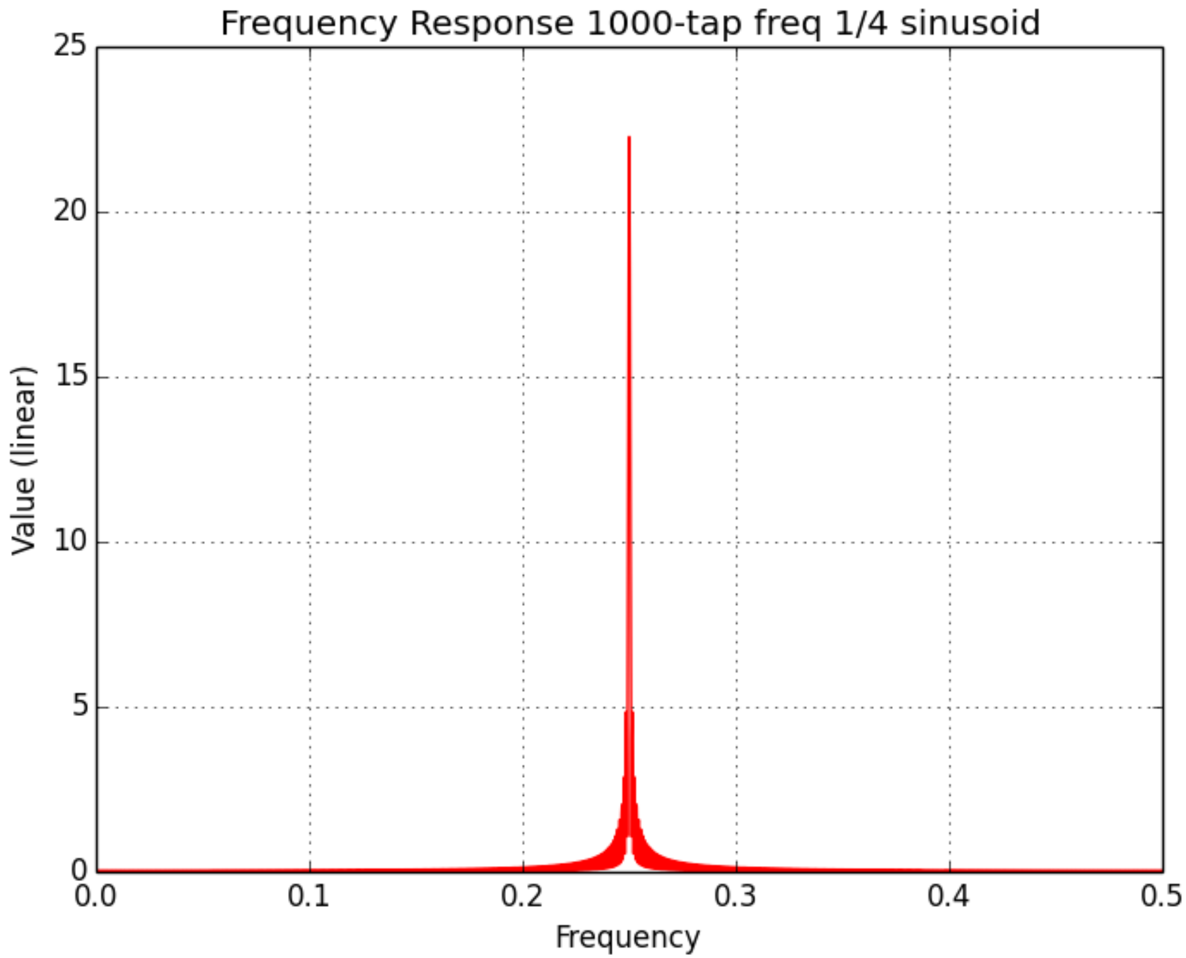




Sinusoidal taps are a bandpass filter

CU Boulder

The more taps, the narrower the filter's bandwidth

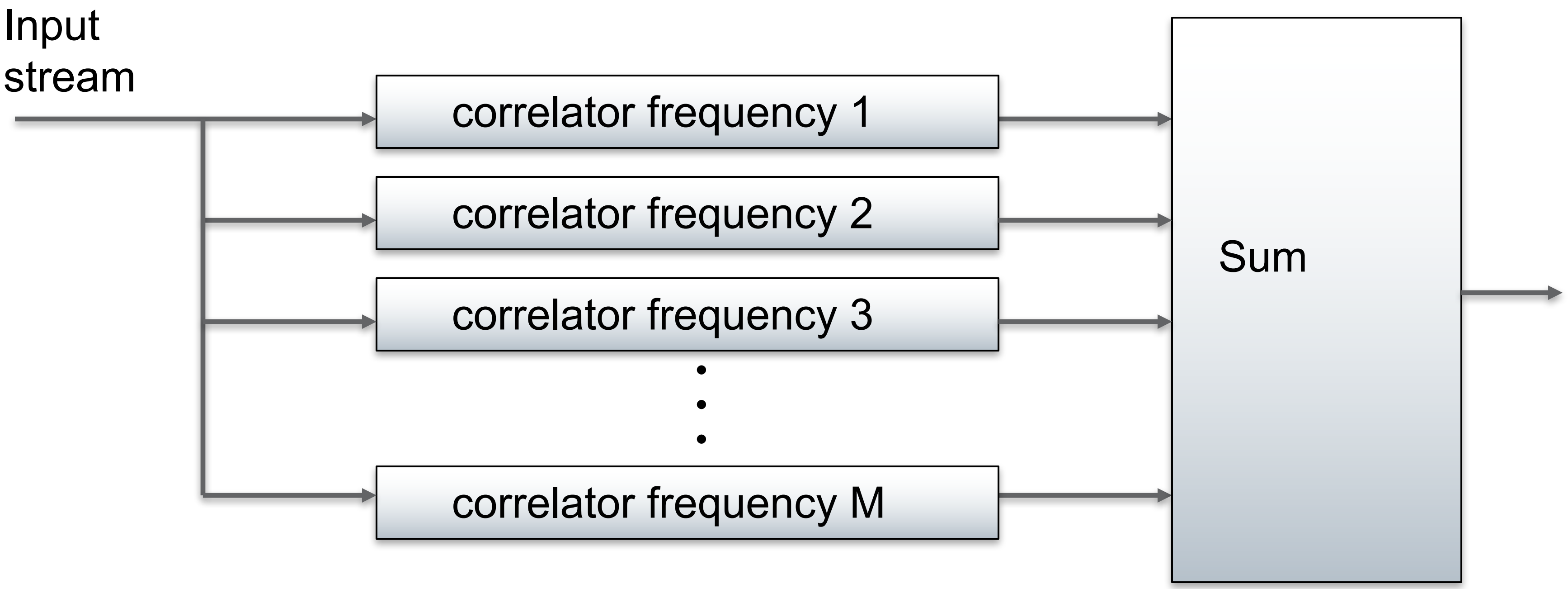


One way to think about filters

- **Intuitively, a filter should “let through what we want and stop everything else”**
  - A correlator with sinusoidal taps lets only a desired sinusoid through
    - All other sinusoids are very strongly attenuated as the tap length increases

What if we want more than one sinusoid?

- **Use multiple FIR filters in parallel**
  - Set each to let through a frequency we want
  - Sum the outputs of the correlators



A more efficient solution

- **Collapse multiple correlators into 1**
  - Add the taps for each desired frequency to get a single set of taps

