Digital Signal Processing

Recursive Filters

Today's Topics

- Introduction to Recursive Filters
- Difference Equation Format
- Single Pole Implementation
- Recursion Coefficient Equations

Filter Classification

FILTER USED FOR:	FILTER IMPLEMENTED BY: Convolution *Finite Impulse Response (FIR)*	FILTER IMPLEMENTED BY: Recursion *Infinite Impulse Response (IIR)*
Time Domain *(smoothing, DC removal)*	Moving average (Ch. 15)	Single pole (Ch. 19)
Frequency Domain *(separating frequencies)*	Windowed-sinc (Ch. 16)	Chebyshev (Ch. 20)
Custom *(Deconvolution)*	FIR custom (Ch. 17)	Iterative design (Ch. 26)

FIR Filters

- So far we have worked with FIR filters
  - Finite Impulse Response filters

- The length of the impulse response is finite – Fixed number of samples in the impulse response sequence

- We have implemented these using convolution

FIR Filters

- The transfer function of the FIR filter is

$$H(z) = \sum_{k=0}^{M-1} a_k z^{-1}$$

- The filter transfer function has all zeros

FIR Filters

- The difference equation for FIR filters

$$y[n] = \sum_{k=0}^{m-1} h_m x[n-k]$$

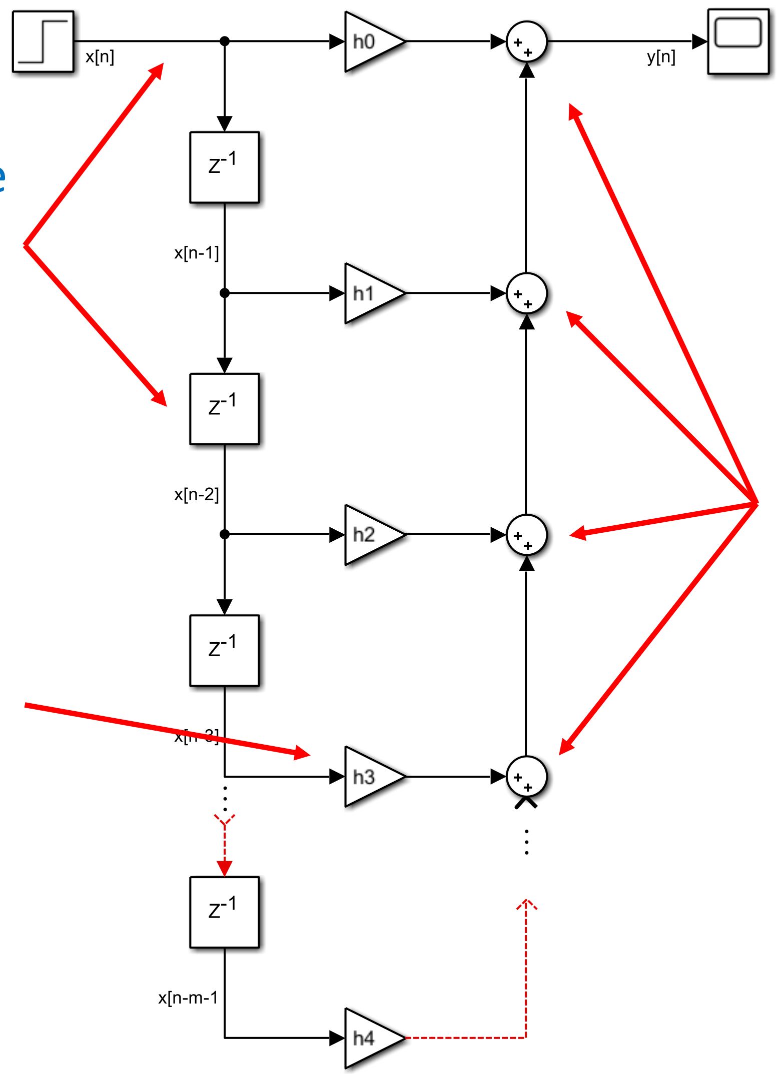
$$y[n] = h_0 x[n] + h_1 x[n-1] + h_2 x[n-3] + \cdots + h_{m-1} x[n-m+1]$$

- The output of the filter is computed only using the samples of the filter input signal

$$x[n], x[n-1] \ldots$$

FIR Flow Diagram

Input Samples enter the filter and m values are saved.



The output of the multiplications are then summed

Each value is value is multiplied by a value of the impulse response

Recursive Filters

- Recursive filters compute the next output using:
  - The filter input values $x[n], x[n-1]$ ...
  - *and* the past filter output values $y[n-1], y[n-2]$, ...

- The difference equation is:

$$y[n] = a_0 x[n] + a_1 x[n-1] + a_2 x[n-2] + a_3 x[n-3] + \cdots \\ b_1 y[n-1] + b_2 y[n-2] + b_3 y[n-3] + \cdots$$

Recursive Filters

- The transfer function of the recursive filter is

$$H(z) = \frac{\sum_{k=0}^{M-1} a_k z^{-1}}{1 - \sum_{k=0}^{N-1} b_1 z^{-1}}$$

Zeros are the roots of the numerator

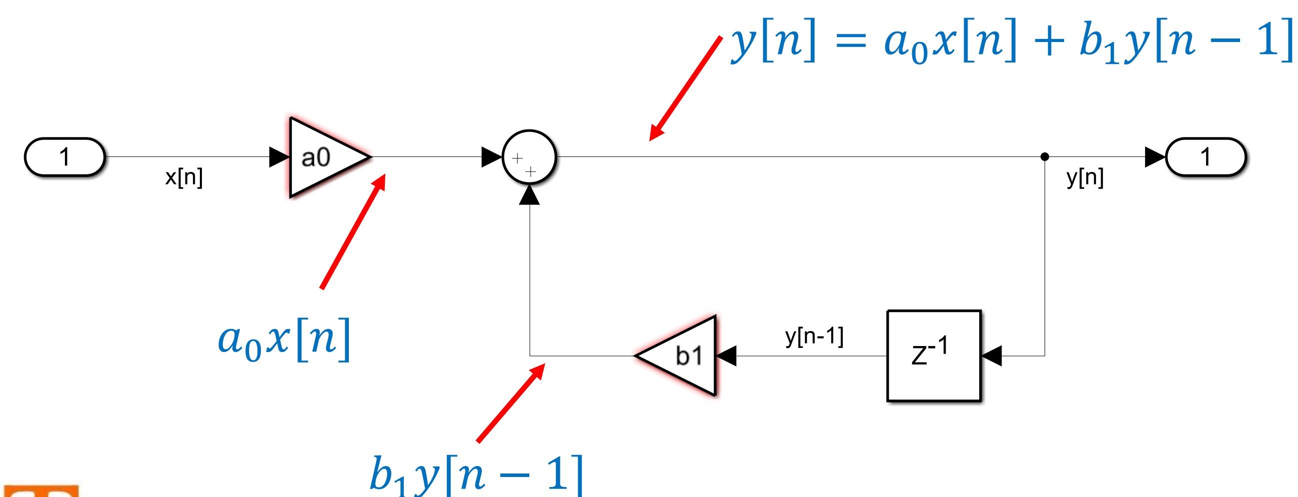
Poles are the roots of the denominator

- The recursive filter has both poles and zeros in its transfer function

Block Diagram of the Recursive Filter

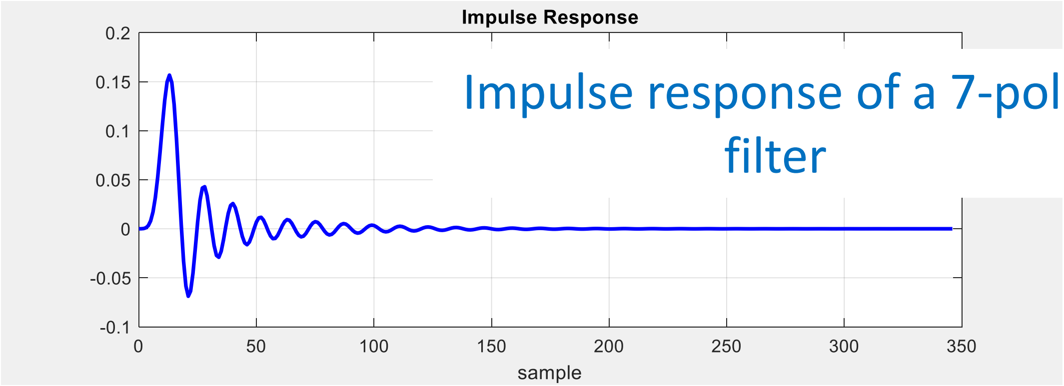
- Here is a block diagram of one implementation of a simple *single* pole recursive filter

$$y[n] = a_0 x[n] + b_1 y[n-1] \quad \Longleftrightarrow \quad H(z) = \frac{a_0}{1 - b_1 z^{-1}}$$



