Analog Filter Designer

version 1.3

User Guide

James C. Squire
Professor
Virginia Military Institute
squirejc@vmi.edu
http://www.jimsquire.com

The Analog Filter Design toolbox (called AFD from here on) bundles a suite of tools for analog filter design and analysis under a single GUI window. Given a type of filter and desired cutoff frequency it can calculate pole and zero placements, display time and frequency-domain system responses, calculate needed resistor and capacitor values for various active-circuit implementations, display user-supplied ASCII waveforms before and after filtering in the time and frequency domains, and complete several other useful functions.

AFD was originally written because some of Matlab's early functions for analog filter design did not work correctly. Many of Matlab's m files worked in the state-space domain instead of the frequency domain, and its design methods introduced extraneous poles and zeros when transformed between domains. Although Matlab corrected its faulty code years ago, AFD remains popular because it incorporates features such as analog active circuit design and filter order comparisons that are either awkwardly implemented or lacking in Matlab and its toolboxes.

Table of Contents

1. Getting Started

AFD runs under Matlab version 6.5 or higher, using either the professional or student editions. Matlab is available from http://www.mathworks.com, and the student edition from any university bookstore.

Download the current version of the AFD toolbox from http://www.jimsquire.com under "research". The toolbox is saved as the single compressed file afd.zip. It does not have subdirectories and can be unzipped anywhere; I suggest creating a folder for it called "AFD" that resides off the main toolbox directory (frequently c:\Matlab\toolbox).

After the AFD files are installed, start Matlab and put the toolbox on Matlab's path. To do this, from Matlab's menu select "File" then "Set Path". Click the "Add Folder" button, then browse to the AFD folder location. Select it, choose "OK", and then choose "Save" from the bottom of the Set Path dialog box.

To run, type "afd" at the Matlab prompt. If Matlab reports "Undefined function or variable" then the AFD toolbox is not on the Matlab path. Typing "afd" will start the "Specify Filter" window; all AFD functionality is obtained using this screen and its menus. As you select design and analysis functions from its menu, the Specify Filter window will open a corresponding new window and place itself in the background. The Specify Filter screen is the only window that has a menu structure for opening analysis windows; therefore the moment the Specify Filter window is closed, the AFD system effectively closes as well (since there is then no way to access other functions). Think of it this way: the Specify Filter window holds the definition for the current filter; it must be kept open at all times. The other analysis windows access the filter information from the Specify Filter window, and they may be open or closed at will.

2. Filter Primer

AFD will implement each of the five most common analog filter types: Bessel, Butterworth, Chebychev I, Chebychev II, and elliptic filters, each which may be designed as a lowpass or highpass filter. Bandpass and bandstop filters may be designed by concatenating low and highpass filters since the hardware implementation choices have very high input impedances and low output impedances.

The behavior of each filter type can be understood in comparison to the behavior of an ideal zero-phase filter that has a gain magnitude of exactly 1 in the passband (e.g. for a lowpass filter the passband consists of all frequencies lower than the critical frequency), a gain of exactly zero in the stopband, and a phase angle of zero everywhere (i.e. a sinusoid in will yield a sinusoid out at the same phase). This ideal filter is not realizable, either in a physical circuit, or in simulation. Although each of the five real filter types will have steeper cutoffs (i.e. a sharper transition between the passband and stopband) as the filter order (and therefore complexity of circuit implementation) increase, all have non-zero passband phase shifts, even in the limit as filter order approaches infinity. Stopband phase characteristics are in general unimportant, as the signal should be greatly attenuated in the stopband.