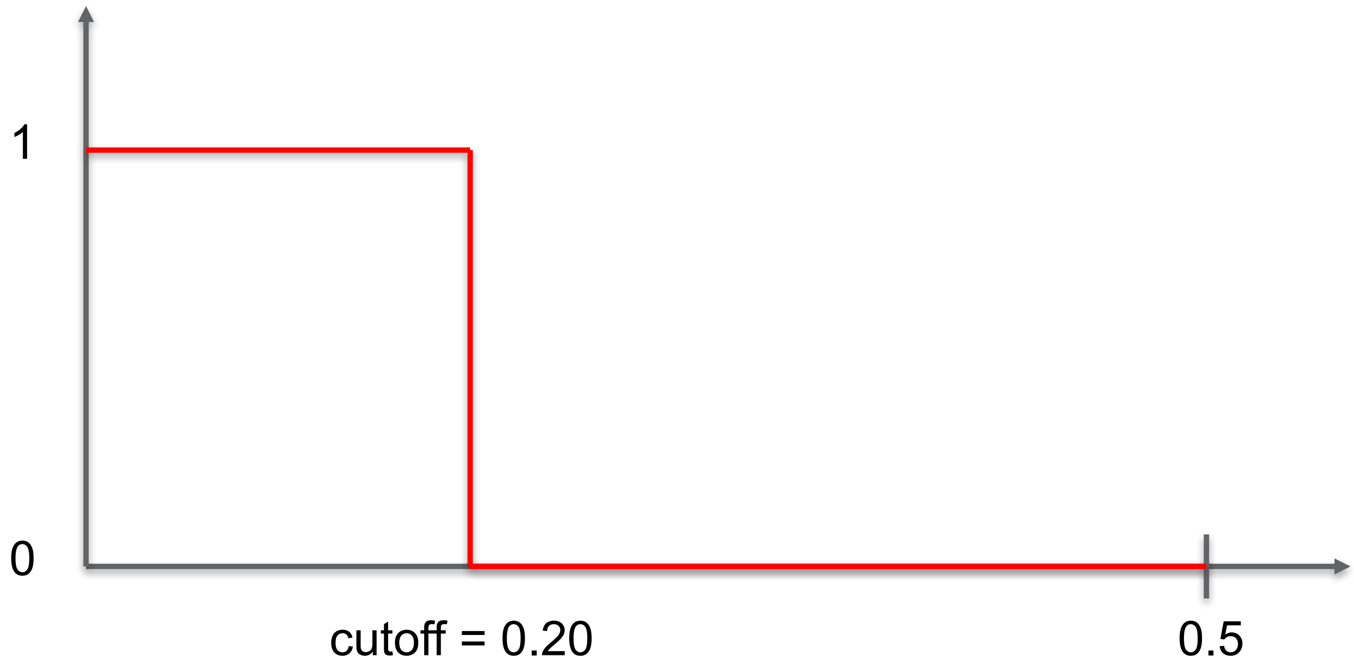
  - This is a digital filter
    - ✓ By choosing the taps the right way, we let through the frequencies we want and suppress the others
    - ✓ Another way of saying this: by choosing the right taps, we can tailor the frequency response

Building a LPF

- Choose a number of taps to use
- Choose a “cutoff” frequency
- Generate taps for a series of sinusoids that span the desired frequency range
- Add the taps together
- Normalize to have energy 1

