DIGITAL FILTER COEFFICIENT DESIGN

Filter Coefficient Design

- There are many algorithms to find the coefficients for a digital filter. A DSP course will tell you digital filters can be developed that share characteristics with common analog filters such as:
  - Butterworth
  - Chebyshev
  - Bilinear transformation
  - Elliptic
- Some specify no ripple in the pass band or the stop band since this is often a desirable characteristic

Parks-McClellan Method

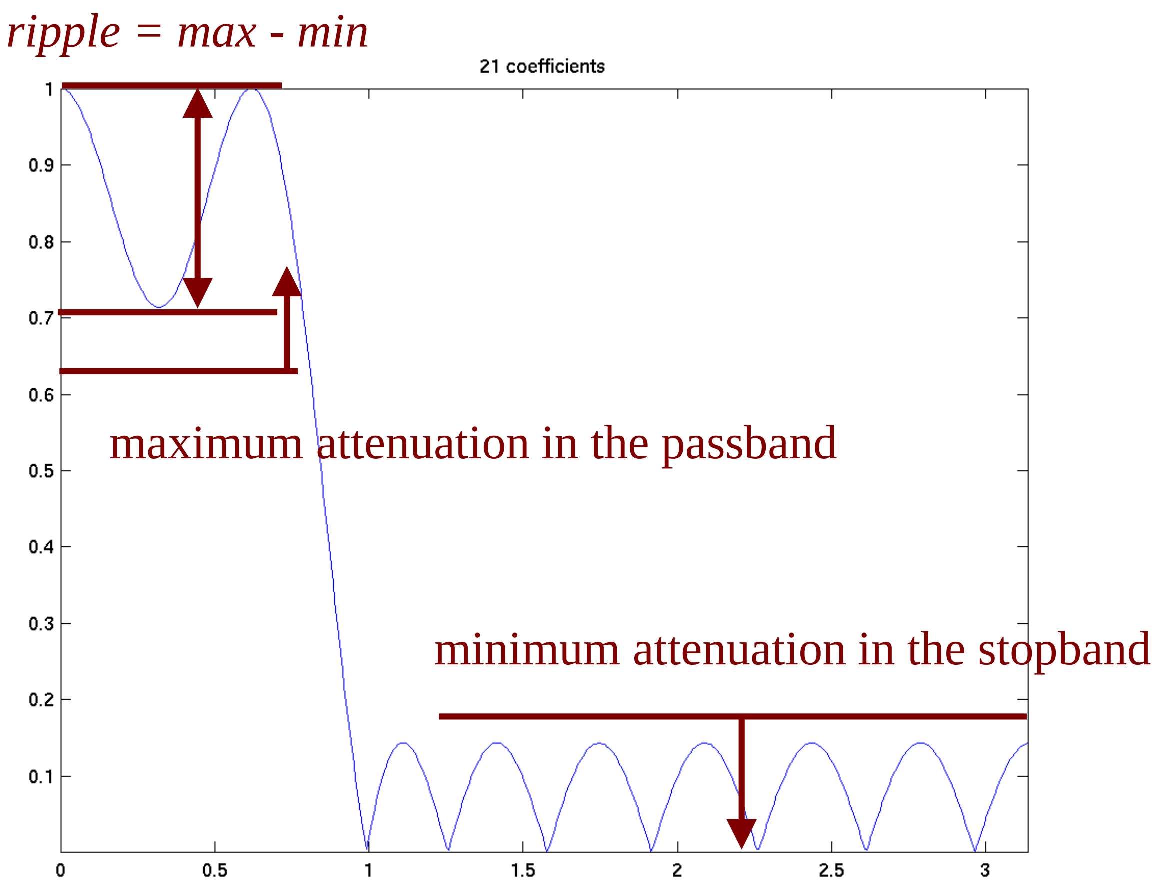
- Parks-McClellan method is a popular method for designing digital filters
  - Published in the early 70s
  - Iterative
  - Computationally efficient
  - Works by specifying the
    1. length of the filter and
    2. frequency/magnitude pairs
  - See Oppenheim & Schafer for a thorough discussion