Recursive Filters

- The recursive filter is a feedback system
- The impulse response of the recursive filter is theoretically infinite
- Also known as Infinite Impulse Response or IIR filters



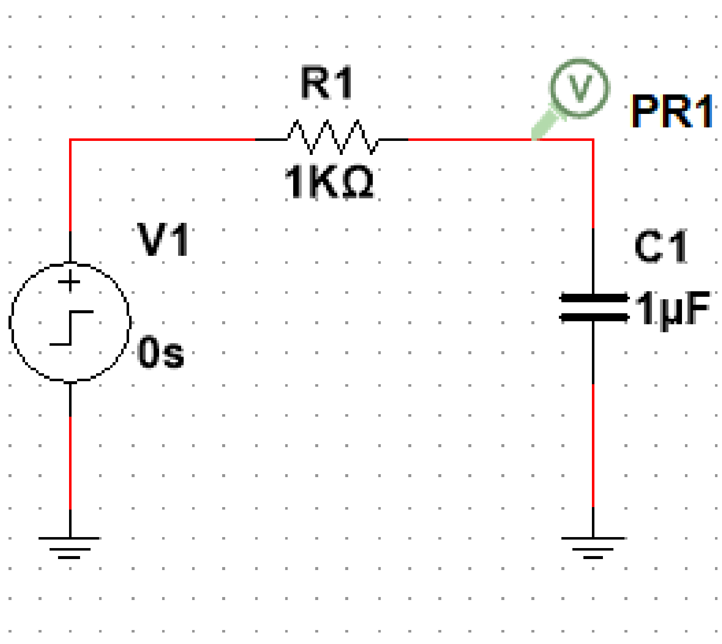
Impulse response of a 7-pole filter

Computational Requirements

- The FIR filter requires N multiply and accumulates for each output sample
- The IIR filter can achieve good attenuation characteristics with greatly reduced number of calculations
- The filter can run faster

The Single Pole RC Filter

- The single pole recursive filter is analogous to the continuous time 1-pole RC filter
- The filter has a time constant of $\tau = RC$
- It has a corner frequency of $f_c = 1/(2\pi RC)$

