A Matlab Function for FIR Half-Band Filter Design

Half-band filters have -6 dB frequency of 1/4 the sample rate, and odd symmetry of the frequency response about 1/4 the sample rate. Given the odd-symmetry of the response, the passband and stopband edge frequencies are symmetric with respect to fs/4. This symmetry makes the halfband filter ideal for decimation by 2 or interpolation by 2. And, remarkably, the odd-indexed coefficients of FIR half-band filters are zero, except for the main tap coefficient.

FIR Half-band filters are not difficult to design. In an earlier post [1], I showed how to design them using the window method. Here, I provide a short Matlab function `halfband_synth` that uses the Parks-McClellan algorithm (Matlab function `firpm [2])` to synthesize half-band filters. Compared to the window method, this method uses fewer taps to achieve a given performance. The function's code is listed at the end of the article. The function call is as follows:

```
b= halfband_synth(fpass,fs,ntaps)
```

where

fpass = desired passband edge frequency, Hz. fpass must be less than fs/4.
fs = sample rate, Hz.
ntaps = desired number of filter taps

ntaps + 1 must be an integer multiple of 4, so ntaps = 7, 11, 15, …
ntaps = 3 is not allowed. (For ntaps = 3, the coefficients are [1 2 1]/4).

b = vector of filter coefficients. Given indexing [b_0 b_1 b_2 …], the odd-indexed coefficients are zero, except for the main tap.

Following are two of examples of half-band filter synthesis using the function.

Example 1.

Let fs = 1 Hz, fpass = 0.15*fs, and ntaps = 11. The following Matlab code computes the half-band coefficients.

```
fs= 1;
fpass= 0.15*fs;
ntaps = 11;
b= halfband_synth(fpass,fs,ntaps);
```

The function prints fstop to the workspace: `fstop =  0.3500`
and it computes the coefficients:

```
b= [.0163 0 -.0683 0 .3038 .5 .3038 0 -.0683 0 .0163]
```

The filter has (ntaps + 3)/2 = 7 non-zero coefficients. The coefficients are plotted in Figure 1 (top). The frequency response can be computed as follows:

```
[H,f]= freqz(b,1,256,fs);
Hmag = abs(H);
HdB= 20*log10(Hmag);
```

The magnitude of H is plotted in Figure 1 (bottom), showing the odd symmetry with respect to $f_s/4$. Note |H| = 0.5 at $f_s/4$. The dB-magnitude response is plotted in Figure 2, where we see that the response is equiripple in the stopband (top) and passband (bottom), as expected from the Parks-McClellan algorithm.

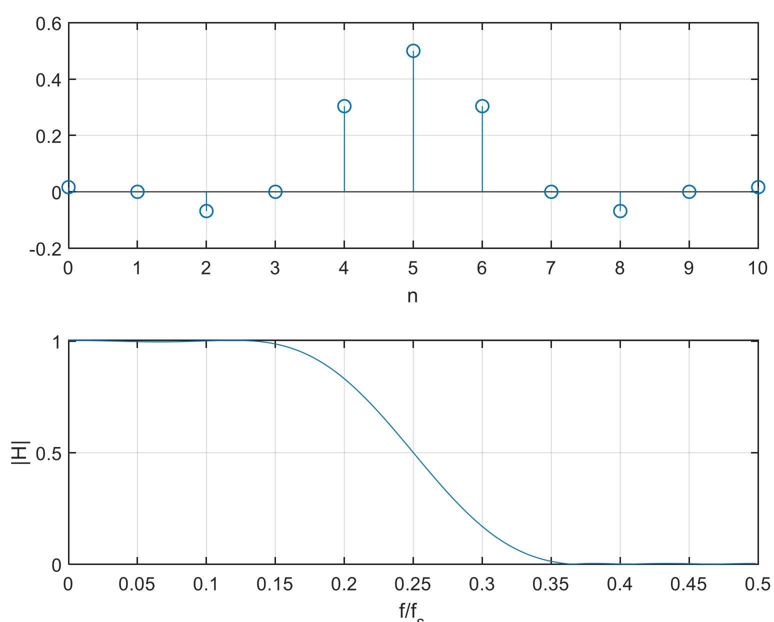

Figure 1. Top: 11-tap half-band filter coefficients. Bottom: Magnitude response.