Filter Specification

- Filter specifications are frequently given in dB as min/max attenuation/ripple over frequency regions
- An example filter specification:
  - Low-pass filter
  - Maximum +/− 4dB ripple in passband
  - Sampling frequency is 100 MHz
  - Passband from DC to 12.5 MHz
  - Minimum attenuation 22dB from 19 MHz to 50 MHz

Attenuation and Ripple

- Key filter specifications
  - Min attenuation in stopband
  - Max attenuation in passband
  - Max ripple

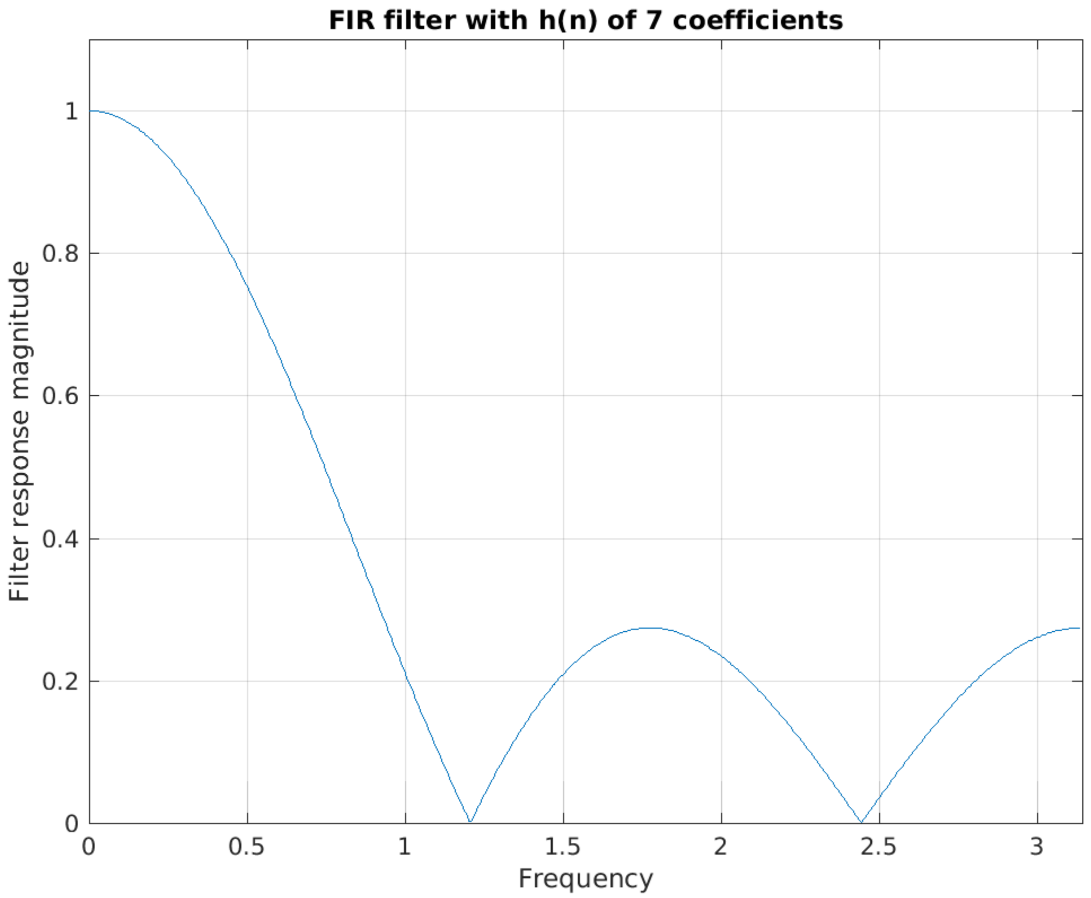


Example Filter

- The same example filter specification getting ready to be entered into matlab:
  - Low-pass
  - Notes:
    - 12.5 MHz $= 0.25\pi$
    - 19 MHz $= 0.38\pi$
    - 50 MHz $= \pi$
    - 100 MHz $= 2\pi = f_s$
  - frequencies specified as fractions of π: [0 0.25 0.38 1];
  - corresponding amplitudes: [1 1 0 0];
  - Parks-McClellan ignores every other interval starting with the second one ($0.25\pi - 0.38\pi$). But this is ok—in this example, we don't care about transition band between 0.25π and 0.38π anyway
  - Use the `remez()` function in matlab

