International Journal of Scientific Engineering and Applied Science (IJSEAS) – Volume-2, Issue-2, February 2016
ISSN: 2395-3470
www.ijseas.com

SIGNAL PROCESSING TOOLS FOR DIGITAL FILTER DESIGN

[1]Engr. Dr. Eyenubo O. J., [2]Engr. Oniyemofe C. O., [3]Engr. Tanno K. O. and
[4]Engr. Okonye S.
[1, 3 & 4]Department of Electrical/Electronic Engineering,
[2]Department of Computer engineering
Delta state Polytechnic, Otefe-Oghara, Delta State, Nigeria.

Abstract:

Matlab provides different options for digital filter design, which include function calls to filter algorithms and a graphical user interface called Sptool (Signal Processing tools). A variety of filter design algorithms are available in Matlab for both IIR and FIR filters. This paper discusses the different options in Matlab and gives examples of lowpass, highpass, and bandpass filter designs. Results show that the graphical user interface Sptool is a quicker and simpler option than the option of making function calls to the filter algorithms. Sptool has a more user friendly environment since the spectrum of the filter is immediately displayed to the user, and the user can quickly zoom in and examine particular areas of interest in the spectrum (i.e. the passband). However, the shortcoming of Sptool is that it only displays the magnitude response of the filter, not the phase response.

Keywords—Band-pass filter, Butterworth response, Circuit Maker, MATLAB

1.0 Overview of Matlab

The following is with reference to (MATLAB 8.1). MATLAB (MATrix LABoratory) is a high performance language for technical computing. It integrates computation, visualization and programming in an easy-to-use environment (Mathworks, 1997). The basic data element in MATLAB is an array. Many matrix based functions like matrix multiplication and array dot products can be executed in a fraction of time it would take to write a similar program in a scalar non-interactive language such as C or FORTRAN. MATLAB features a family of application specific solutions called toolboxes for signal processing, neural networks and wavelets to name a few. These toolboxes are a collection of functions written as M-files. The signal processing toolbox (SPtool) for example includes an interactive environment for analyzing and manipulating signals designing filters (Mathworks, 1996). MATLAB also has a number of easy to use plotting and graphical functions, which make MATLAB an attractive choice for developing attractive visualization applications; (Mathworks, 1997) gives an introduction to the use of MATLAB. The vectorized nature of MATLAB, the abundant collection of functions and visualization options makes it a good choice for visualizing DSP concepts. Development of an educational tool for classroom instruction needs a powerful Graphical User Interface (GUI).

Matlab has several design algorithms that can be used to create and analyze both Infinite Impulse Response (IIR) and Finite Impulse Response (FIR) digital filters. The IIR filters that can be created in Matlab are Butterworth, Chebyshev type 1 and 2, and elliptic. The FIR filter algorithms in Matlab are equiripple, least squares, and Kaiser Window. The Matlab code required to implement these filters involves bilinear transformations and function calls to analog prototype filters. The following sections give examples of Matlab implementation of the IIR filters listed above.

1.1 Graphical User Interface (GUI) development using MATLAB

MATLAB provides-GUIDE (GUI Design Environment) to develop GUIs using "drag and drop objects" such as buttons, sliders and pop-down menus to name a few. Impressive GUIs can be developed in a short time. An M-file performing a particular task for each object in the GUI is written separately using many of the in-built MATLAB functions. The GUIDE callback editor manages the actions associated with the selection of a particular object (for e.g. clicking a button) by linking an object to its respective M-file. [10], describes the GUI development in MATLAB.

Once the GUI is developed, one can use the mouse and on-screen options to visualize, hear and manipulate signals (audio and images); we have used a number of signals that can be analyzed for example in audio; we use male and female voices, music and standard waves (sine, chirp, square, triangle). A few demos also use images as inputs to illustrate multidimensional DSP [3, 6] algorithms. All audio samples used by our demo modules are .wav files that have been stored differently for ease of programming. For classroom instruction, the demos are hyperlinked from an MSWord document. MATLAB always runs the Matlab\toolbox\local\matlabrc.m on startup.

1.2 Fourier Series

In this demo, a selected standard signal is decomposed into its constituent sinusoids and the user can view the frequency representation of the signal.

The user also has an option of adding one frequency component at a time to see how these sinusoids add up to form the signal under consideration. This can also be used to illustrate Gibb's phenomenon wherein, a square wave is selected as the input. These vectors are plotted on the same figure to show the evolution of the square wave.