The next closest to an ideal zero phase filter is an ideal linear phase filter. "Linear phase" refers to a filter that introduces a proportionally increasing phase shift with frequency. Mathematically, the derivative of the phase angle with respect to frequency, $d\theta/df$, yields the group delay, i.e. the apparent delay in time of a sinusoid at the output with respect to the sinusoid at the filter's input. A filter that introduces a phase offset that increases proportionally with frequency everywhere in the passband means sinusoids of all frequencies in the passband will appear at the output with the same time delay; thus an input signal composed entirely of frequencies within the passband will appear unchanged (except for a delay) at the filter output. Thus, the morphology (shape) of the signal is unchanged in an ideal linear-phase filter. It is important to understand that even if the filter is ideal (in the sense that it has a gain magnitude of 1 in the passband and a gain of 0 in the stopband), if it has a non-linear phase, then a signal that lies in the passband will appear quite "garbled" after filtering. The energy within each frequency component will be unchanged in this nonlinear

phase idea filter, but each frequency will appear phase-shifted relative to the others. Speech-processing filters, therefore, should be close to linear-phase to maintain intelligibility, but a touch-tone decoding filter need not be since it just measures whether a sinusoid of a particular frequency present or not and phase is unimportant.

Having a gain magnitude of one and linear phase are both frequency-domain characteristics of ideal filters. Ideal filters may also be thought of as having desirable time-domain characteristics, such as a lack of oscillations in response to a step input.

All "ideal" filters (i.e. a gain magnitude of one in the passband and a gain of zero in the stopband) have infinite order (the order of the filter corresponds to the number of poles in the transfer function, which is another way of saying the highest derivative of the output signal in the differential equation that describes the filter's operation) and thus unfortunately require an infinite-complexity circuit to implement. All finite order filters of any type composed of N poles and no zeros are types of lowpass filters and for a high enough frequency all will exhibit a gain magnitude vs. frequency curve that drops at -20N dB per decade of frequency. Similarly, placing N additional zeros at the origin will transform the lowpass filter into a highpass filter that for low enough frequencies will exhibit a +20N db per decade frequency increase in gain magnitude. What makes each of the five basic types of filters unique is that their placement of poles (and for some, zeros) provides unique tradeoffs between the desired characteristics of having linear phase, fast transitions between pass and stopbands, and lack of ringing in the time domain step response.

Although no filter that can be built as a circuit can have perfectly linear phase in the passband, the Bessel (sometimes called Thompson) filters have the closest approximations. These filters appear to be perfect time-delay elements to passband frequencies as the filter order approaches infinity. Unfortunately, these filters also have the longest (in frequency) transitions between stopband and passband frequencies for a given order among all other filter types. Medical ECG machines commonly use Bessel filters to remove coupled 60Hz noise because the Bessel filter will not change the ECG shape. Bessel filters do not oscillate ("ring") in response to a step input, and lowpass filters have no zeros (highpass zeros are all on the origin).

The Butterworth filter is a very common filter, with slightly worse non-linear phase characteristics than the Bessel filter in return for a slightly sharper transition between passband and stopband frequencies. Like the Bessel filter it is monotonic in its passband, meaning that in the transition from passband to stopband (e.g. low to high frequencies for the lowpass filter) the magnitude of the gain steady decreases with no ripple., and lowpass filters have no zeros, and all zeros in the highpass configuration reside at the origin. Unlike the Bessel filter, the Butterworth filter exhibits slight ringing in its step response.

Chebyshev I filters are not monotonic and have ripple in their passband. The amount of ripple (called G_p in the AFD window) can be chosen during design; the greater the undesirable ripple, the sharper the transition (for a given filter order) between pass and stopbands. These filters typically have steeper transitions between pass and stopbands than the Bessel or Butterworth filters. Like the Bessel and Butterworth filters, they have either no zeros (for the lowpass case) or have all zeros located at the origin (for highpass filters). A typical value of G_p is 3dB. Step responses exhibit considerable ringing.

Chebyshev II filters have ripple in their stopband. They have off-origin zeros in both low and highpass filters. While stopband ripple appears more desirable than passband ripple (since ripple only occurs in frequencies that are mostly attenuated), the off-origin zeros make this type of filter more difficult to implement in circuits. The steepness between pass and stopbands are similar to that of the Chebyshev I filters. Chebyshev II filters let the designer specify how great the attenuation should be between the pass and stopbands (40 dB is typical); for a given filter order, the greater the attenuation (desirable), the greater the width between the pass and stopbands. AFD calls the stopband attenuation G_s. The ringing in the step response of these filters is considerable.