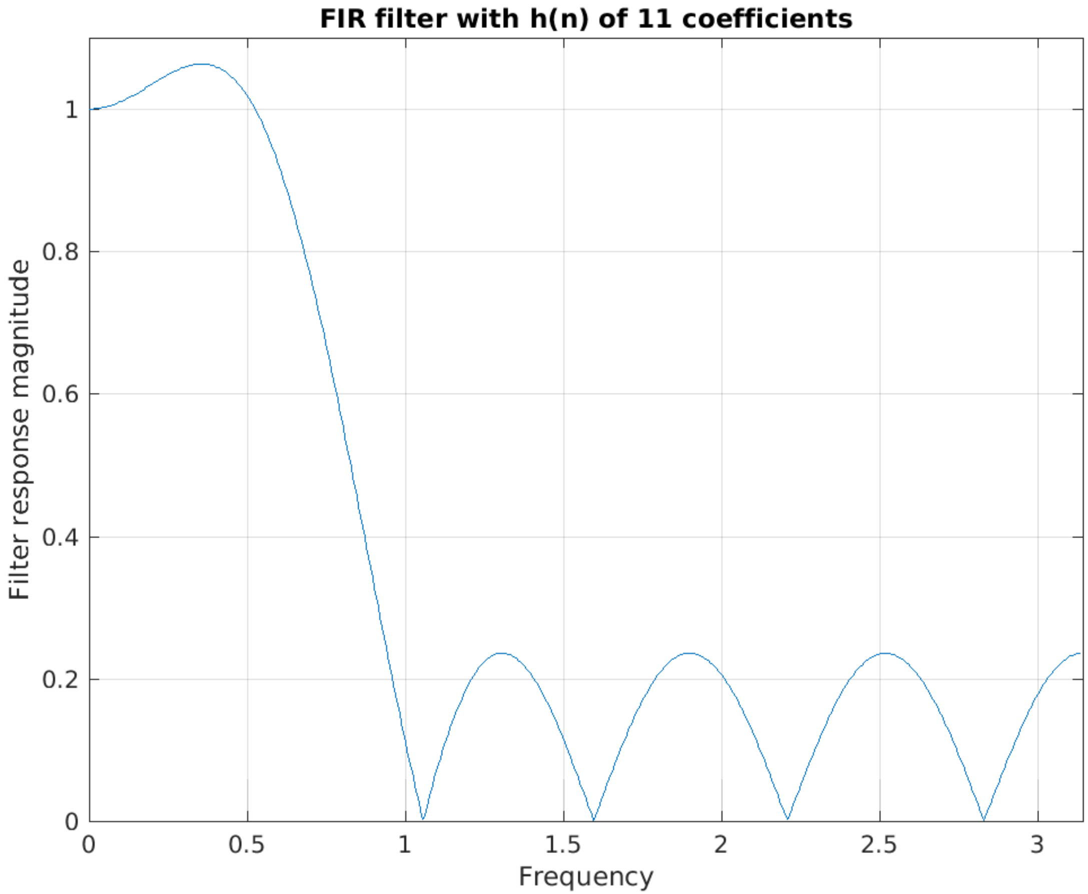
Example Filter

- 7 coeffs.