1.3 Square Wave from Sine Waves

The Fourier series expansion for a square-wave is made up of a sum of odd harmonics. We show this graphically using MATLAB.

We start by forming a time vector running from 0 to 10 in steps of 0.1, and take the sine of all the points.

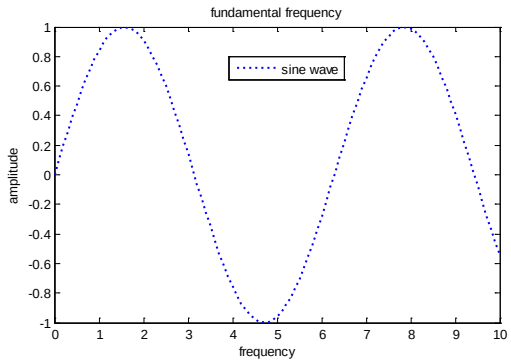

Figure 1

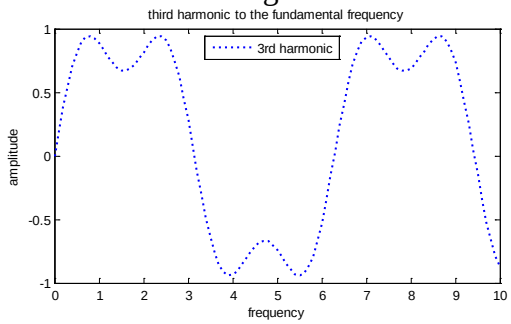

Figure 2

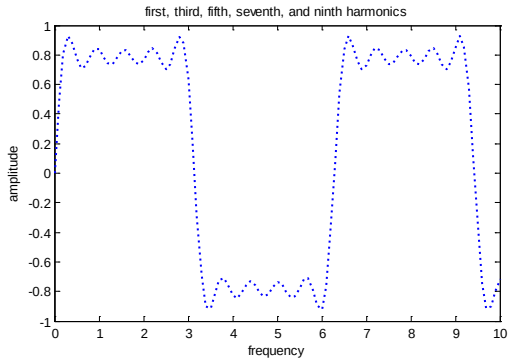

Figure 3

For a finale, we will go from the fundamental to the 9th harmonic, creating vectors of successively more harmonics, [1, 2] and saving all intermediate steps as the rows of a matrix. These vectors are plotted on the same figure to show the evolution of the square wave.

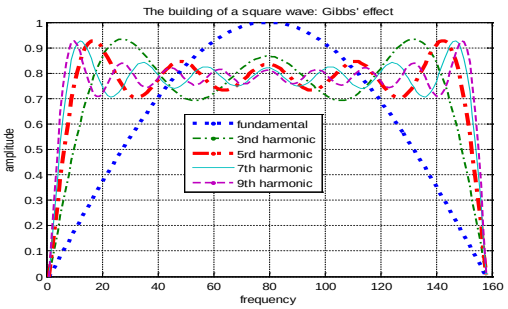

Figure 4

2.0 Filter design

FIR filters - The effects of various windows for the windowing technique of FIR filter design is illustrated. The user can vary the window length and examine the frequency response of these windows. A GUI for FIR filter design using windows allows the user to specify the window to be used, the filter order and other frequency specs for filter design. The designed filter can be applied to an input signal to hear the effect of the filter on the audio sample. The magnitude and phase response of the filter can be viewed too. We have similar demos for FIR filter design using frequency sampling and Remez exchange algorithm. IIR filters - The GUI for IIR filters is similar to the one used for the FIR case except that the design is now using analog prototypes such as Butterworth, Chebyshev and Elliptical filters. The bilinear transformation is used for converting the analog filters to the digital domain. The minimum filter order is obtained using the "Auto" design mode. The pole zero plot can be also be viewed for the designed filter. Besides the examples discussed above, we also have developed a number of other demos to explain concepts in Discrete Fourier transforms, multirate DSP, short time Fourier transforms etc. A median filter demo (for audio and images) is also developed to illustrate the superior performance of non-linear filters over linear systems.

2.1 Filter classification

An ideal frequency-selective filter is a system that passes a pre-specified range of frequency components without any attenuation but completely rejects the remaining frequency components.

The range of input frequencies that is left unaffected by the filter is referred to as the pass band of the filter, while the range of input frequencies that are blocked from the output is referred to as the stop band of the filter. In terms of the magnitude spectrum, the absolute value of the transfer function $|H(\omega)|$ of the frequency filter, therefore, toggles between the values of A and zero as a function of frequency ω. The gain $|H(\omega)|$ is A, typically set to one, within the pass band, while $H(\omega)|$ is zero within the stop band. Depending upon the range of frequencies within the pass and stop bands, an ideal frequency-selective filter is categorized in four different categories. These categories are defined in the following.

