

Supplement to Chapter 5 of *The Science of Digital Media* – Digital Audio Processing

Mathematical Modeling Exercise – Digital Audio Processing > Creating Your Own FIR Filters ¹

Modeling Environment: MATLAB

Introduction:

One method for creating an FIR filter "from scratch" is called the windowing method. Chapter 5 of *The Science of Digital Media* shows how to derive functions for low-pass, high-pass, bandpass, and bandstop FIR filters beginning with an idealized graph of the desired frequency response. The resulting functions are given in the table below.

Type of filter	$h_{ideal}(n), n \neq 0$	$h_{ideal}(0)$
Low-pass	$\frac{\sin(2\pi f_c n)}{\pi n}$	$2f_c$
High-pass	$-\frac{\sin(2\pi f_c n)}{\pi n}$	$1 - 2f_c$
Bandpass	$f_2 \frac{\sin(2\pi f_2 n)}{\pi n} - f_1 \frac{\sin(2\pi f_1 n)}{\pi n}$	$2(f_2 - f_1)$
Bandstop	$f_1 \frac{\sin(2\pi f_1 n)}{\pi n} - f_2 \frac{\sin(2\pi f_2 n)}{\pi n}$	$1 - 2(f_2 - f_1)$

Table 1 Functions for FIR filters

To smooth the discontinuities at the filter edges, a windowing function must also be applied. Some common windowing functions are given in the following table.

1 Rectangular windowing function	$w(n) = 0.5 + 0.5 \cos\left(\frac{2\pi n}{N}\right)$ Hanning windowing function
$w(n) = 0.54 + 0.46 \cos\left(\frac{2\pi n}{N}\right)$ Hamming windowing function	$w(n) = 0.42 + 0.5 \cos\left(\frac{2\pi n}{N-1}\right) + 0.08 \cos\left(\frac{4\pi n}{N-1}\right)$ Blackman windowing function

Table 2 Windowing functions

The algorithm for creating an FIR filter is given below:

¹This material is based on work supported by the National Science Foundation under Grant No. DUE-0340969. This worksheet was written by Jennifer Burg (burg@wfu.edu).

```

algorithm FIR_low_pass filter
/*Input:  f_c, the cutoff frequency for the lowpass filter, in Hz
          f_samp, the sampling frequency of the audio signal to be filtered,
in Hz
          N, the size of the filter; assume N is odd
Output:  a low-pass FIR filter in the form of an N-element array */
{
  /*Normalize f_c and ω_c so that π is equal to the Nyquist angular
frequency*/
  f_c = f_c/f_samp
  ω_c = 2*π*f_c
  middle = N/2 /*Integer division, dropping remainder*/
  /*Create the filter using the low-pass filter function from Table 1*/
  /*Put a dummy value in for n = 0 to avoid a divide by 0 error*/
  for n = -N/2 to N/2
    if (n = 0) ftr(n+middle) = 1
    else ftr(n+middle) = sin(2*π*f_c*n)/(π*n)
  ftr(middle) = 2*f_c
  /*Multiply the elements of ftr by a windowing function chosen from Table 2.
We use the Hanning window function/
  for n = 0 to N-1
    ftr(n) = ftr(n) * (0.5 + 0.5*cos((2*π*n)/N))
}

```

The filter that you create by the algorithm above is an FIR filter. It is the impulse response, $h(n)$, that is applied in the equation for FIR convolution below.

$$y(n) = h(n) \otimes x(n) = \sum_{k=0}^{N-1} h(k)x(n-k)$$

where $x(n-k) = 0$ if $n-k < 0$

Equation 1

$h(n)$ is just a vector of multipliers – i.e., coefficients – to be applied successively to sample values. The number of coefficients is the order of a filter, N .

Exercise 1

Using the algorithm and functions above, create an order FIR low-pass filter for keeping in all frequencies below 1000 Hz and filtering out all other frequencies. Try your filter on an audio file that has just one frequency greater than your cutoff frequency. See if it is filtered out.

Exercise 2

Try your filter on a short clip of classical music. Listen to the difference between the filtered and unfiltered versions.

Exercise 3

Graph the frequency response of the original audio file from Exercise 2 and compare it to a frequency response of the filtered audio file.

Exercise 4

Graph the filter function.

Exercise 5

Try different windowing functions and compare the results by listening to the audio and looking at the frequency response graphs.