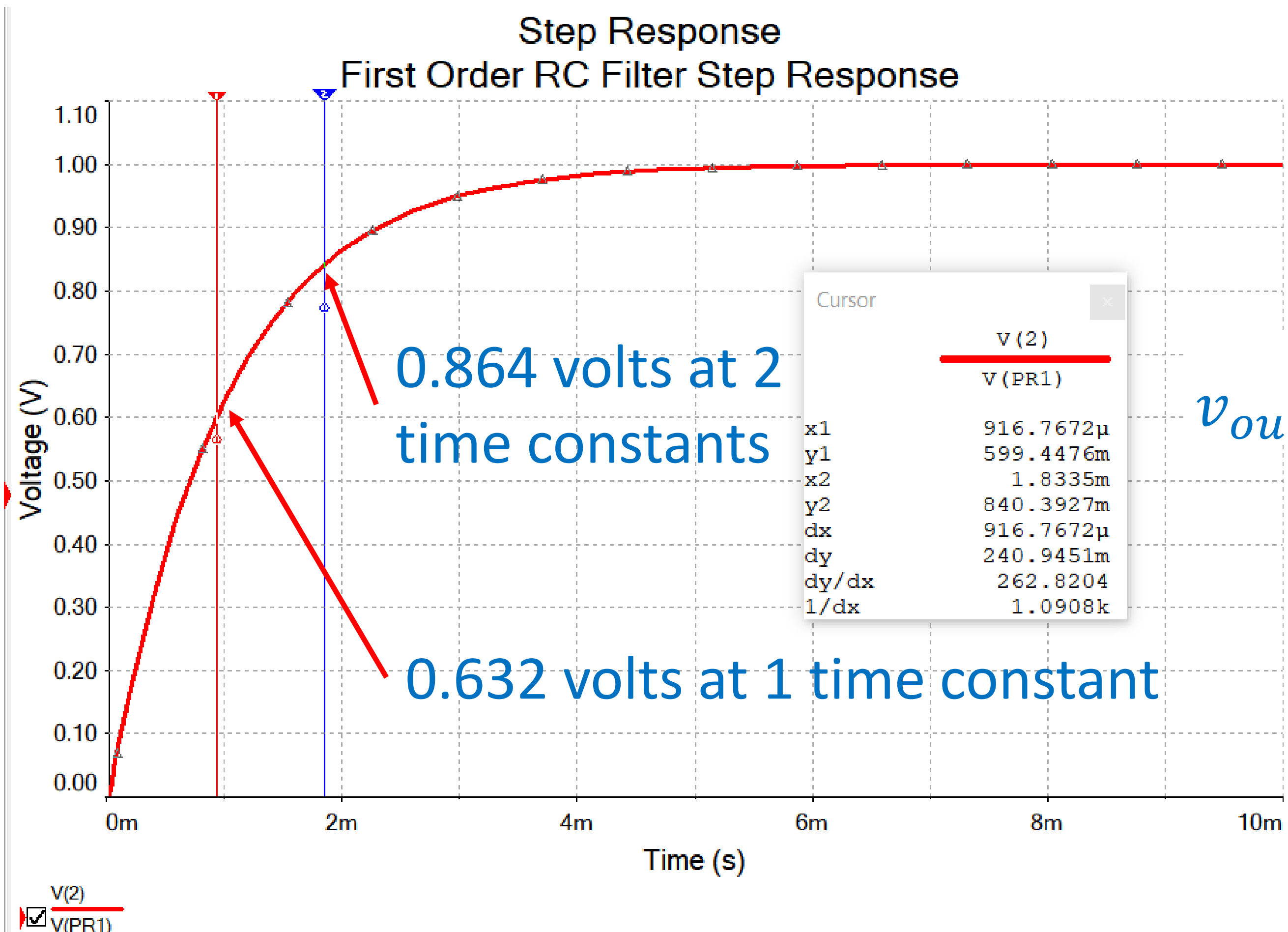


$$\tau = RC = (1K)(1\mu F) = 1mSec$$

$$f_c = \frac{1}{2\pi\tau} = \frac{1}{2\pi(1mSec)} = 159\ Hz$$

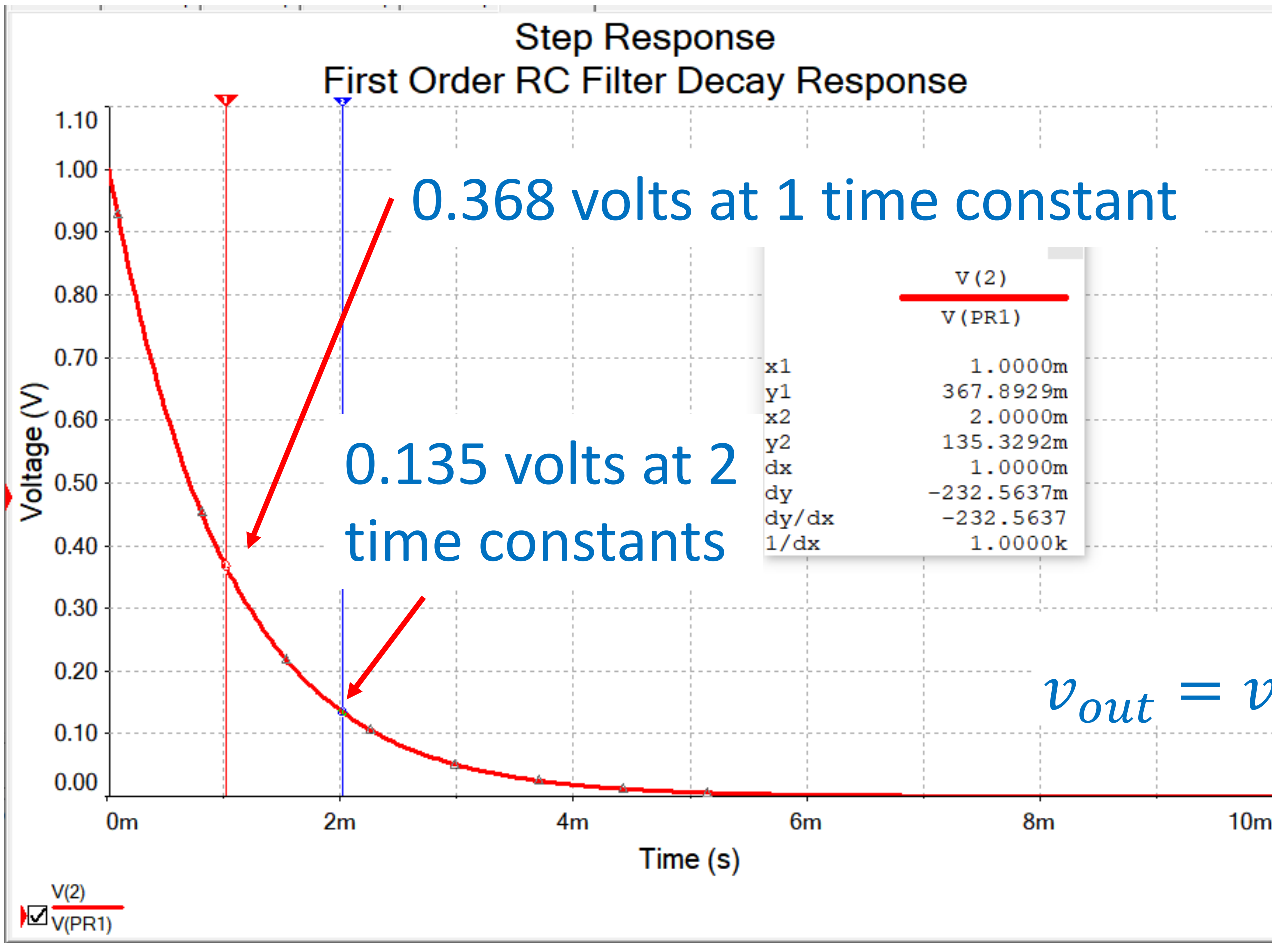
Single Pole RC Filter Step Response

- The step response of the RC filter for a 1V input



$$v_{out} = v_{in}\left(1 - e^{-\frac{t}{\tau}}\right)$$

Decay Response of the RC Filter



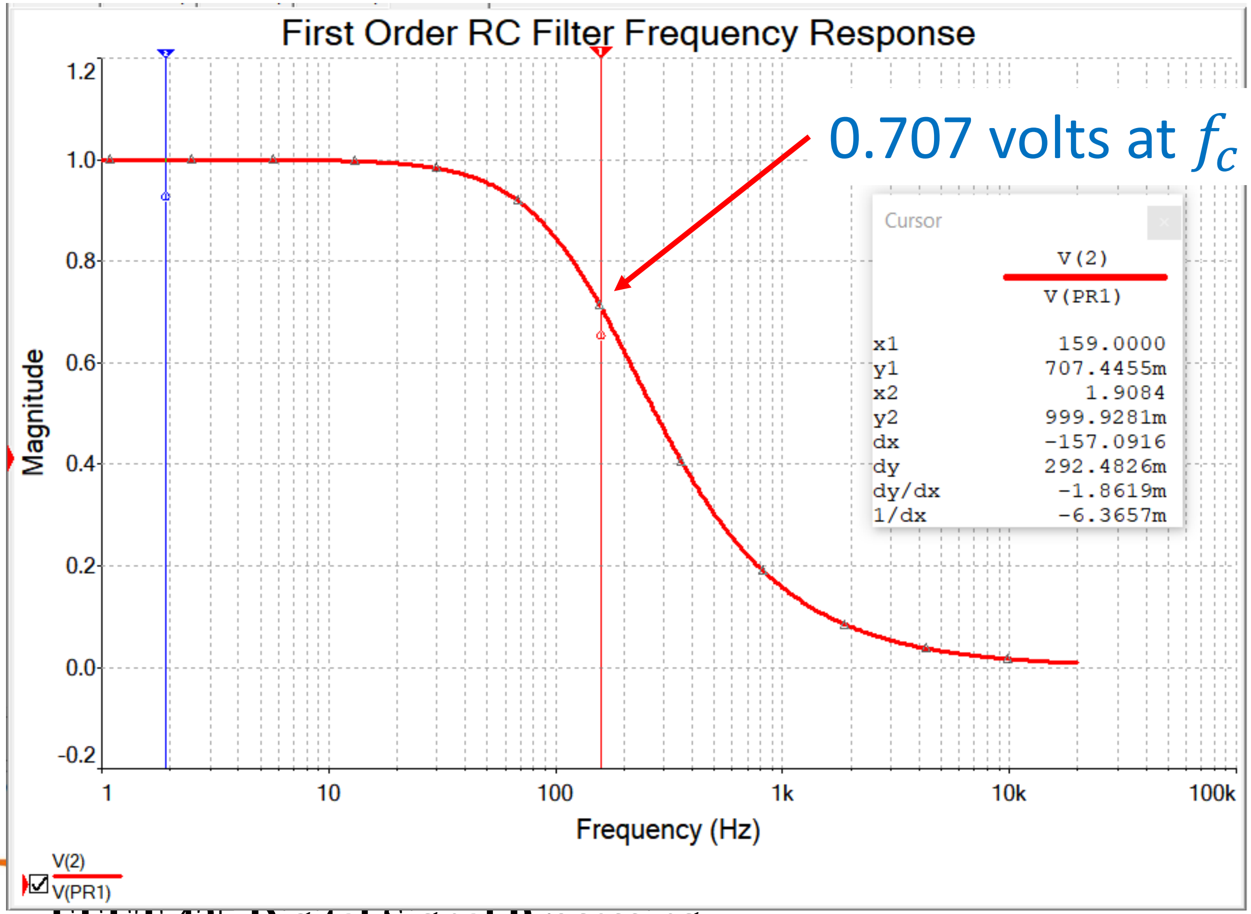
0.368 volts at 1 time constant

0.135 volts at 2 time constants

$$v_{out} = v_{in}(e^{-\frac{t}{\tau}})$$

Single Pole RC Filter Frequency Response

- Frequency Response of the RC Filter

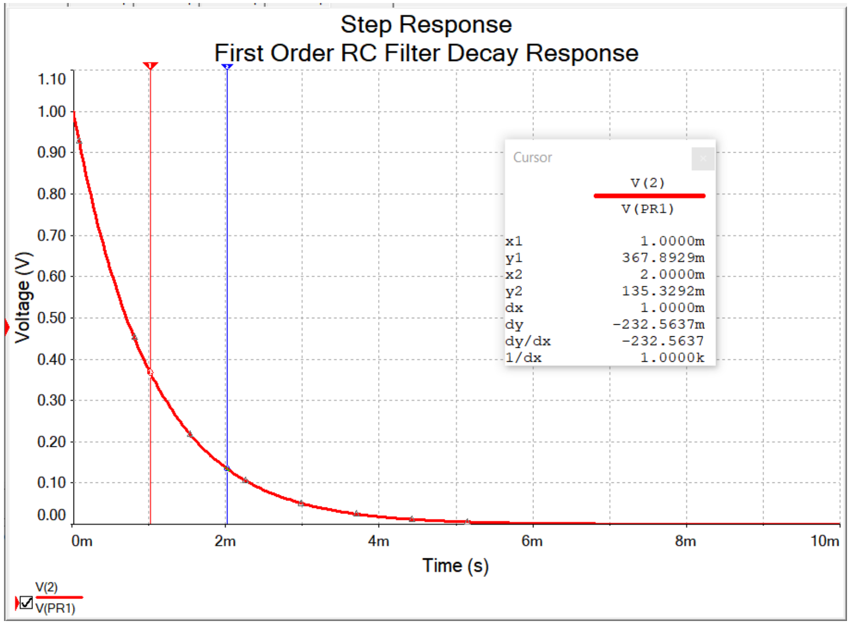


The Single Pole RC Filter

- The amount of decay versus time for an RC filter is:

$$V_{out} = V_{in} e^{-t/\tau}$$

$$\frac{V_{out}}{V_{in}} = e^{-t/\tau} \qquad \tau = RC$$



So, in one time constant the output would be $e^{-1} = 0.368$ of the input

The Single Pole Recursive Filter

- Analogous to the time constant τ of an RC filter the value d is the time constant for a discrete single pole filter
  - d is in <u>samples</u>
- The amount of decay in one time constant is the same for the recursive filter

$$x[n] = x[0]e^{-n/d}$$

$$\frac{x[n]}{x[0]} = x = e^{-n/d}$$

The Single Pole Recursive Filter

- The amount of decay from sample to sample is

$$x[n] = x[0]e^{-n/d} \qquad \frac{x[1]}{x[0]} = x = e^{-1/d}$$

- We can relate this to the corner frequency of the filter

$$f_c = \frac{1}{2\pi d} \qquad d = \frac{1}{2\pi f_c} \qquad x = e^{-2\pi f_c}$$ For the recursive filter

$$f_c = \frac{1}{2\pi\tau}$$ For the analog RC filter

Calculating the Filter Coefficients

- From the value of x we can easily find the single pole filter coefficients, a_0 and b_1

$$y[n] = a_0 x[n] + b_1 y[n-1]$$

$$x = e^{-1/d} \qquad \text{or} \qquad x = e^{-2\pi f_c}$$

Then:

$$b_1 = x$$

$$\mathrm{a}_0 = 1 - x$$

First Order Recursive LPF Filter Example

- Create a first order recursive filter with a time constant, d of 20 samples

Find the value of x: $x = e^{-1/d} = e^{-\left(\frac{1}{20}\right)} = 0.9512$

This is an equivalent to: $f_c = \frac{1}{2\pi d} = \frac{1}{2\pi 20} = 0.00795$

Where f_c is relative to the sampling rate

Then $b_1 = x = 0.9512$

$$\mathrm{a}_0 = 1 - x = 0.0488$$

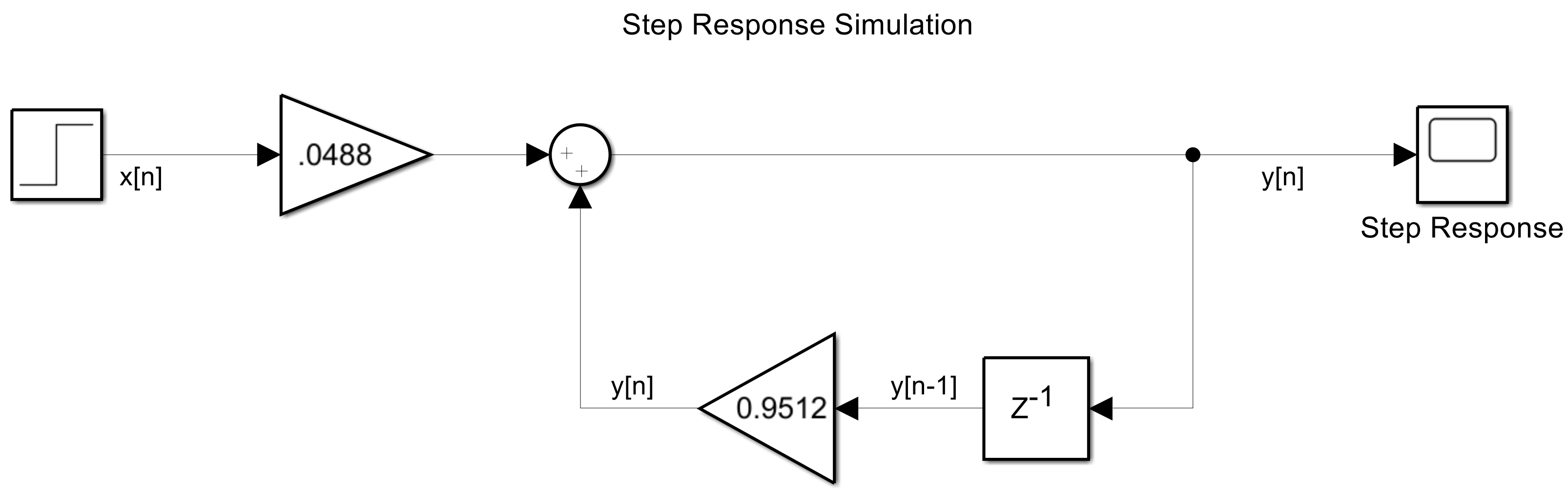
Recursive LPF Filter Example

- For a time constant of 20 samples

$$b_1 = 0.9512$$

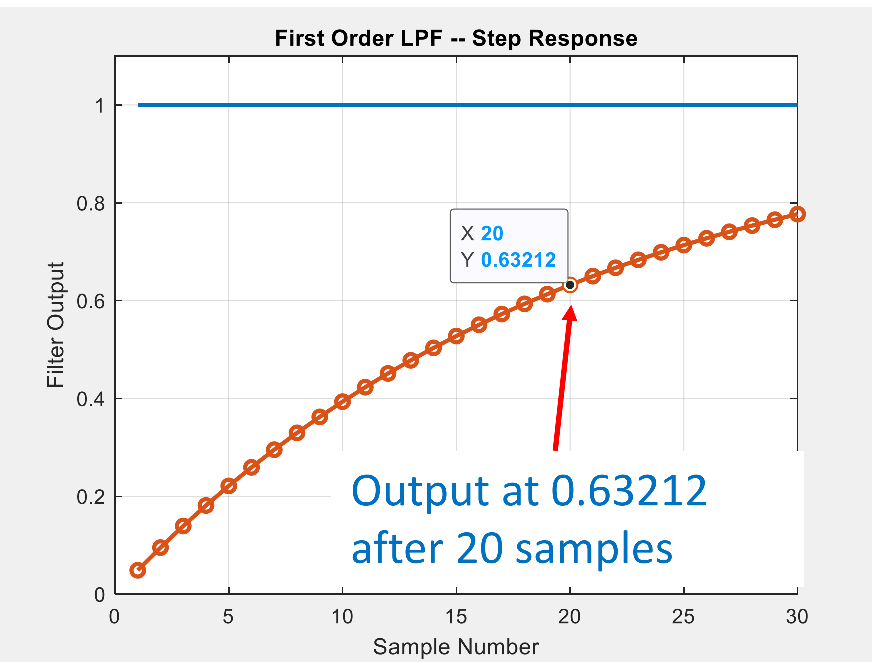
$$\mathrm{a}_0 = 0.0488$$

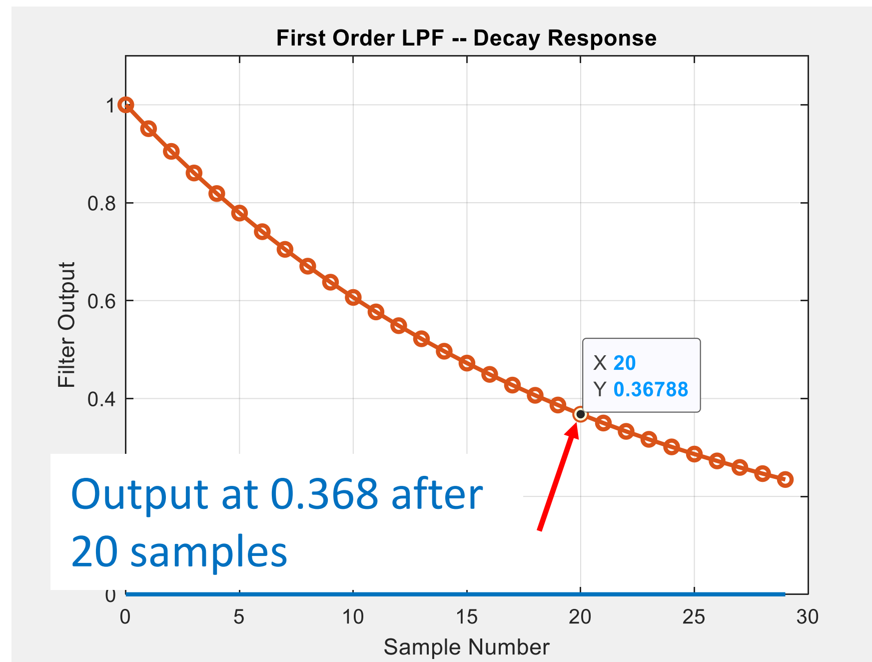
$$y[n] = a_0 x[n] + b_1 y[n-1] = 0.0488\,x[n] + 0.9512\,y[n-1]$$



First Order Recursive Filter Example

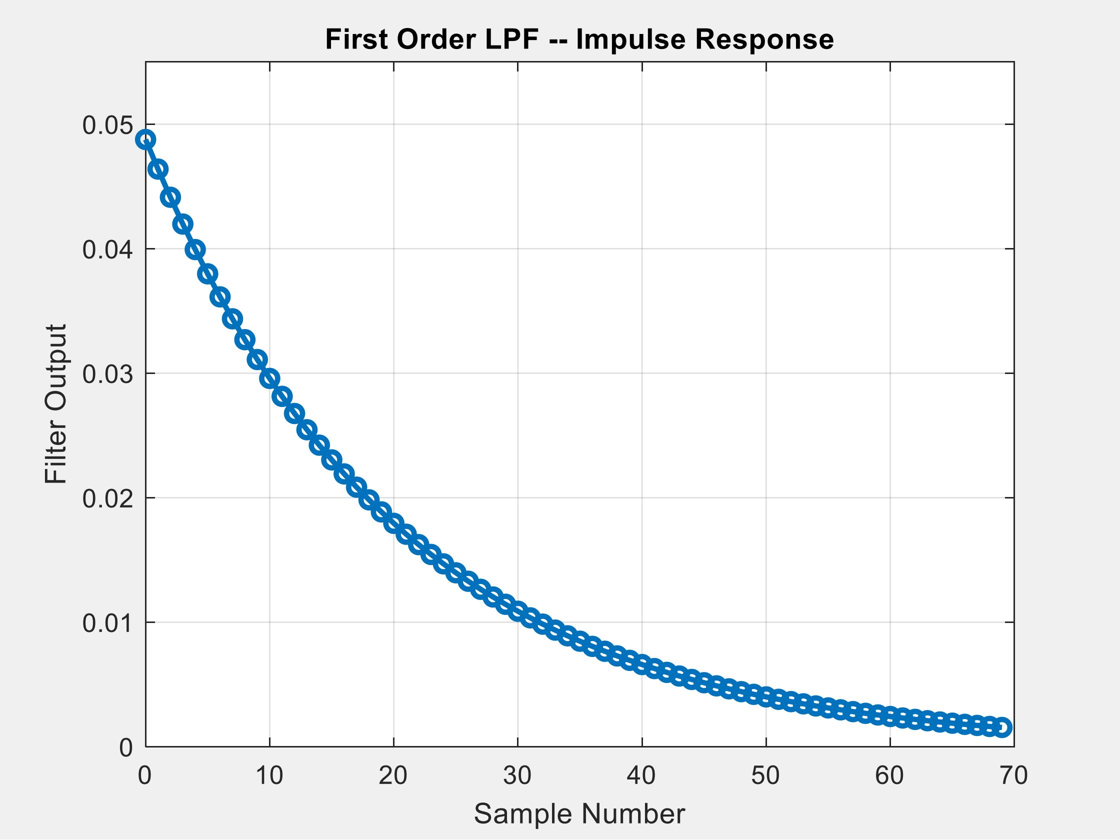
- First order LPF Step and Decay responses for a filter with a time constant of 20 samples