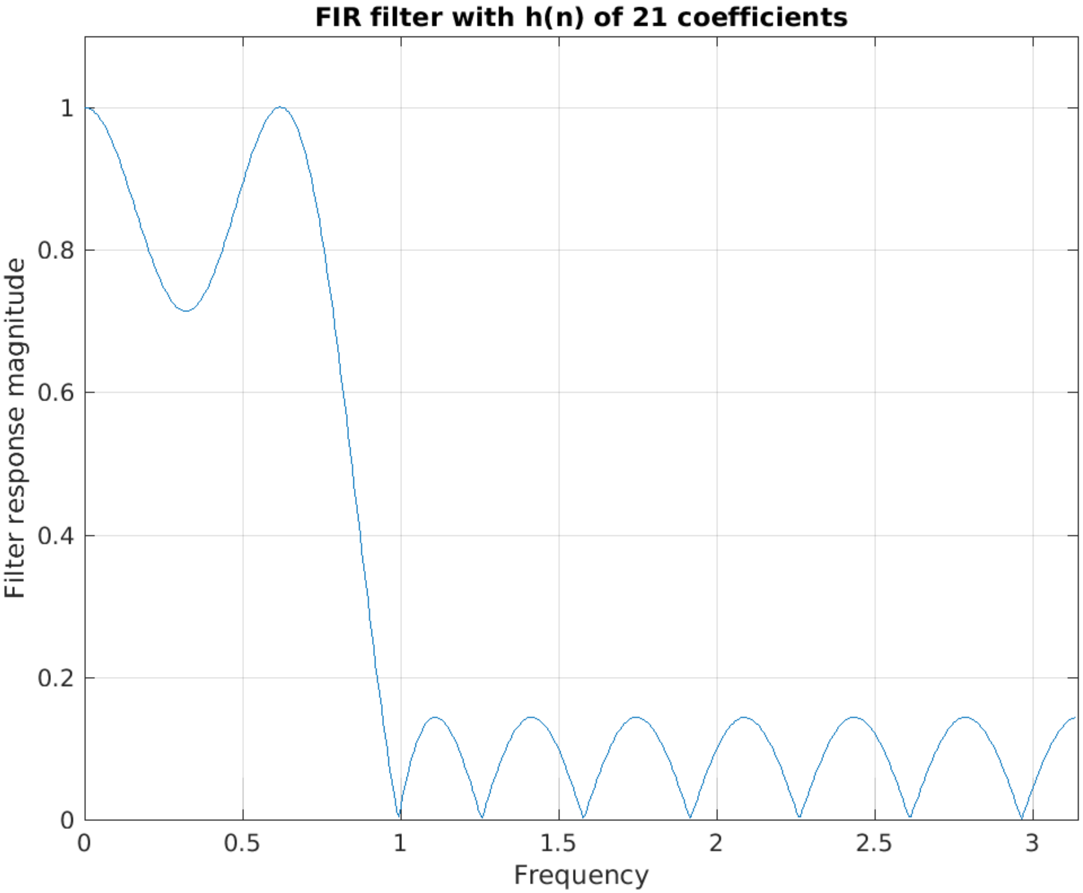
Example Filter

- 11 coeffs.