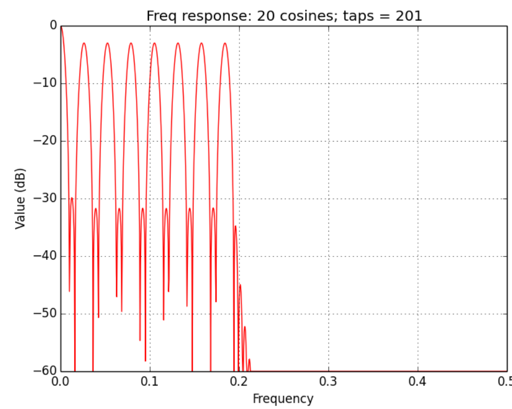
Example: A LPF

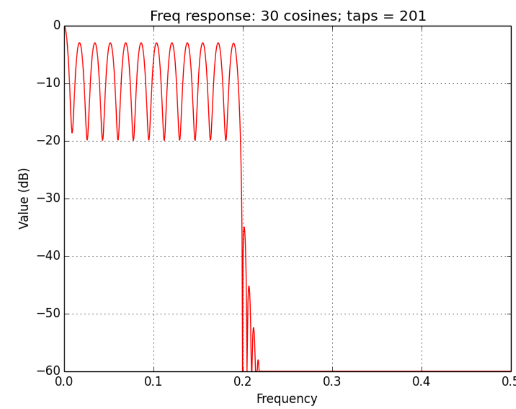
- **Ideal frequency response**

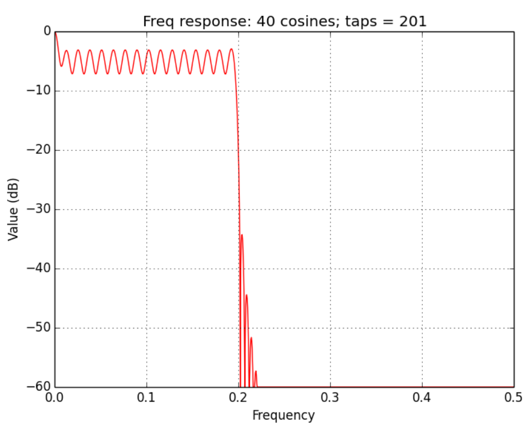


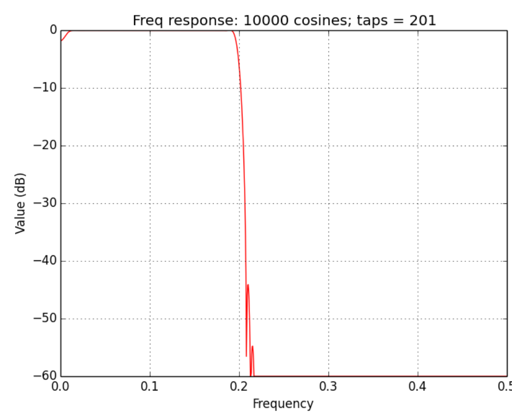
Letting progressively more sinusoids through