2.2 Lowpass filters

The transfer function $H_{lp}(\omega)$ of an ideal lowpass filter is defined as follows:

$$H_{lp}(\omega) = \begin{cases} A & |\omega| \leq \omega_c \\ 0 & |\omega| > \omega_c \end{cases} \qquad 2.1$$

where ω_c is referred to as the cut-off frequency of the filter. The pass band of the lowpass filter is given by $|\omega| \leq \omega_c$ while the stop band of the lowpass filter is given by $\omega_c < |\omega| < \infty$.

2.3 Highpass filters

The transfer function $H_{hp}(\omega)$ of an ideal highpass filter is defined as follows:

$$H_{hp}(\omega) = \begin{cases} 0 & |\omega| \leq \omega_c \\ A & |\omega| > \omega_c \end{cases} \qquad 2.2$$

where ω_c is the cut-off frequency of the filter. In other words, the transfer function of an ideal highpass filter $H_{hp}(\omega)$ is related to the transfer function of an ideal lowpass filter $H_{lp}(\omega)$ by the following relationship:

$$H_{hp}(\omega) = A - H_{lp}(\omega). \qquad 2.3$$

The pass band of the lowpass filter is given by $\omega_c < |\omega| < \infty$, while the stop band of the lowpass filter is given by $|\omega| \leq \omega_c$.

2.4 Bandpass filters

The transfer function $H_{bp}(\omega)$ of an ideal bandpass filter is defined as follows:

$$H_{bp}(\omega) = \begin{cases} A & \omega_{c1} \leq |\omega| \leq \omega_{c2} \\ 0 & \omega_{c1} < |\omega| \text{ and } \omega_{c2} < |\omega| < \infty \end{cases} \qquad 2.4$$

where ω_{c1} and ω_{c2} are collectively referred to as the cut-off frequencies of the ideal bandpass filter. The lower frequency ω_{c1} is referred to as the lower cut off, while the higher frequency ω_{c2} is referred to as the higher cut off. Unlike the highpass filter, the bandpass filter has a finite bandwidth as it only allows a range of frequencies ($\omega_{c1} \leq \omega \leq \omega_{c2}$) to be passed through the filter.

2.5 Bandstop filters

The transfer function $H_{bs}(\omega)$ of an ideal bandstop filter is defined as follows:

$$H_{bs}(\omega) = \begin{cases} 0 & \omega_{c1} \leq |\omega| \leq \omega_{c2} \\ A & \omega_{c1} < |\omega| \text{ and } \omega_{c2} < |\omega| < \infty \end{cases} \qquad 2.5$$

where ω_{c1} and ω_{c2} are, respectively, referred to as the lower cut-off and higher cut-off frequencies of the ideal bandstop filter. A bandstop filter can be implemented from a bandpass filter using the following relationship.

$$H_{bs}(\omega) = A - H_{bp}(\omega). \qquad 2.6$$

The ideal bandstop filter is the converse of the ideal bandpass filter as it eliminates a certain range of frequencies ($\omega_{c1} \leq \omega \leq \omega_{c2}$) from the input signal.

2.6 Lowpass Filter Design

Using Matlab, a lowpass digital filter is designed using various analog prototypes: Chebyshev, Butterworth, and Elliptic. The optimum filter type is chosen on the basis of implementation complexity, magnitude response, and phase response.

2.6.1 Matlab Code (Butterworth):

```
% Lowpass digital filter with Butterworth -I analog prototype
%
% Digital Filter Specifications:
wp = 0.125*2*pi; % digital passband frequency in Hz (normalized)
ws = 0.1375*2*pi; % digital stopband frequency in Hz (normalized)
Rp = 0.5; % passband ripple in dB
As = 20; % stopband attenuation in dB
% Analog Prototype Specifications:
Fs = 1; T = 1/Fs;
OmegaP = (2/T)*tan(wp/2); % prewarp prototype passband frequency
OmegaS = (2/T)*tan(ws/2); % prewarp prototype stopband frequency
% Analog Chebyshev-1 Prototype Filter Calculation:
[c, d] = chb1(OmegaP, OmegaS, Rp, As);
% Bilinear Transformation:
[b, a] = bilinear(cs, ds, Fs);
%
[db,mag,pha,grd,w] = freqz(b,a);
```

```
plot(w*8000/2/pi,db);
xlabel('frequency                (Hz)');
ylabel('decibels'); title('Magnitude
in dB');
```

This exact code is also used for the elliptic and Butterworth designs. The only change is in the filter calculations of each type. Instead of calling **chb1**(), the elliptic filter design calls a function "**elliptic**()" and the Butterworth design calls a function "**butterworth**()".