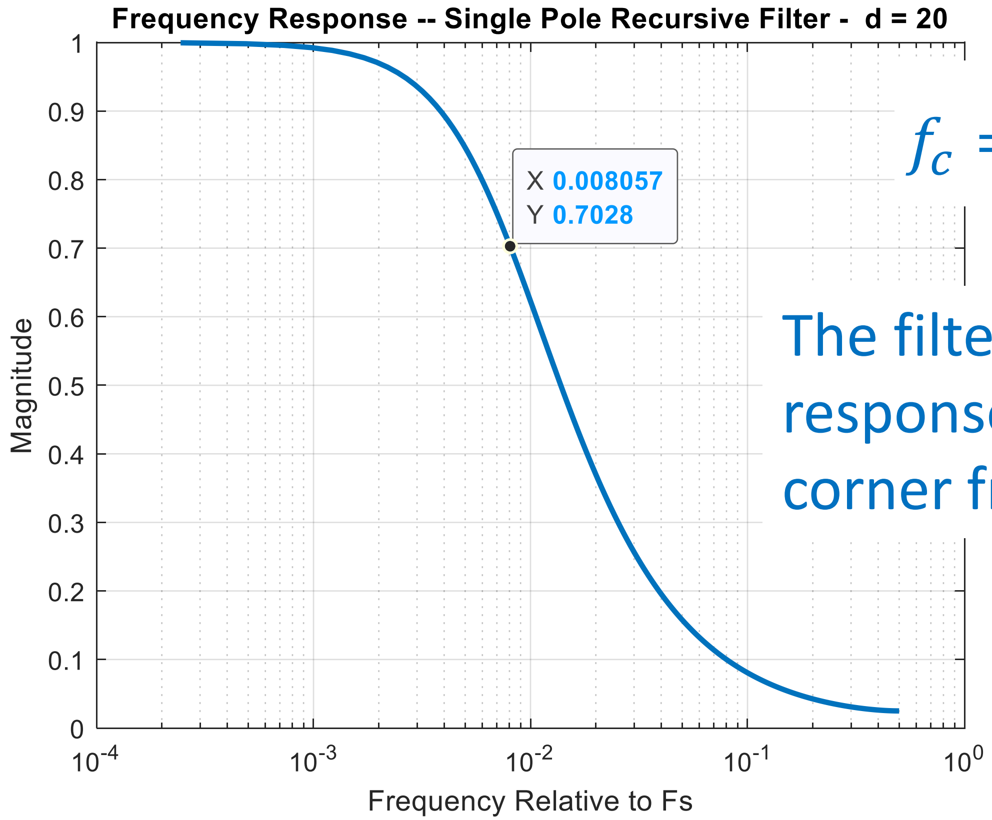


First Order Recursive Filter Impulse Response

- Placing an impulse on the input results in the impulse response



First order Recursive LPF Frequency Response



$$f_c = \frac{1}{2\pi d} = \frac{1}{2\pi\ 20} = 0.008$$

The filter has an amplitude response that is .707 (-3 dB) at the corner frequency.

How to Find the Frequency Response Using MATLAB

- MATLAB uses the numerator and denominator from the Z-transform
- Write the difference equation as a Z-Transform

$$y[n] = a_0\, x[n] + b_1 y[n-1]$$

$$Y(z) = a_0 X(z) + b_1 Y(z) z^{-1}$$

$$Y(z)(1 - b_1 z^{-1}) = a_0 X(z)$$

$$\frac{Y(z)}{X(z)} = H(z) = \frac{a_0}{1 - b_1 z^{-1}}$$

How to Implement in MATLAB

- Write the numerator and denominator of the Z-transform as vectors in decreasing order
- Use the freqz command to find frequency response

MATLAB

$$\frac{Y(z)}{X(z)} = H(z) = \frac{a_0}{1 - b_1 z^{-1}} \Rightarrow \begin{array}{l} Numerator = [a_0] \\ Denominator = [1, -b_1] \end{array}$$

$$[h, w] = freqz(Numerator, Denominator, numPoints)$$

h is the complex frequency response

w is a vector of frequencies relative to 2π

MATLAB Example

First order Recursive Demo

$$\frac{Y(z)}{X(z)} = H(z) = \frac{a_0}{1 - b_1 z^{-1}}$$

```
timeConstant = 20;
x = exp(-1/timeConstant);


b1 = x;
a0 = 1-x;


%  Find the frequency response

[h,w] = freqz(a0, [1,-b1], 2048);

figure
semilogx(w/(2*pi), abs(h),'LineWidth',2)
grid on
title('Frequency Response -- Single Pole Recursive Filter -  d = 20')
xlabel('Frequency Relative to Fs')
ylabel('Magnitude')
```

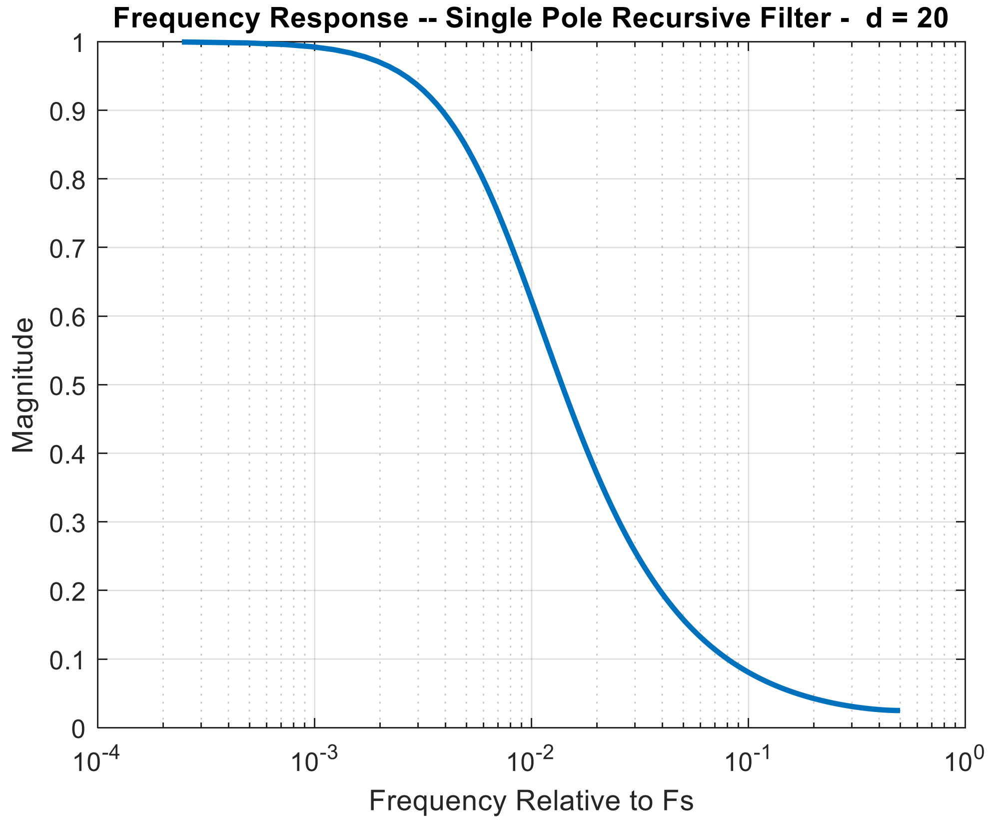
Computing the values of b_1 and a_0 from the time constant