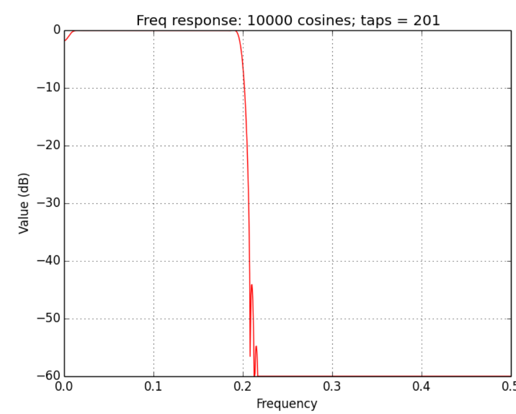
CU Boulder







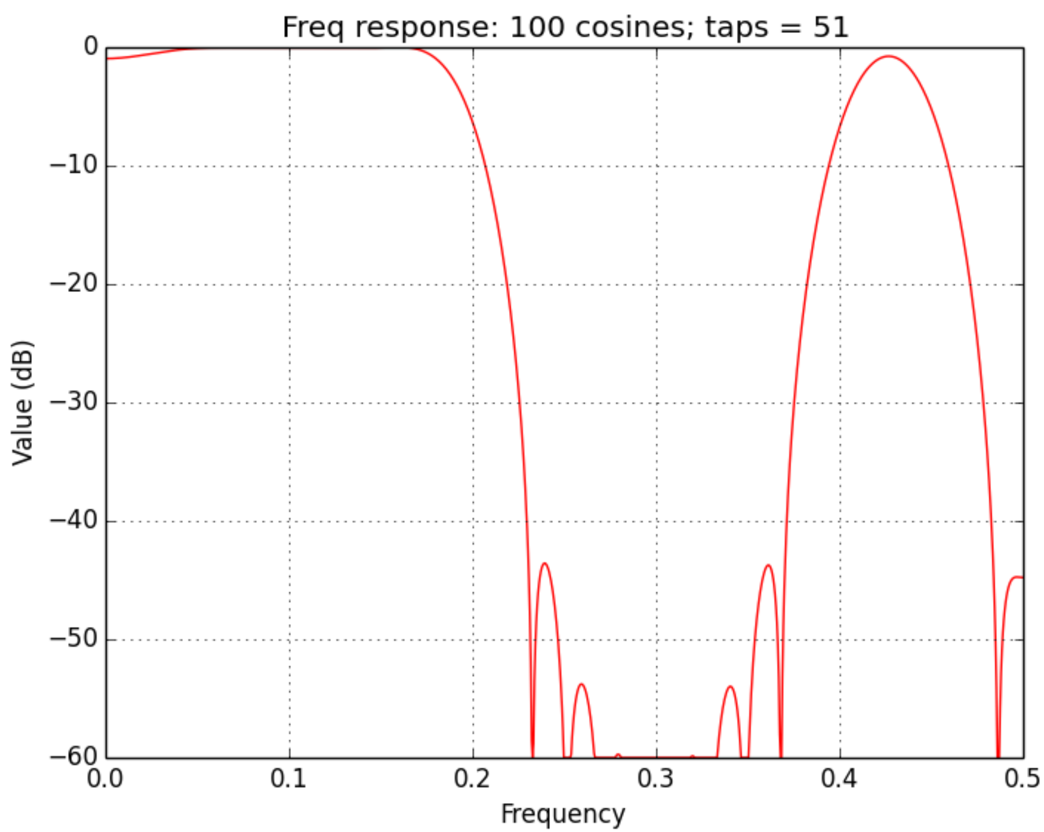




An optional Hann windows was applied to the taps before summing

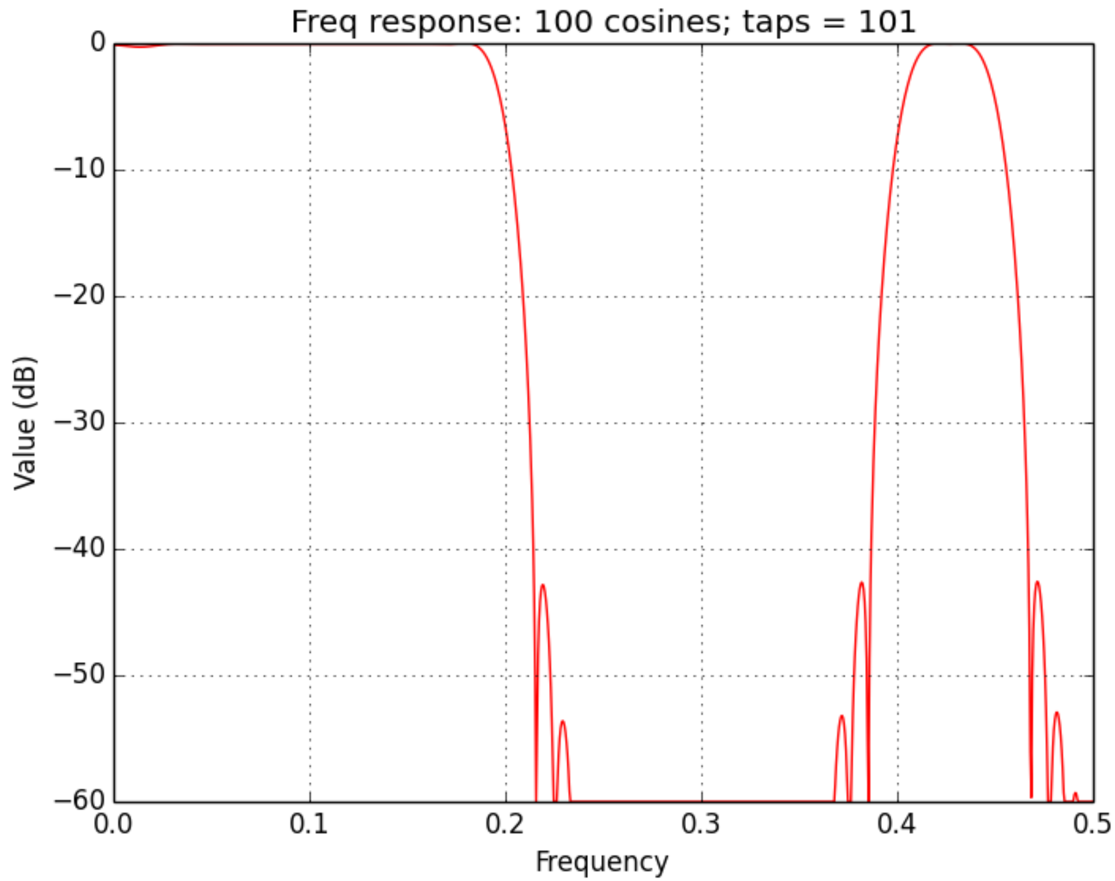
A more complex frequency response (51 taps)