The elliptic filter has ripple in both its passband and stopbands. It is specified by two parameters: passband ripple (G_p) and the minimum attenuation of frequencies in the stopband (G_s). The greater the passband ripple and the less the minimum attenuation of stopband frequencies, the steeper the transition between pass and stopbands. These types of filters in general have by far the steepest transitions between pass and

stopbands of the five basic filter types, but are difficult to build because, like the Chebyshev II filters, both lowpass and highpass configurations have off-origin zeros that require more complex circuits to implement. Elliptic filters also have pole and zero locations very close to each other along the jω axis; slight variations in component values can cause these filters to become unstable (should a pole wander into the right-half plane) or significantly less efficient, and thus also require very precise component tolerances. These filters also exhibit the highest levels of ringing in their step response of the five filter types.

Note that the units for frequencies in AFD are always Hz, as opposed to the usual Matlab standard of rads/sec. AFD restricts the order of filters to $1 \leq n \leq 8$ since the circuit implementation of filters of orders greater than 8 becomes extremely difficult because of exquisite sensitivities to component values.

3. Analog Filter Designer's Windows

3.1 Specify Filter

The Specify Filter window is the main window in AFD. All analysis and design windows are accessed using the menu system from this primary window. Closing this window is therefore tantamount to closing the entire AFD system. The analysis and design windows in contrast can be opened and closed independently. Each of the analysis and design windows are explained in subsequent sections of this document.

Filters can be saved and loaded from the File menu. Once saved, the file name of the filter will appear in the window title bar. Currently only the information selected in the Specify Filter window is saved. If a circuit selection and capacitor/resistor values are chosen in the Build Circuit window, these values are not saved.

The purpose, type, and filter order popup menus are self-explanatory. F_c refers to the critical frequency in Hz, which is the frequency at which the filter attenuates the signal about -3dB (exactly, $2^{-1/2}$) below the passband frequency gain. The passband frequency gain is by default 1, but may be set to any positive value using the passband gain text box.

For the filter types that have ripple in the passband or stopband (the Chebychev I, II, and elliptic filters) the number of ripples changes with filter order. This means that the DC gain of lowpass elliptic filter, for instance, may not be 1 even if the passband gain is set to 1, because for even order filters the passband ripple makes the DC gain equal to the passband gain minus the passband ripple. Consult the picture that appears as each type and order of filter is selected; although the picture is not to scale and does not necessarily draw the correct number of ripples that occur, it does show where the ripple is located, what the DC and high frequency gains are, and how critical frequency is defined for the filter type and odd/even order chosen (i.e. it is sensitive to whether the order is odd or even, but not otherwise to the order number).

See the Filter Primer reference section above for an explanation of G_p (passband ripple in dB) and G_s (stopband attenuation in dB).

3.2 Specify (Order Required)

The Specify Filter Order Required window is the only window in AFD that does not use the data entered from the main Specify Filter screen. Instead it permits a quick comparison among the five filter types to determine the order of each needed to meet a user-specified attenuation in dB at a user-specified width between the pass and stopbands. It assumes the passband gain magnitude is 1 (= 0 dB); therefore the maximum stopband gain that it will permit specified is -3 dB.

Any orders required that are greater than 15 are simply listed as ">15" since it is rarely practicable to build a circuit that can implement filter orders greater than 8, let alone 15.

3.3 Analyze – List Transfer Function

List Transfer Function displays the transfer function H(s) required to implement the filter specified in the main Specify Filter window. It can be displayed in either polynomial or factored (zero, pole, gain) form. In this version, only the transfer function using ideal components is displayed; selecting a particular circuit implementation and choosing standard-valued resistors and capacitors in the Build Circuit screen will result in a slightly different transfer function.