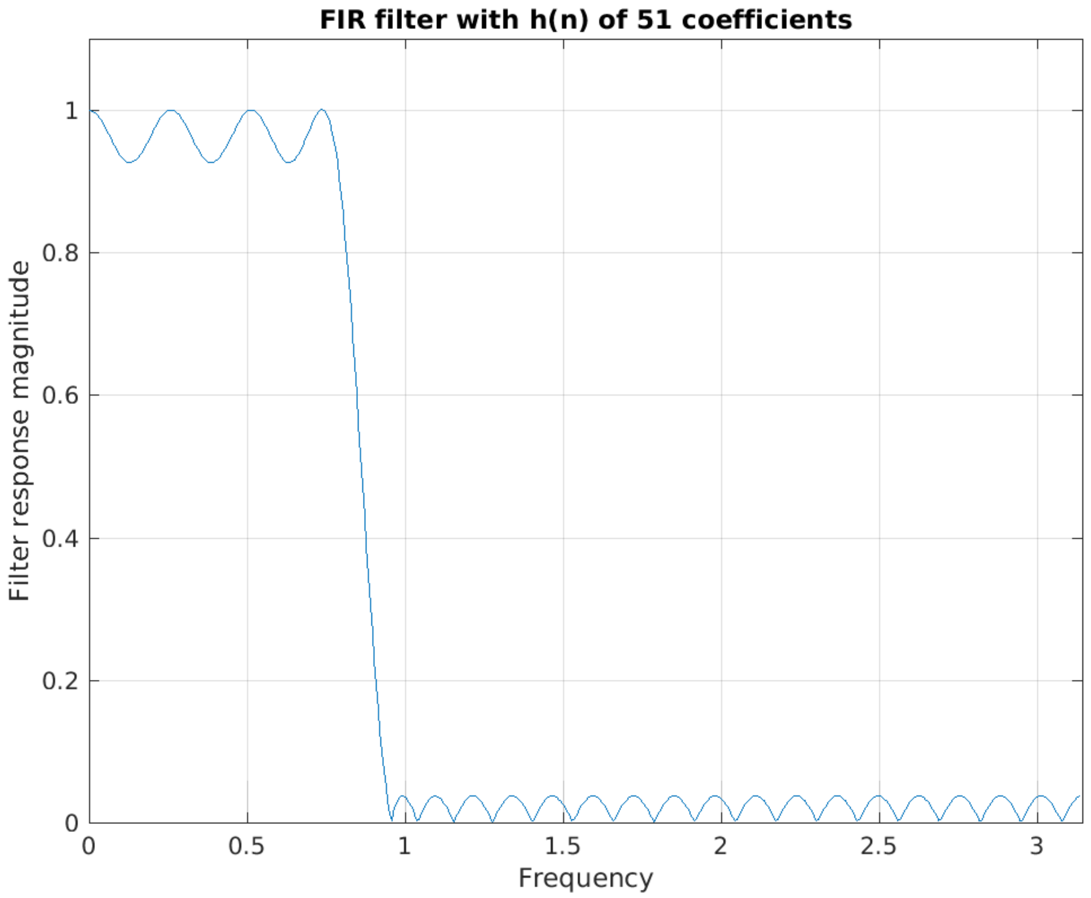
Example Filter

- 21 coeffs.



Example Filter

- 51 coeffs.