Desired frequency response is 1 for *f* <= 0.2 and 0.4 <= *f* <= 0.45



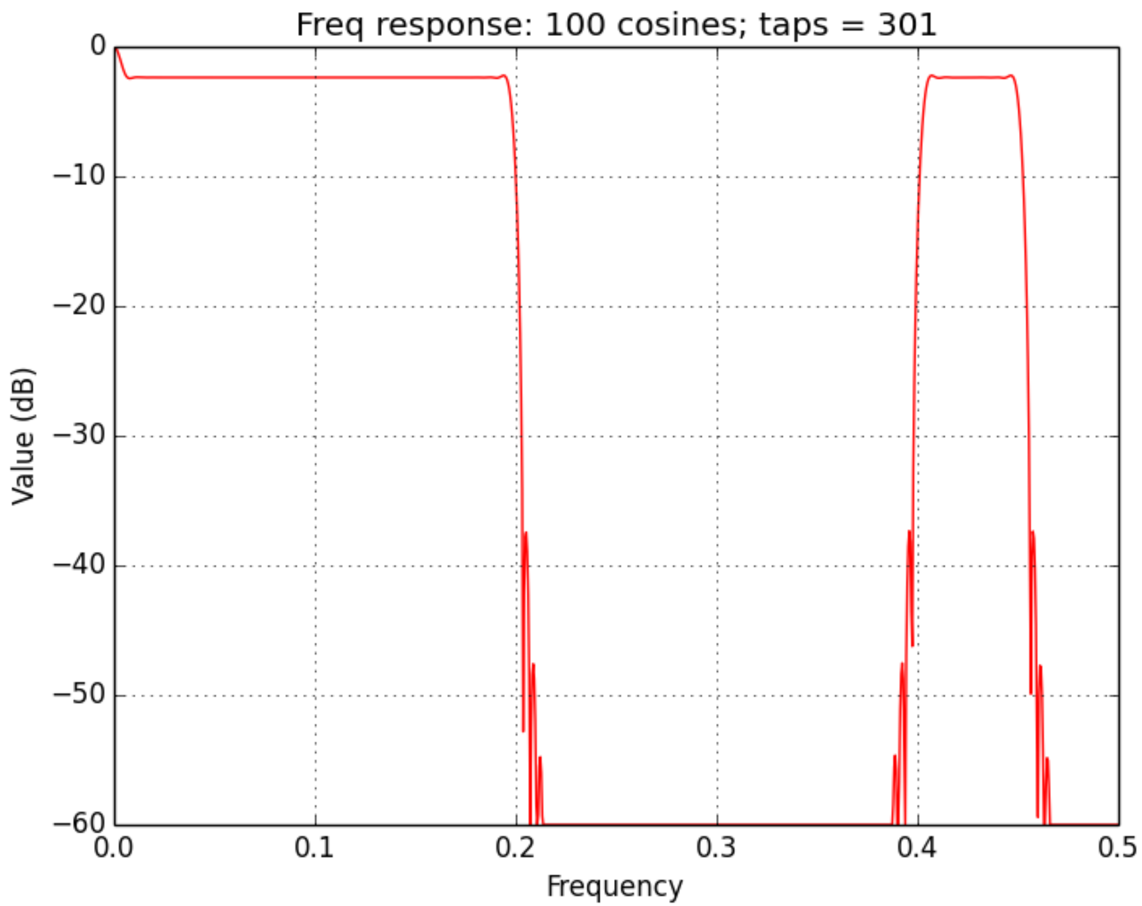
And for 101 taps...