3.4 Analyze – Plot Poles And Zeros

This window plots the zeros and poles of the filter selected in the main Specify Filter window. Poles and zeros may lie on top of each other, and thus obscure the fact that several are co-located. Repeated zeros at the origin, for example, are guaranteed to happen for highpass Bessel, Butterworth, or Chebyshev I filters of order greater than 1. The List Transfer Function window above can clear up any ambiguity.

If a circuit topology and resistors and capacitors have already been selected for the circuit in the Build Circuit window then two sets of poles and zeros will be plotted: black for the ideal pole/zero locations, and blue for the pole/zero locations as implemented using the standardized value choices of the resistors and capacitors.

3.5 Analyze – Time Response

The filter response to an impulse, step, ramp, or parabola are all handled by the Time Response window. These are computed using a simple trick: only code to compute the impulse response was created. The step response is calculated by determining the impulse response of a filter with one additional pole at the origin, the ramp response with two additional poles at the origin, and the parabola with three additional poles at the origin…the additional poles form successive integrations of the impulse response.

Several different metrics are also displayed to enable comparisons among different filter behavior; specifically, the peak value, settling time, and final values are graphically displayed and listed in a table. There are some subtleties to the definitions of these metrics which can be revealed by pausing the mouse over the metric name (e.g. over the word "peak") for a popup "tooltip" description.

The peak time is computed only over the region of time that is currently plotted, which is not necessarily the peak over all time. If the time limits are computed using their default values, the plotted region will nearly always contain the true global peak (assuming one exists…most ramp responses, for instance, grow without bound).

The settling time is computed at the point when the output reaches within the (maximum value on the screen – minimum value on the screen) x (selected percentage) of the true global final value. The t and y values are only listed if this occurs within the plotted time region.

The final value is the true final value, regardless of the time region plotted.

If a circuit topology and resistors and capacitors have already been selected for the circuit in the Build Circuit screen then the "use actual R/C values" option will be enabled, and permit plotting of the actual circuit response in addition to the ideal circuit response.

3.6. Analyze – List Frequency Response

The List Frequency Response window allows the user to compare filter performance (i.e. displays the complex filter gain as a gain magnitude and phase) at two different user-provided frequencies. The gain

magnitude can be displayed either in linear (standard) form or in dB. The phase is computed in both radians and degrees. The phase is not "unwrapped"; it will always be in the range $-180° < \theta \leq 180°$.

3.7. Analyze – Plot Frequency Response

Plot Frequency Response window plots both the gain magnitude and phase angle of the filter's response as frequency is varied. In the default axis scaling of gain magnitude in dB and logarithmic frequency it is called the Bode plot. The Plot Frequency Response window plots the "unwrapped" phase response; in a 4^{th} order lowpass Bessel filter, for instance, the phase will smoothly decline from 0° at DC to asymptotically approach -360° at high frequencies.

The analyze box works similarly to the above List Frequency Response window. It also draws a vertical red line in the plot window at the user-input frequency. By default, the red line and frequency-to-be-analyzed edit box is set to the critical frequency (the half-power frequency) selected in the main Specify Filter window. Like the List Frequency response window, the phase angle in the analyze box is not unwrapped.

If a circuit topology and resistors and capacitors have already been selected for the circuit in the Build Circuit screen then the "use actual R/C values" option will be enabled, and permit plotting of the actual circuit response in addition to the ideal circuit response.

3.8. Analyze – Filter User Data

The Filter User Data window permits the user to supply sampled data and view plots of it in the time and frequency domains before and after filtering with the filter defined in the main Specify Filter window. By default a sinusoid of the critical frequency is created and shown pre-and post-filtering; it will show about a 30% reduction in magnitude (precisely $2^{-1/2}$, the half-power point) and a 45° shift in phase.