Computing the frequency response
Using Numerator and Denominator vectors

MATLAB Example

- Frequency response of Single Pole Recursive Filter

$$y[n] = 0.0488\, x[n] + 0.9512\, y[n-1]$$



Recursive LPF ICP

- Design a single pole low pass recursive filter with a corner frequency of 30 BPM in a system using a sample rate of 600 BPM

1. First find the corner frequency relative to the sample rate
2. Find the value of x for the filter:
3. Compute the coefficients
4. Use MATLAB to plot the frequency response

Recursive LPF ICP

NOPRINT

- Design a single pole low pass recursive filter with a corner frequency of 30 BPM in a system using a sample rate of 600 BPM

First find the corner frequency relative to the sample rate

$$f_c = \frac{f_{c\_hz}}{f_s} = \frac{30\ BPM}{600\ BPM} = 0.05$$

Find the value of x for the filter:

$$x = e^{-2\pi f_c} = e^{-2\pi \times 0.05} = 0.7304$$

Compute the Recursion Coefficients

NOPRINT

- The first order recursion coefficients are:

$$x = e^{-2\pi f_c} = 0.7304$$

Then:

$$b_1 = x = 0.7304$$

$$\mathrm{a}_0 = (1 - x) = (1 - 0.7304) = 0.2696$$

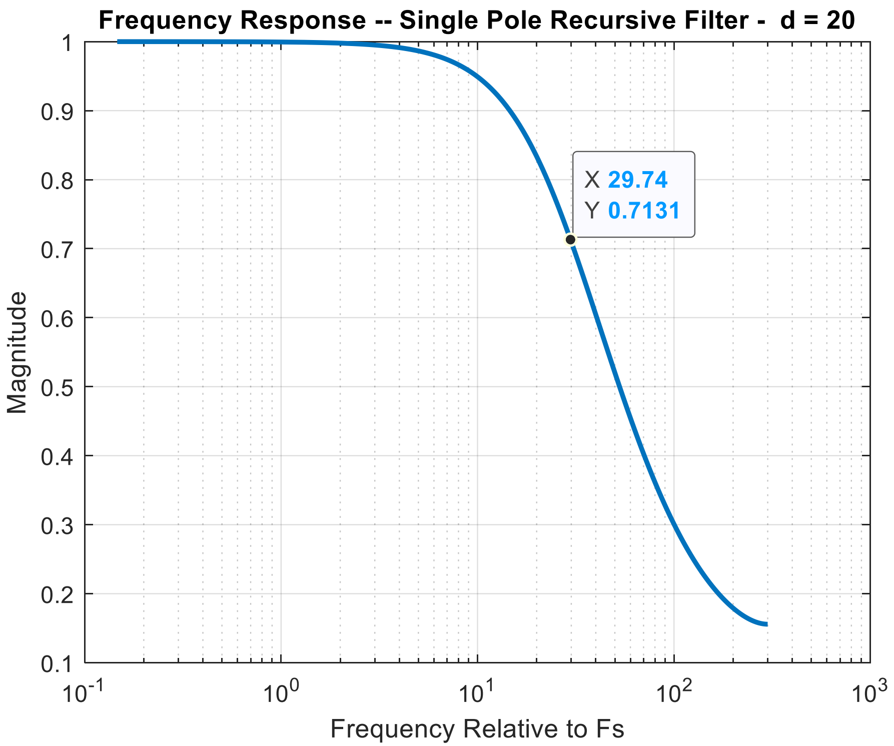
NOPRINT

Plot the Frequency Response

- Use MATLAB to compute the frequency response

```
[h,w] = freqz( 0.2696, [1,-.7304], 2048);
semilogx(w/(2*pi), abs(h),'LineWidth',2);
grid on

title('Frequency Response');
xlabel('Frequency (Relative to Fs)');
ylabel('Magnitude')
```



Recursive First Order High Pass Filter

- In a similar way a 1st order high pass filter can be designed
- The high pass filters has an additional coefficient a_1

$$y[n] = a_0\, x[n] + a_1\, x[n-1] + b_1 y[n-1]$$

- The coefficients are also calculated differently

$$x = e^{-2\pi f_c} \qquad a_0 = (1+x)/2 \qquad a_1 = -(1+x)/2$$

$$b_1 = x$$

Recursive First Order High Pass Filter Example

- Design a 1st order high pass filter with a corner frequency of 40 bpm relative to a sample rate of 600 bpm

Find the corner frequency relative to the sample rate ➡ $f_c = \frac{f_{hpf}}{f_s} = \frac{40}{600} = 0.0667$

Find the value of x: ➡ $x = e^{-(2\pi f_c)} = e^{-(2\pi\ 0.0667)} = 0.6578$

Then

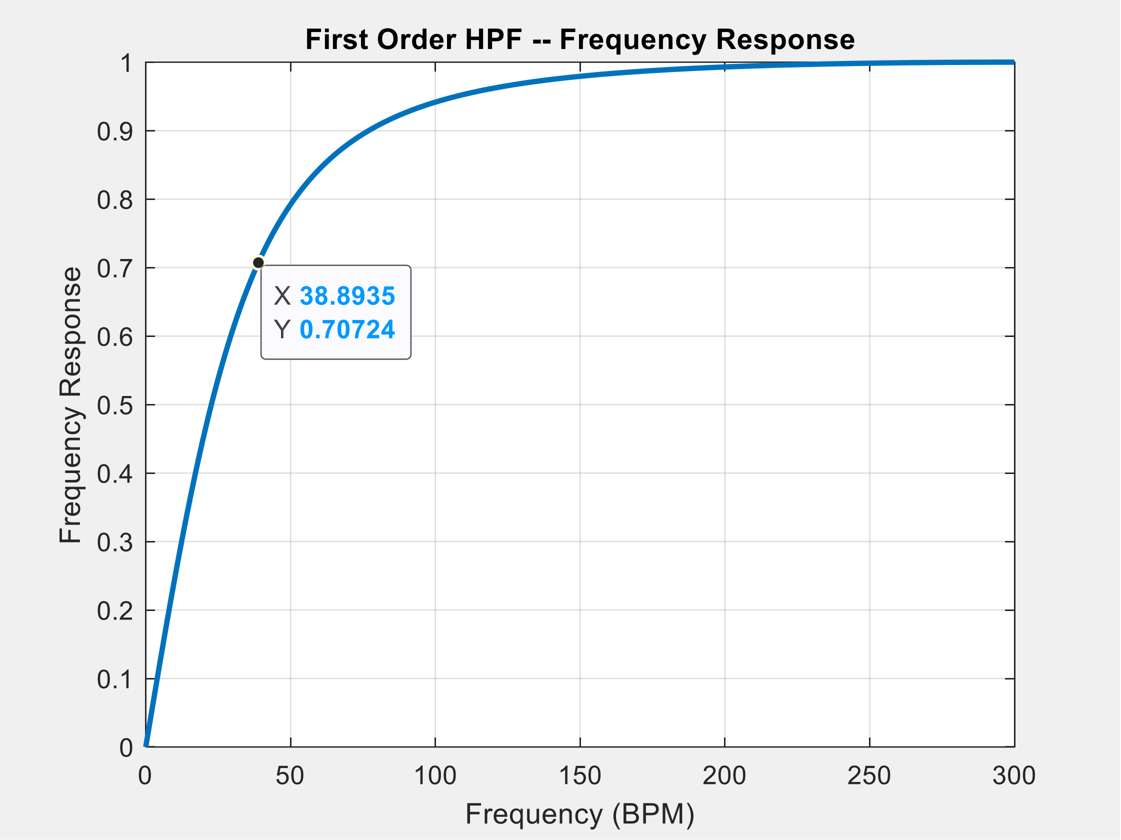
$$b_1 = x = 0.6578$$

$$\mathrm{a}_0 = \frac{1+x}{2} = 0.8289$$

$$\mathrm{a}_1 = -\frac{1+x}{2} = -0.8289$$

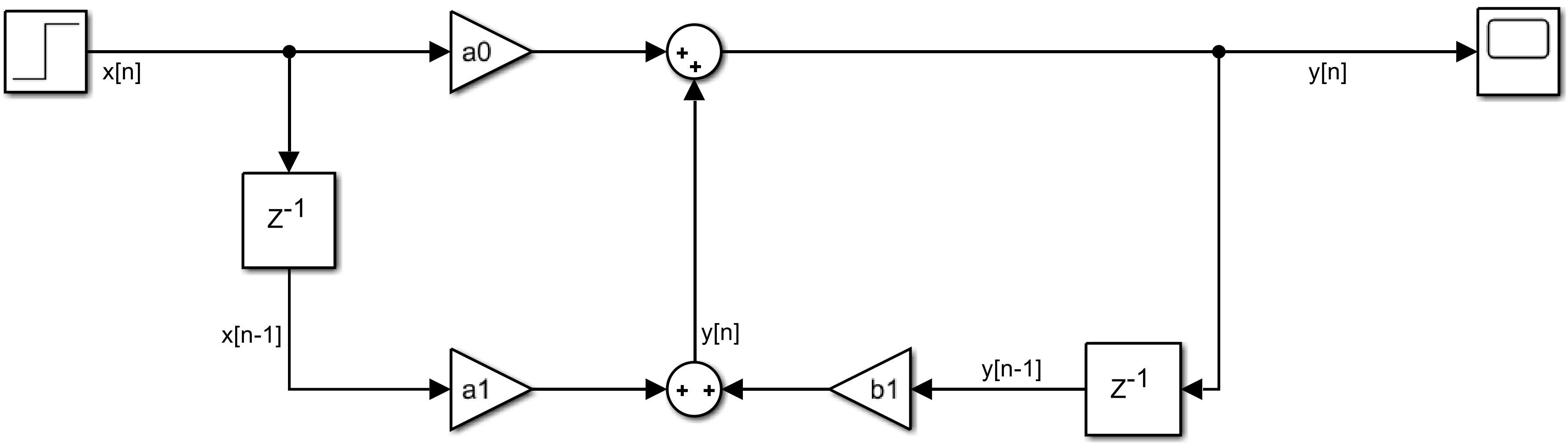
Recursive First Order High Pass Filter Example