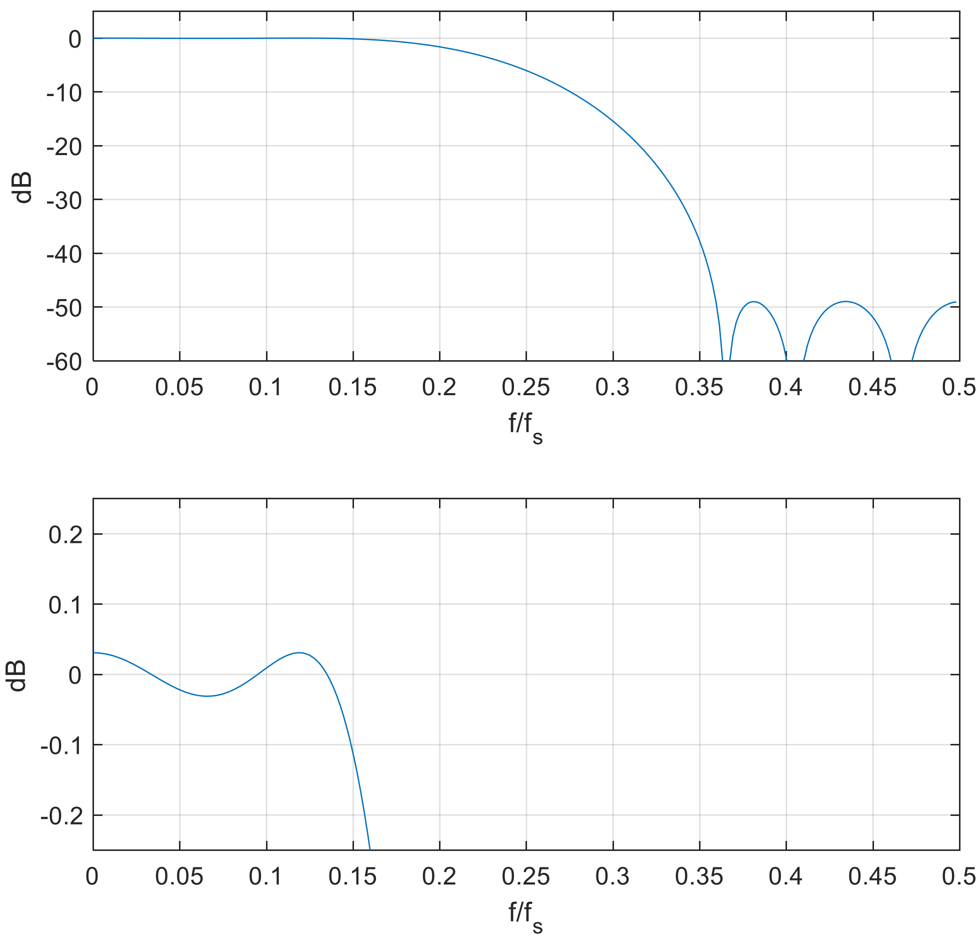

Figure 2. Top: 11-tap half-band filter dB-magnitude response.
Bottom: dB-magnitude response in the passband.

Example 2.

Let fs = 200 Hz, fpass = 40 Hz, and ntaps = 35. The following Matlab code computes the half-band coefficients.

```
fs = 200;
fpass= 40;
ntaps = 35;
b= halfband_synth(fpass,fs,ntaps);
```

The filter has (ntaps + 3)/2 = 19 non-zero coefficients. The coefficients and magnitude response are plotted in Figure 3. The dB-magnitude response is plotted in Figure 4.

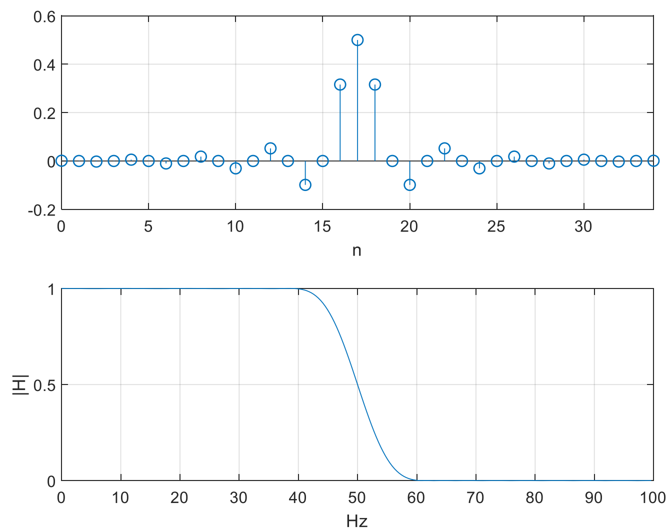

Figure 3. Top: 35-tap half-band filter coefficients. Bottom: Magnitude response.