The format for user-supplied data is a single-column of ASCII data. By default, the Load button will search for files ending in '.txt' in the AFD active directory, but it can load data with any suffix and if pointed to a different directory it will remember that as the data directory for the current session (i.e., until AFD is closed). The provided data file may optionally also include a sampling frequency (which if supplied may be overridden in the Sample Frequency edit box). The format to embed sample frequency information in the data file requires the first line of data to consist of the sample frequency followed by the words " Hz sample frequency". Only sample frequencies in Hz are permitted (e.g. "10 Hz sample frequency" is a valid first line, but "10 kHz sample frequency" is not).

The words "default" (without the quotation marks) can be written in either the Time or Power Spectral Density analysis boxes to view reasonable choices, or numeric values may be chosen. If 0 is chosen as the lower frequency and logarithmic spacing is chosen for the frequency axis, the 0 will be changed to a small number greater than 0.

As with the other windows, if a circuit topology and resistors and capacitors have already been selected for the circuit in the Build Circuit screen then the "use actual R/C values" option will be enabled, and permit plotting of the actual circuit response in addition to the ideal circuit response.

3.9. Build – Build Circuit

An active circuit implementation using opamps can be designed using the Build Circuit window. The Build Circuit window is different from all other windows because its data is persistent; if a circuit topology and resistor and capacitor standard values are chosen, and the window is then closed, many other windows will be able to access the new filter transfer function (which will be slightly different from the ideal transfer function if resistor and capacitor tolerances other than 0% are chosen). Further, the Build Circuit window, if reopened, will show the values last chosen. All values will be reset to default values if the filter type is changed in the main Specify Filter window.

The Build Circuit is in a separate menu choice not only because it represents a different type of operation (a design operation rather than an analysis type), but because if there is interest, passive or transmission line implementations may be added in future versions.

The window is divided into three different sections: the top third displays variables common to the entire filter, the middle third displays the circuit type for the currently-selected stage, and the bottom third allows the user to choose specific component values and view calculated values.

The top third displays the number of circuit stages needed to construct the filter, and highlights the currently selected stage in red. Only zero-order (gain), first-order, and second-order opamp circuits are available for design, so a fifth-order filter, for example, must be first be divided into two second-order stages and a single first order stage. First order and gain stages are always placed last, and since first order stages can be of either polarity (inverting or non-inverting) and implement any gain, there will never be both a first-order filter and a gain stage. AFD intelligently groups pole and zero pairs and gains in such a way to minimize the likelihood that any particular stage's high Q (tendency to oscillate at large magnitudes) will saturate following stages. Passband polarity refers to whether a 180° phase shift occurs in the passband and may be set to "don't care" (in which case the choice of polarity is made to yield the simplest circuit), or to "inverting" or "non-inverting". Component tolerance refers to whether standard value resistors and capacitors will be used (e.g. whether a 6.7 kΩ resistor will be rounded to the closest 5% standard value of 6.8 kΩ).

The middle third of the window shows the circuit schematic for the currently selected stage and allows the user to change the selected circuit implementation from the recommended circuit type. The recommended circuit type is the simplest circuit that satisfies the specified polarity choice, the required passband gain, the locations of the poles and zeros it must implement, and the preciseness in position with which the poles and zeros must be located. For example, a high-order elliptic filter may have poles and zeros that are nearly co-located, and thus requires a more complex circuit type (like the Ackerberg-Mossberg) that is less sensitive to variations in V_{offset} of the opamps and has many identical resistor values allowing the use of matched resistor packages to be used, rather than a Multiple Feedback configuration that although simpler to build has a higher sensitivity to non-ideal opamp characteristics and component variance that will likely cause trouble. Unlike many other filter-design software packages that list every conceivable circuit implementation, AFD only allows the user to choose circuit types that are optimal in some sense (either in circuit simplicity or in insensitivity to component variance). State-variable circuits, for instance, although are commonly seen in circuit reference books because the are pedagogically appealing, do not offer the simplicity of the Multiple Feedback biquad nor the component variance insensitivity of the Ackerberg-Mossberg, and hence are not available. The stage details button opens another window (described in the section below) that provides much more detail about the behavior of the currently-selected stage. The Print Stage button prints the schematic and component values for reference while building the circuit.