- The corner frequency of the filter is at 40 bpm



Recursive First Order High Pass Filter Example

- The HPF has an additional input coefficient
- The feedback path is the same.



Recursive HPF ICP

- Design a single pole high pass recursive filter with a corner frequency of 5 kHz in a system using a sample rate of 20 kHz

1. First find the corner frequency relative to the sample rate
2. Find the value of x for the filter:
3. Compute the coefficients
4. Use MATLAB to plot the frequency response

NOPRINT

Recursive HPF ICP

- Design a single pole high pass recursive filter with a corner frequency of 5 kHz in a system using a sample rate of 20 kHz

First find the corner frequency relative to the sample rate

Find the value of x for the filter:

$$f_c = \frac{f_{c\_hz}}{f_s} = \frac{5\ kHz}{20\ kHz} = 0.25$$

$$x = e^{-2\pi f_c} = e^{-2\pi \times 0.25} = 0.2079$$

Compute the Recursion Coefficients

NOPRINT

- The first order recursion coefficients are:

$$x = e^{-2\pi f_c} = 0.2079$$

Then:

$$b_1 = x = 0.2079$$

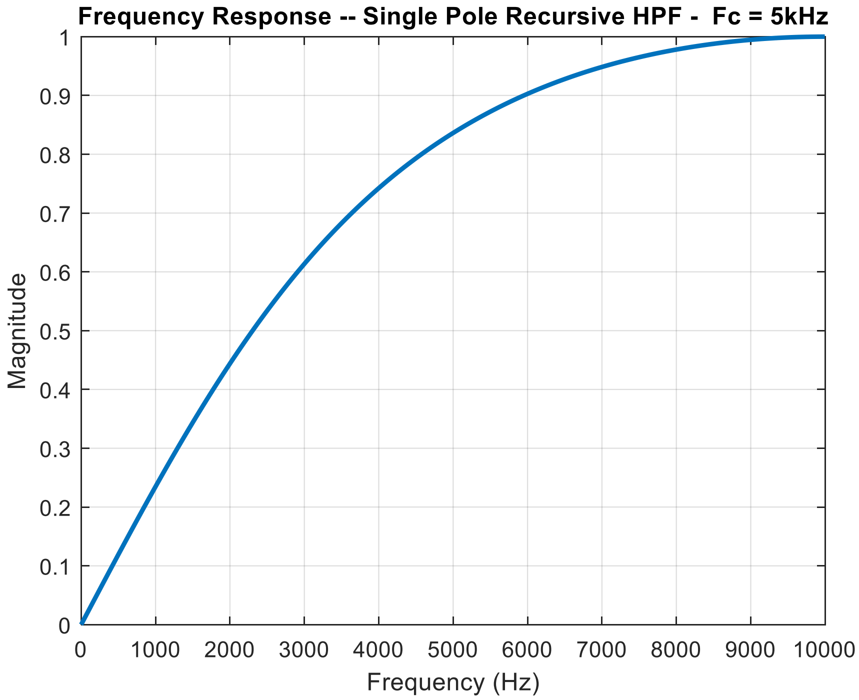
$$\mathrm{a}_0 = \frac{1+x}{2} = 0.6039$$

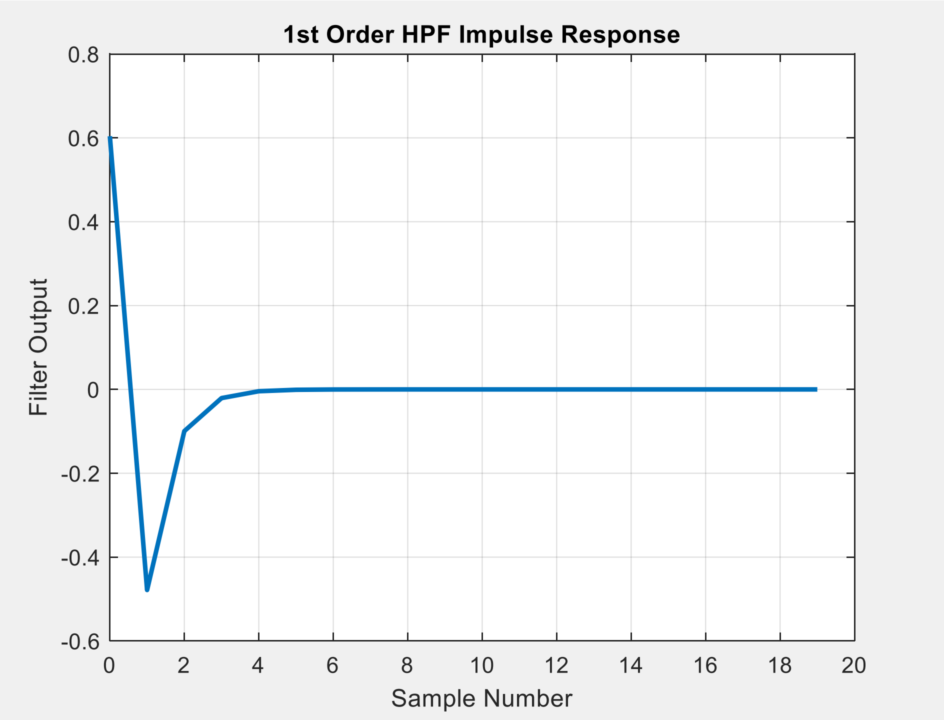
$$\mathrm{a}_1 = -\frac{1+x}{2} = -0.6039$$

1^{st} Order HPF Frequency and Impulse Response

NOPRINT

- Plot the frequency and impulse response





Infinite Impulse Response

- Here are the values of the impulse response out to 20 samples

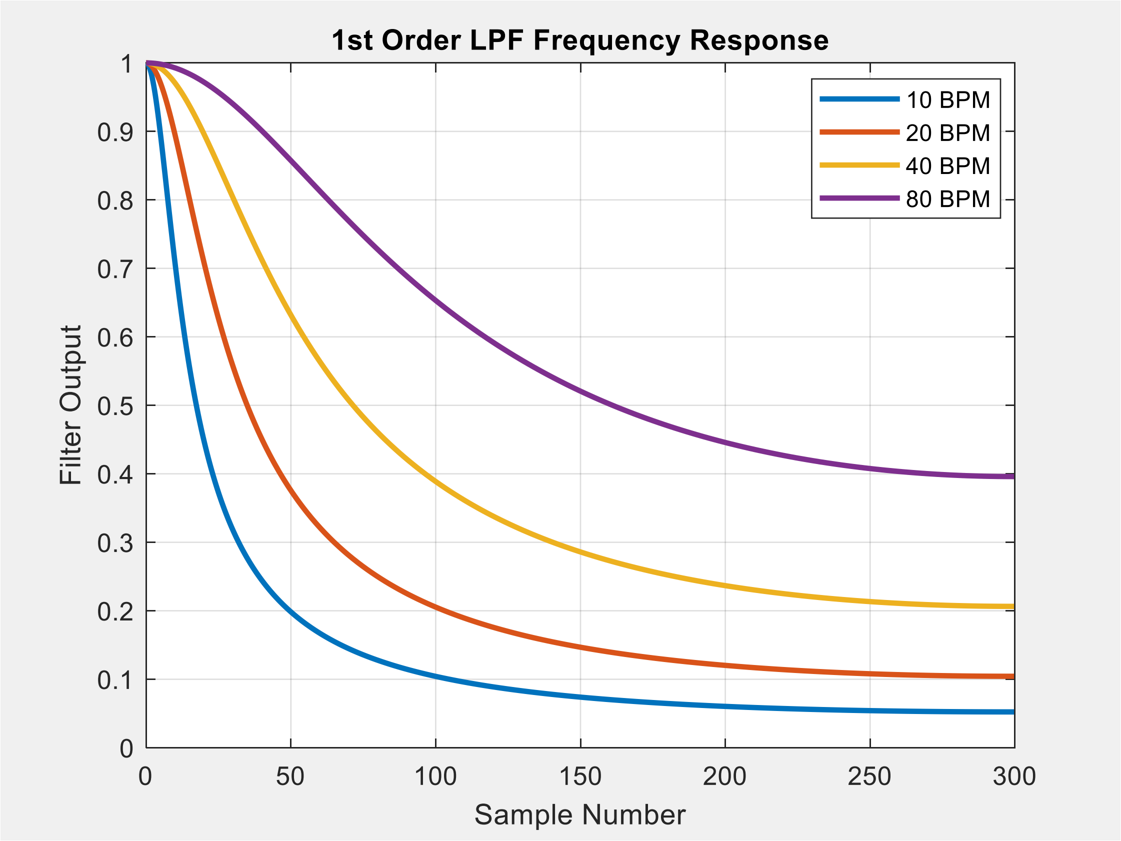
```
impResponse =

   0.603939788175381
  -0.478393040868114
  -0.099448142664816
  -0.020673237766032
  -0.004297543908601
  -0.000893371607069
  -0.000185713711201
  -0.000038606087607
  -0.000008025417136
  -0.000001668320314
  -0.000000346809720
  -0.000000072094658
  -0.000000014987007
  -0.000000003115493
  -0.000000000647647
  -0.000000000134633
  -0.000000000027987
  -0.000000000005818
  -0.000000000001209
  -0.000000000000251
```