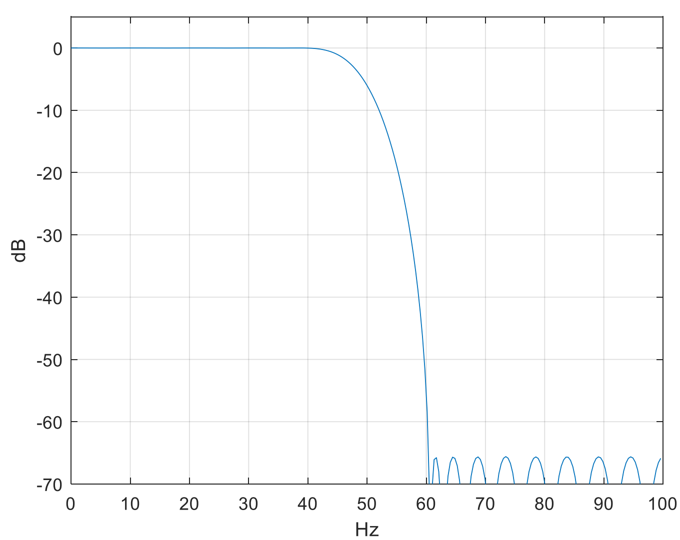

Figure 4. 35-tap half-band filter dB-magnitude response.

How It Works

The simplest method for designing an equiripple half-band FIR filter would be to compute the stopband edge frequency that is symmetrical about fs/4 with the passband edge frequency, and set a goal function and frequencies as follows, where fpass is passband edge frequency and fs is sample frequency:

```
fstop = fs/2 - fpass              % Hz stopband edge frequency
a= [1 1 0 0];                     % amplitude goal function
f= [0 fpass fstop fs/2]/(fs/2);   % normalized frequencies
```

The coefficients would be computed using firpm. The odd-index coefficients from firpm would be close to zero, but not exactly zero. These coefficients would then be set to exactly 0.

Here, we use a more accurate method described by Vaidyanathan and Nguyen [3]. To illustrate the method, we start by considering the half-band filter coefficients from Example 1:

```
b= [.0163 0 -.0683 0 .3038 .5 .3038 0 -.0683 0 .0163]
```

Now define coefficients g:

```
g = 2*b(1:2:end)          (1)

  = 2*[.0163 -.0683 .3038 .3038 -.0683 .0163]
```

The magnitude response |H(f)| of coefficients b is plotted in the top of Figure 5, and the magnitude response |G(f)| of coefficients g is plotted in the bottom. The passband edge of |G| is exactly 2*fpass, where fpass is the passband edge of |H|, and |G| is zero at fs/2. We call g a *one-band* filter.

Now let's reverse this process by first synthesizing the one-band filter g and using it to compute the half-band filter coefficients b. The passband edge is 2*fpass and the stopband is just the single frequency fs/2, so we can specify an ideal |G(f)| as follows:

```
a = [1 1 0 0];                     % amplitude goal function
f = [0 2*fpass fs/2 fs/2]/(fs/2);  % normalized frequencies
```

The desired filter b has ntaps = 11, and the length of g is:

```
gtaps = (ntaps + 1)/2 = 6          (2)
```

Now g can be synthesized using firpm:

```
g = firpm(gtaps-1,f,a);
```

The half-band filter b is realized by inserting zeros between the elements of g/2, which is the inverse of the process shown in Equation 1. Finally, the main tap is set to 0.5. The synthesis process requires only gtaps = (ntaps + 1)/2 coefficients, vs ntaps for the conventional approach. This results in quicker computation and more accurate coefficients.