The following figures show the magnitude and phase responses of each type of filter.

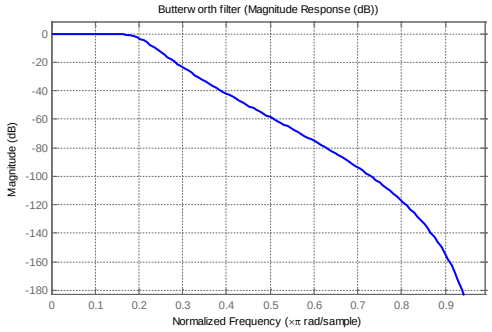

Figure 5 Butterworth filter (Magnitude Response)

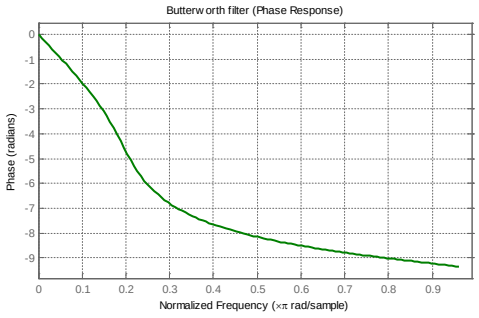

Figure 6 Butterworth filter (Phase Response)

The Matlab code outputs the filter order and the filter coefficients. For this example, the Chebyshev filter order was nine. The elliptic filter had an order of five, and the Butterworth filter order was thirty-two.

Several conclusions can be drawn about these low-pass filter designs from this simple example. First, in general, for a given set of design constraints, the elliptic filter design algorithm will result in the simplest filter (in terms of complexity). The most complex filter is the Butterworth filter with an order of thirty-two. In terms of passband ripple, the [3, 5] Butterworth filter gives the optimum response. In the passband, there is almost no ripple (monotonic). The elliptic and Chebyshev filters both have much more ripple in the passband. So, there is a tradeoff between these three different types of filters. In terms of magnitude response and complexity, the elliptic ripple is most likely the optimum choice. However, the elliptic ripple has a phase response that is more nonlinear than the Chebyshev and Butterworth filters. Therefore, if a sharp cutoff and relatively low complexity is required, the choice would be the elliptic filter. If the phase response would need to be more linear, a Chebyshev or Butterworth filter should be chosen over the elliptic filter.

2.7 Highpass and Bandpass Filter Design

Matlab provides functions for implementing lowpass-to-highpass and lowpass-to-bandpass conversions. By providing a filter order, the passband ripple, and the 3dB cutoff [1] frequency to the function **cheby1**(), a highpass filter can be designed. The filter order is found using the function **chebord** (). For a Butterworth prototype, the functions are **butter**() and **buttord**(). For the elliptic prototype, the functions are **ellip**() and **Ellipord**().

The following Matlab code is used to design a Chebyshev bandpass digital filter with a passband at 1100Hz and a 100Hz transition band.

```
% Bandpass Chebyshev Digital Filter
```

```
ws = 0.125*2*pi; % digital stopband
frequency in rad/s
wp = 0.1375*2*pi; % digital passband
frequency in rad/s
Rp = 0.5; % passband ripple in dB
As = 20;
[N,wn]                                  =
cheb1ord(wp/pi,ws/pi,Rp,As);
[b,a] = cheby1(N, Rp, wn, 'high');
[db,mag,pha,grd,w] = freqz_m(b,a);
plot(w*8000/2/pi,db);
axis([800 1200 -22 1]);
```

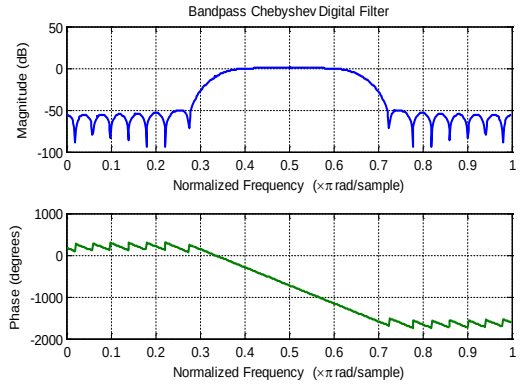

Figure 7 Chebyshev filter
(Magnitude & Phase Response)