Characteristics of the 1-pole Recursive Filter

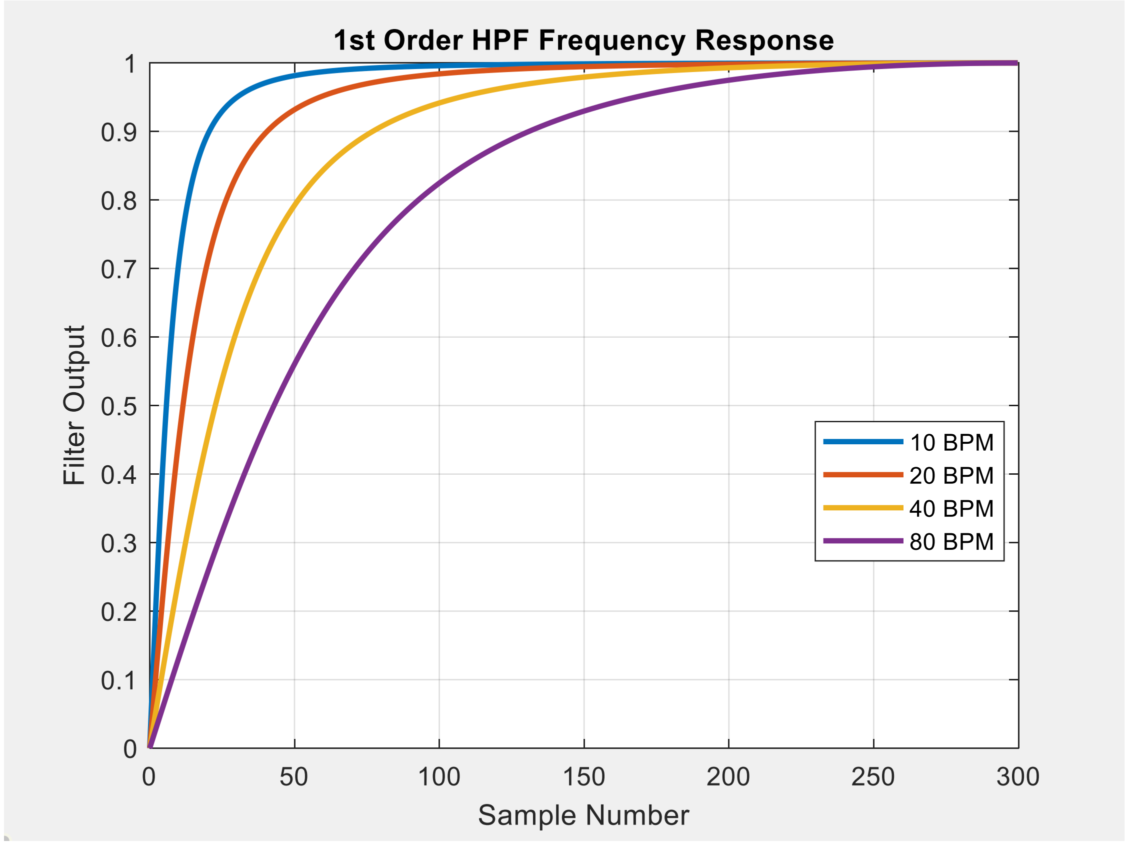
- The first order recursive filters have a smooth step response
- Good for time domain filtering.
- Does not have a sharp roll off to separate closely spaced frequencies

Comparing LPF Corner Frequencies



As I change the corner frequency the response gets lower in frequency but the relative attenuation stays the same

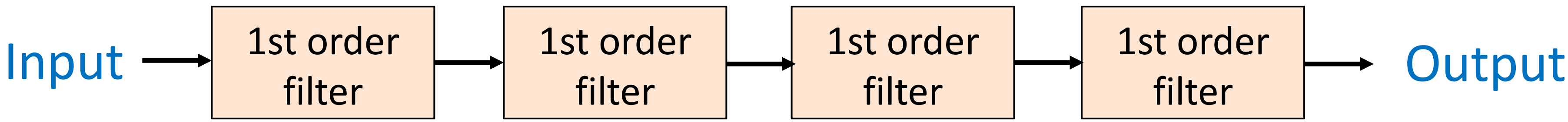
Comparing HPF Corner Frequencies

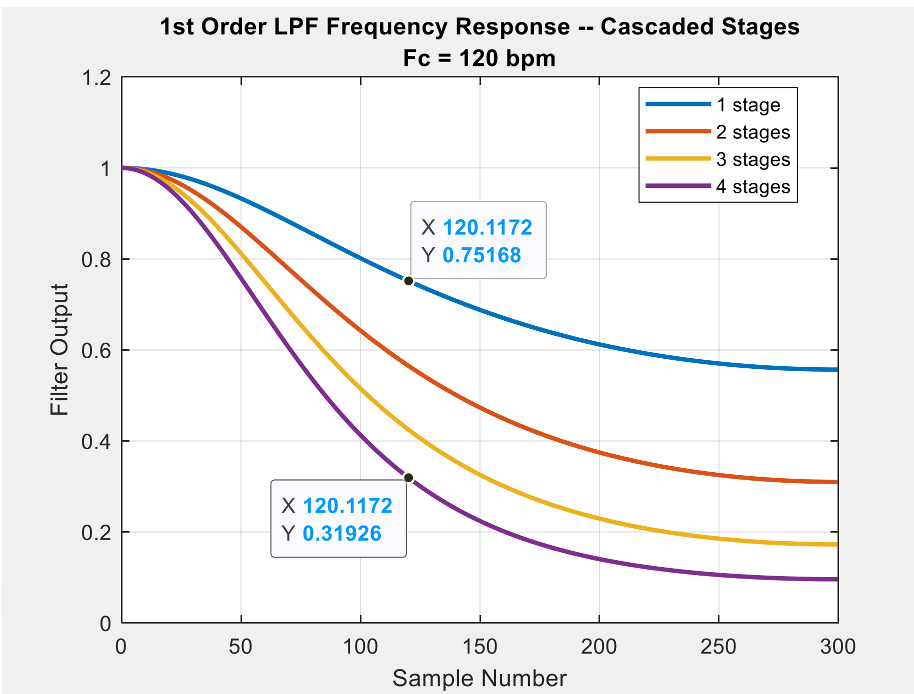


As the corner frequency gets lower the response gets lower in frequency but the relative attenuation stays the same

Cascading Multiple Stages

- Single pole filters can be cascaded, but their roll-off doesn't improve dramatically





1st Order Response of 1-4 cascaded stages

Other Filter Types

- Bandpass and bandstop (notch) filters can also be made with recursive filters (see text)

- Higher order filters can be designed with plentiful design tools

- MATLAB will be used in Lab 08 to create Butterworth and Chebyshev higher order filters

Recursive Filter Implementation

- Recursive equations are more easily implemented using single precision floating point numbers than fixed point integers.

- If integer representation is used for IIR filters, there is risk of instability due to round off error.

Summary

- Introduction to Recursive Filters
- Difference Equation Format
- Single Pole Implementation
- Recursion Coefficient Equations

49