Desired frequency response is 1 for *f* <= 0.2 and 0.4 <= *f* <= 0.45



And for 301 taps...

Desired frequency response is 1 for *f* <= 0.2 and 0.4 <= *f* <= 0.45



Designing an FIR filter

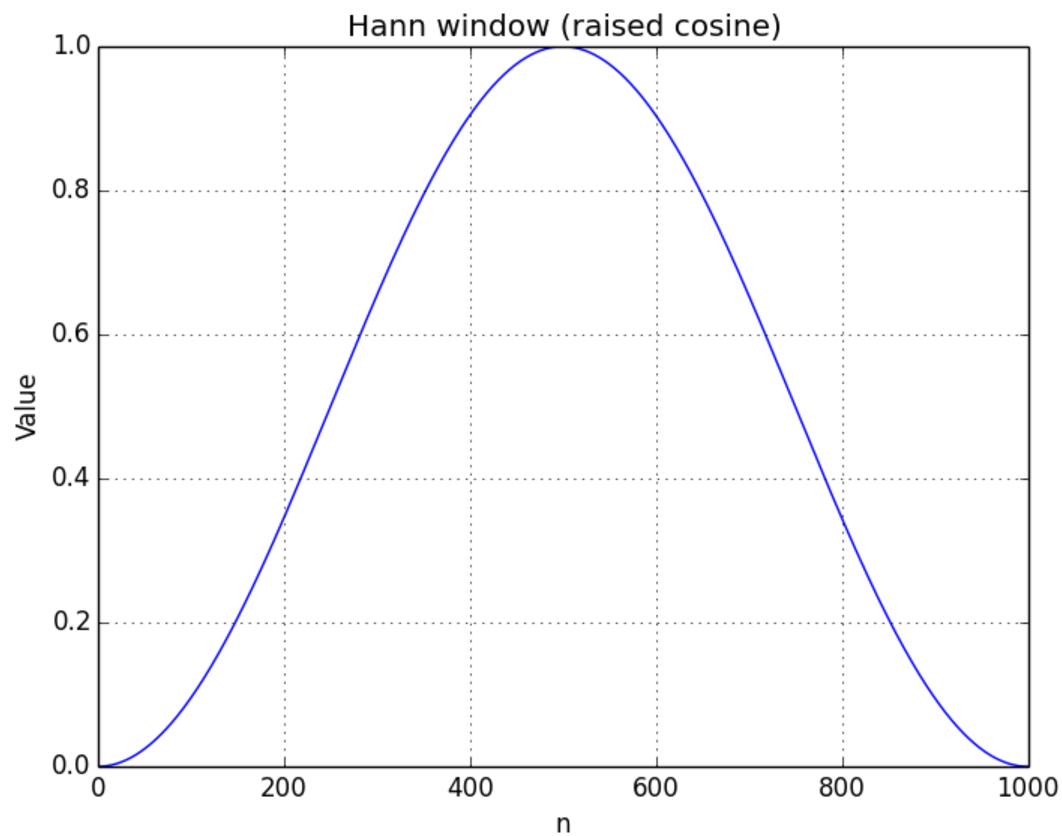
- **Define the ideal frequency response**
- **Choose a number of taps and a windowing function**
- **Pick a small frequency increment and move from f = 0 to f = ½ by this increment. For each frequency, if it is within a passband, generate sinusoidal taps at that frequency**
  - Optional: Window the taps (e.g. with a Hann Window)
  - Normalize the taps to have energy 1
  - Accumulate the taps in a final tap vector
- **Normalize the final tap vector to have energy 1**

Windowing function example

- **If we multiply a filter's taps by a tapered "window" the frequency response is changed**

Hann Window

$$w(n) = \sin^2\left(\frac{\pi n}{N-1}\right)$$



Hann windowed frequency response of a sinusoid

The main lobe is broader but the first side-lobe is smaller