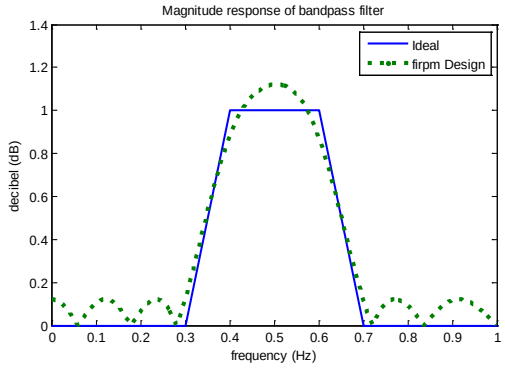

Figure 8 Bandpass filter

3.0 Sptool

Matlab has a very useful visualization tool for designing and analyzing digital filters called
Signal Processing Tool, or Sptool. Sptool is a graphical user interface capable of analyzing and manipulating signals, filters, and spectra. For filter design, Sptool allows the user to select the filter design algorithm to use when creating the filter. The design algorithms included in Sptool are FIR filters (equiripple, least squares, Kaiser Window) and IIR filters (Butterworth, Chebyshev type 1 and 2, elliptic). The user is also allowed to specify the type of filter (lowpass, bandpass, highpass, or bandstop). Sptool designs the filter and displays the magnitude response and the filter order. [2, 7]

The figures below show actual Sptool screenshots for a lowpass filter design using the specifications given above. The Chebyshev Type 1 algorithm was used for these screenshots. By using the zoom options, different parts of the spectrum can be analyzed quickly and easily with Sptool. The second screenshot is a windowed view of the passband of the spectrum contained in the first screenshot.

4.0 Conclusion

In this paper, the development of MATLAB based demos for the visualization of DSP concepts. Currently the demos have to be downloaded and therefore need a local copy of MATLAB. MATLAB server can be used to run the demos on a server and the results can be displayed at the client end. This would obviate the need for a local copy of MATLAB at the client end. More information on the MATLAB server can be got from http://www.mathworks.com/products.

References

1. Hanselman, Duane, and Littlefield, Bruce, 1998, Mastering Matlab 5. Prentice Hall. Upper
Saddle River, NJ.

2. Ingle, Vinay K. and Proakis, John G. 1997, Digital Signal Processing Using Matlab. PWS
Publishing Company.

3. Proakis, John G. and Manolakis, Dimitris G. 1996, Digital Signal Processing: Principles,
Algorithms, and Applications, 3rd Edition. Prentice Hall. Upper Saddle River, NJ,

4. Ziemer, Rodger E., Tranter, William H., and Fannin, D. Ronald. 1993, Signals and Systems:
Continuous and Discrete, 3rd Edition. Macmillan Publishing Company,

5. The MathWorks, Inc., Natick, MA, MATLAB*:* 1996, "The Language of Technical Computing".

6. A. Ambardar and C. Borghesani, Mastering DSP Concepts Using MATLAB. Prentice-Hall, 1998.

7. R. G. Jacquot, J. C. Hamann, J. W. Pierre, and R. F. Kubichek, 1997, "Teaching digital filter design using symbolic and numeric features of MATLAB," *ASEE* Computers in Education, vol. VII, pp. 8–11.

8. "Getting started with MATLABVersion 5," The Mathworks Inc May 1997

9. "MATLAB Signal Processing Toolbox," The Mathworks Inc Dec 1996

10. "Building GUIs with MATLABVersion 5," The Mathworks Inc June 1997

Eyenubo O. J. had his first degree in Electrical/Electronic (P/M) from University of Benin, second degree in Electrical/Electronic (P/M) from University of Benin. He has authored other Journals as Mitigation of harmonics in Power Quality; An Influence of Supply Voltage Frequency on Dynamics of Single-Phase Capacitor Induction Motor; A Study of Noise in Power Generators; A Study of Harmonics in Power Generators; Stator Ground Fault Protection; Engineering Learning and Assessment: Current and Emerging Trend; International Journal for Harmonics Removal in Power Generators Systems Using Filters; The Problems Challenges and Solutions to Insecurity in Nigeria and bagged his Ph.D in Power Quality Analysis: The Reduction of Harmonics Contents in Power Systems Using Electronic Filter in 2015 from University of Benin. He is a member of the Nigerian Society of Engineers also registered with The Council for the Regulation of Engineering in Nigeria.