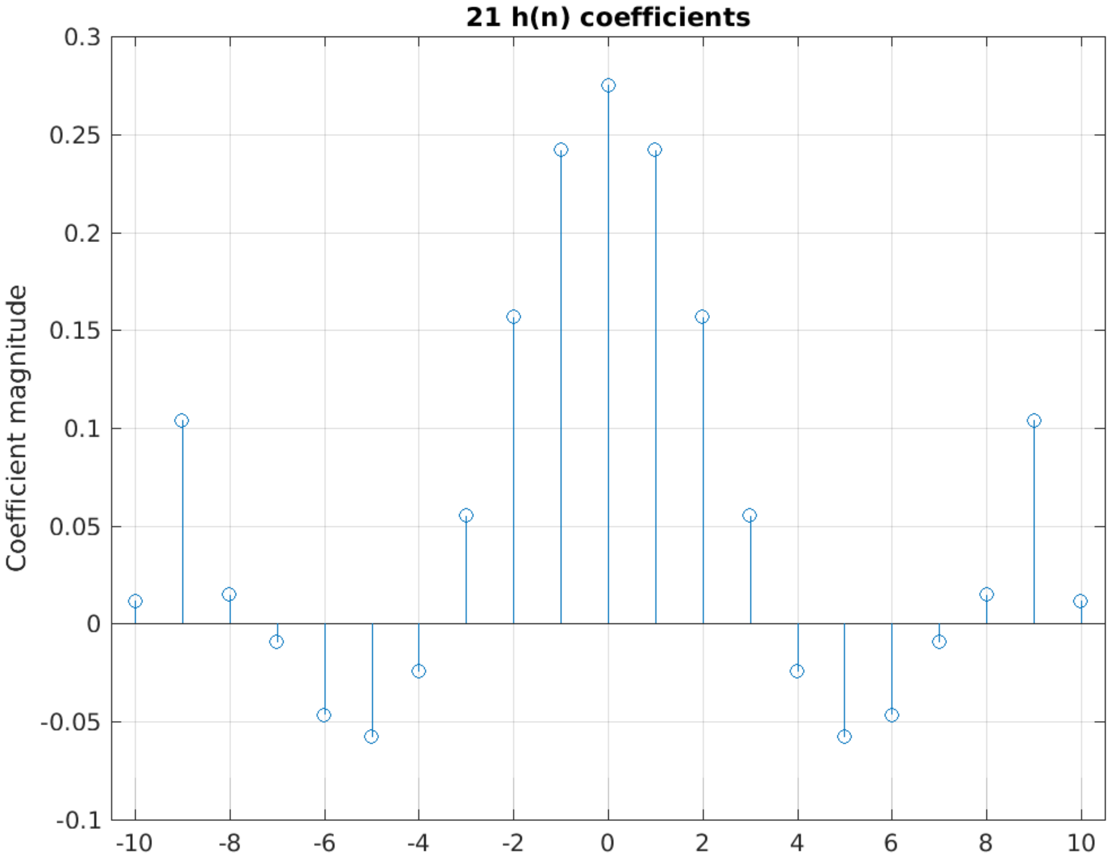


Example 21-tap Filter

- Use the `remez()` function for filter design
  `>> help remez`
  to get more information on the matlab function
- Notice the remez function's first argument is the number of desired taps *minus 1*
- ```
  coeffs = remez(20, [0 0.25 0.30 1],
                     [1  1    0    0]);
  ```
- To plot the coefficients, use
  `stem(-10:10, coeffs);`
  ```

Example Filter Coefficients

- This plot shows the coefficients of 21-tap filter
- This is a low-pass filter which is a rect() in the frequency domain
- The low-pass filter has a sinc() shape in time domain



Matlab for Examples

- This matlab code produced the plots shown in this section
- In these examples, the filter response is clearer on a linear scale, so freqz()'s output was output into the variable "H" (magnitude) and plotted normally rather than using freqz()'s automatic plotting.

```
% exampfilt.m
%
% Develops and plots four low-pass filters of varying lengths with the
% same frequency/amplitude specs.
%
% 2020/02/20  Cleaned up plots, added axis labels and titles, added png plots
% 2005/02/10  Added PrintOn* variables

%----- Initialize
PrintOn         = 1;
spec_f          = [0 0.25 0.3 1];
spec_amp        = [1 1    0   0];
axislimits      = [0 pi 0 1.1];

%----- Main
figure(1); clf;
[H,W]=freqz(remez(6,spec_f,spec_amp));
H = H ./ abs(H(1));
plot(W,abs(H));
axis(axislimits); grid on; title('FIR filter with h(n) of 7 coefficients');
xlabel('Frequency');
ylabel('Filter response magnitude');
if PrintOn  print -dpng  exampfilt1.png;   end

figure(2); clf;
[H,W]=freqz(remez(10,spec_f,spec_amp));
H = H ./ abs(H(1));
plot(W,abs(H));
axis(axislimits); grid on; title('FIR filter with h(n) of 11 coefficients');
xlabel('Frequency');
ylabel('Filter response magnitude');
if PrintOn  print -dpng  exampfilt2.png;   end

figure(3); clf;
[H,W]=freqz(remez(20,spec_f,spec_amp));
H = H ./ abs(H(1));
plot(W,abs(H));
axis(axislimits); grid on; title('FIR filter with h(n) of 21 coefficients');
xlabel('Frequency');
ylabel('Filter response magnitude');
if PrintOn  print -dpng  exampfilt3.png;   end

figure(4); clf;
co=remez(20,spec_f,spec_amp);
stem(-10:10, co);
axis([-10.5 10.5 -0.1 0.3]); grid on; title('21 h(n) coefficients');
ylabel('Coefficient magnitude');
if PrintOn  print -dpng  exampfilt4.png;   end

figure(5); clf;
[H,W]=freqz(remez(50,spec_f,spec_amp));
H = H ./ abs(H(1));
plot(W,abs(H));
axis(axislimits); grid on; title('FIR filter with h(n) of 51 coefficients');
xlabel('Frequency');
ylabel('Filter response magnitude');
if PrintOn  print -dpng  exampfilt5.png;   end
```

Seeing the Frequency Response of Filters

Filter Frequency Response (Method I)

- There are two main methods to see the frequency response of a vector of filter coefficients
- Method 1
  - `freqz()` function in matlab
    - Exact frequency response
    - Very fast

Filter Frequency Response (Method II)

- To see frequency response of a filter (method II)
  1. Make a flat (white) spectrum input signal
  2. Send the signal into the filter and look at the output spectrum
     - Requires many samples for accurate output (not exact)
     - Much slower
     - Sometimes the only way to see spectrum
       - Ex: an arbitrary signal, not a filter response
       - Ex: hardware rounding
       - Ex: signal saturation
     - Example matlab code:

```
in = rand(1, 100000) - 0.5;
out = conv(coeffs, in) + 0.25;     % Hypothetical ¼ LSB bias
abs(fft(out))
psd(out)
spectrum(out)
```

- There are more relevant details in the *Estimating Spectral Magnitude* section