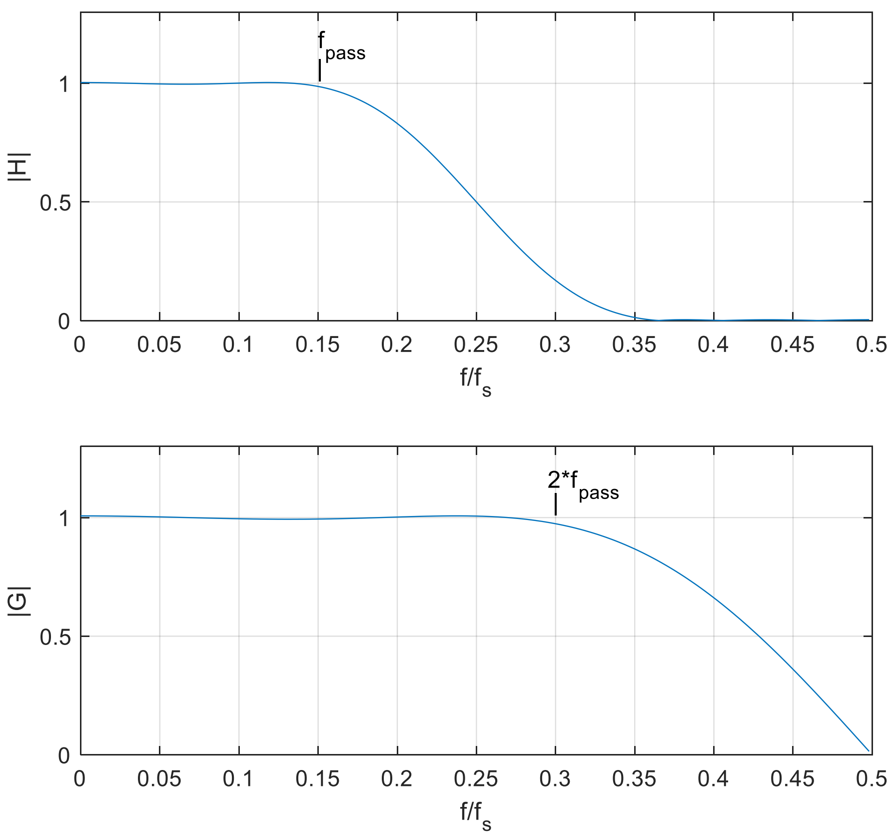

Figure 5. Top: Magnitude response |H| of coefficients b.
Bottom: Magnitude response |G| of coefficients g.

Matlab Function halfband_synth

The code for the function halfband_synth is listed below. The number of taps is called ntaps. ntaps + 1 must be an integer multiple of 4, so ntaps = 7, 11, 15, … It is possible to design a half-band filter with an even number of taps, but this results in a filter with no zero-valued coefficients. The cases ntaps = 9, 13, 17, …, are not used because the end coefficients are zero. Finally, note that Matlab has a function to design half-band filters in the DSP System Toolbox [4].

Disclaimer: I believe this code to be correct, but it may contain errors. Be sure to verify the coefficients before using them in a design.

```
% halfband_synth.m   7/2/25 Neil Robertson
% Half-band filter synthesis function using firpm,
% based on Vaidyanathan and Nguyen [3].
%
% b = halfband_synth(fpass,fs,ntaps)
%   fs = sample rate, Hz.
%   fpass = passband edge frequency, Hz.  fpass < fs/4.
%   ntaps = number of taps. ntaps = 7, 11, 15, ...
%   b = vector of filter coefficients.
%
function b = halfband_synth(fpass,fs,ntaps)

if fpass >= fs/4
    error('fpass must be less than fs/4')
end
if mod((ntaps+1),4) ~= 0
    error('ntaps+1 must be an integer multiple of 4')
end

a = [1 1 0 0];                        % amplitude goal function
f = [0 2*fpass fs/2 fs/2]/(fs/2);     % frequencies for optimization

gtaps = (ntaps + 1)/2;               % one-band filter number of taps
g = firpm(gtaps-1,f,a);               % one-band filter coefficients

b = zeros(1,ntaps);
b(1:2:end) = g/2;                     % half-band coefficients
b(gtaps) = .5;                        % center tap coefficient
fstop = fs/2 - fpass                  % Hz stopband edge frequency
```

References

1. Robertson, Neil, “Simplest Calculation of Half-band Filter Coefficients”, DSPRelated.com, Nov., 2017, https://www.dsprelated.com/showarticle/1113.php

2. Matlab website, “firpm”, https://www.mathworks.com/help/signal/ref/firpm.html

3. Vaidyanathan, P. P., and Nguyen, Truong Q., “A ‘Trick’ for the Design of FIR Half-Band Filters”, IEEE Transactions on Circuits and Systems, VOL CAS-34, No. 3, March 1987. https://authors.library.caltech.edu/records/czbzh-df707

4. Matlab website, “FIR Halfband Filter Design”, https://www.mathworks.com/help/dsp/ug/fir-halfband-filter-design.html

Neil Robertson, July 2025