Component values may be selected in the lower window. Depending on the component tolerances selected, an entered value (such as 2.3 kΩ) may be instantly changed to a standard value (e.g. 2.2 kΩ if 5% resistors are chosen). Occasionally resistors of 0 ohms or "not present" values may be shown; in these cases replace the resistor with a short or open, respectively.

3.10 Build – Build circuit – Stage details

The Stage Details window is accessed using the like-named pushbutton in the Build Circuit window and provides details about the current stage's circuit behavior. This window is useful as an aid when building the circuit, since it permits debugging each stage of the circuit individually (e.g. inject a sinusoid of a known frequency and compare the physical circuit's output with that predicted by the analysis window).

The stage details may be shown using either ideal component values or the actual selected values that are rounded to the nearest standardized value. This produces an unexpected result in one situation. Occasionally in a multiple-stage circuit, the passband gain errors in each stage from selected standard-

tolerance components may unavoidably compound so that the passband gain deviates more than 6% from the ideal passband gain. In that situation AFD automatically adds a final corrective gain stage. That "ideal" gain stage would not be necessary, so if the ideal component choice is selected from this gain stage it will show a simple unity-gain characteristic.

A variety of metrics are calculated and shown: both high and low frequency gain magnitude, the minimum and maximum gain magnitudes and at what frequencies these occur, the stage half-power point, and the gain magnitude is listed and graphed at a user-supplied custom frequency (chosen by default to be the critical frequency of the entire filter). The transfer function H(s) is also shown for that particular stage in both pole/zero and Q/w_o representations.

The Sensitivity Analysis window is particularly useful to determine whether very exact values of each component must be used. It computes how much the given output metric changes (e.g. w_o) given a change in the input component (e.g. C_a). If the sensitivity is 2 in the above example, then a 5% in C_a's value will cause a 10% change in w_o. If the magnitude of the sensitivity is very large then a small change in the component will cause a large change of the metric, indicating that a precise value of the chosen component must be used. A positive or negative value indicates that an increase in the component value causes a respective increase or decrease in the measured parameter value.

4.0 Troubleshooting

There are three common problems that may cause problems using AFD: not having added AFD to the Matlab path, using a version of Matlab prior to 6.5, and not using the latest update of AFD (you may be experiencing a bug that has fixed in the current release).

To add AFD to Matlab's path, from Matlab's menu select "File" then "Set Path". Click the "Add Folder" button, then browse to the AFD folder location. Select it, choose "OK", and then choose "Save" from the bottom of the Set Path dialog box.

Nearly every update of Matlab is shipped with minor inconsistencies with the previous version. At one time there was a separate version of Filt for the student version 3.5, the Windows version 4.0 *and* 4.2 *and* 5.3, *and* the DOS version 3.5. It is nice to know that the folks at Mathworks aren't limiting themselves by always maintaining backwards compatibility, but this type of constant fiddling with standards is annoying to developers. Starting with AFD 2.0, there is simply one version, and it will be kept updated, as long as there is user interest, to the current Matlab version.

To obtain the current version of AFD, visit http://www.jimsquire.com and click "research". The toolbox is saved as the single compressed file afd.zip. It does not have subdirectories and can be unzipped anywhere; I suggest creating a folder for it called "AFD" that resides off the toolbox directory (frequently c:\Matlab\toolbox).

5.0 Suggestions, mailing list for updates, and bugs

Do you find a need to save the circuit topology? Design filters of greater than eighth order? Use other analysis types? Let me know by sending an email to squirejc@vmi.edu with a subject line of "AFD improvement". I am also keeping a mailing list to alert users of updates; send an email to the above address with the subject line "AFD update". Lastly and unfortunately, I am sure some of you will uncover bugs in the program. Please let me know what they are and how to reproduce them; also send me the build number and Matlab version number you are using (accessed by typing "ver" at the Matlab prompt). Send the information with the subject line "AFD bug" and I'll try to correct it in the next release.