Laszlo Hars

Frequency Response Compensation with DSP

In modern telecommunication systems, there are situations when test instruments must work over hundreds of narrowband-frequency channels. It is difficult and expensive, however, to build hardware test equipment with a flat frequency response over their full operational frequency range. In this situation simple finite impulse response (FIR) filters can be applied for gain compensation to improve an instrument's frequency response flatness. This column describes a very fast method for run time design of these filters, while minimizing storage requirements, based on test instrument calibration data.

Filter Table

When a particular center frequency for a channel-under-test is selected, a simple FIR filter can be used to improve test instrument gain flatness. However, computing and storing the necessary filter coefficients is often impractical. The frequency responses of the filters depend on the test center frequency, and this would require storing thousands of sets of the filter coefficients along with a table telling us which filter is needed for each center frequency. The filter coefficients have to be determined at calibration time. If the center frequency can be chosen with high precision this precalculated filter approach is unattractive because a set of filter coefficients is needed for each center frequency, requiring huge storage space and a very long calibration time.

Run-Time Filter Design

A better gain compensation filtering approach is to measure the amplitude characteristics of the test instrument on a sufficiently dense frequency grid when the instrument gets calibrated. Only this table has to be stored. Using interpolation (linear, cubic, or spline, depending the smoothness of the frequency response curve), the required gain compensation filter response can be determined with a handful of arithmetic operations at run time. The compensation filter can be very short if the sampling rate and center frequency are chosen appropriately. Given the required compensation filter gain, we developed a closed-form expression for the filter coefficients allowing us to compute them with only a handful operations. All together the calculations are so fast, that even arbitrary frequency hopping can be implemented with slow, low power DSPs.

Calibration Tables

In most applications, the analog input signal has to be attenuated or amplified by circuits that are, themselves, not perfect either. In theory, we would need a frequency response table for each possible attenuation setting. In practice, however, the frequency response varies only at smaller attenuation values, higher values provide good decoupling between otherwise interfering circuit parts. As such, in practice roughly ten compensation tables are often enough for even high precision measurements. The effects of different attenuation devices are cumulative at proper design (at least at higher attenuation values), that further reduces the number of necessary tables. Temperature compensation can be incorporated, too. The measured ambient or internal temperature represents another dimension for the family of tables.

FIR Versus IIR Filters

Infinite impulse response (IIR) filters have more complicated formulas for their gain response than FIR filters, therefore real-time calculation of the IIR coefficients takes longer. We normally also need constant group delay in the pass band, which is more difficult to achieve with IIR filters. The filters must be stable, that is the roots of the denominator of the transfer function of IIR filters must all lie inside the

To download the software code presented in this article, go to http://www.cspl.umd.edu/spm/tips-n-tricks/.

"DSP Tips and Tricks" introduces practical tips and tricks of design and implementation of signal processing algorithms so that you may be able to incorporate them into your designs. We welcome readers who enjoy reading this column to submit their contributions. Please contact Associate Editor Rick Lyons: ricklyon@onemain.com.

1053-5888/03/$17.00©2003IEEE

complex unit circle, and this is another difficulty to be dealt with. On the other hand, the same long IIR filters can somewhat better approximate a given gain curve. If this curve has very sharp peaks, notches, or edges, IIR filters are needed. Also, the FIR filters can have large group delays. If the group delay must be small or even negative, IIR filters have to be used. The little better approximation of the desired amplitude curve by an IIR filter can be balanced by applying a longer FIR filter, whose coefficient calculations are simpler. The filtering takes about the same time because of the faster FIR filter code. Therefore FIR filters are a good choice. They must be short, having less than ten coefficients, otherwise the calculation of the coefficients gets complicated.

Linear Phase FIR Filters

The frequency response function of an N-tap FIR filter with coefficient sequence $c_0, c_1, \ldots, c_{N-1}$ is given by

$$H(\omega) = \sum_{n=0}^{N-1} c_k e^{-j\omega k}. \tag{1}$$

If the filter has linear phase response (constant group delay), the coefficient sequence must be symmetric or antisymmetric. We want a filter with relative flat amplitude response, that is, close to unity gain everywhere, also at dc. If the coefficient sequence is antisymmetric ($c_k = -c_{N-1-k}$), the dc gain is zero, therefore we need a symmetric coefficient sequence. The filter delay is (N-1)/2. It is easier to compensate or handle an integer number of samples delay, which is what we have if N is odd. In this case the response function becomes

$$H(\omega) = e^{-j(N-1)\omega/2}\left[c_{(N-1)/2} + 2 \cdot \sum_{k=(N+1)/2}^{N-1} c_k \cos\left[\left(k - \frac{N-1}{2}\right)\omega\right]\right]. \tag{2}$$

The factor $e^{-j(N-1)\omega/2}$ represents the delay having a constant magnitude of one, and the real factor in the brackets gives the (signed) amplitude response. The length of the filter can be chosen to be $N = 7$, and it has four free coefficients. We need the desired gain to be a given value at three different frequencies and one degree of freedom remains to enforce a smooth amplitude response curve. Let the coefficient sequence be $[c, b, a, d, a, b, c]$. From (2), the amplitude response of the corresponding FIR filter is

$$A(\omega) = d + 2a \cdot \cos(\omega) + 2b \cdot \cos(2\omega) + 2c \cdot \cos(3\omega). \tag{3}$$

Sampling and Signal Frequency

Usually, if some signal conversion is performed before the amplitude response correction we can simplify the design and save processing time. We need mixing and decimation to reduce the sampling rate f_{Samp} to a little above double the signal bandwidth. This is the minimum, which preserves all the information of the original signal (Nyquist theorem). It usually involves a bandpass filter step too, removing those disturbing signals that would alias to the useful frequency band. This frequency band is best located around the center frequency f_0, where $f_0 = f_{\text{Nyq}}/2 = f_{\text{Samp}}/4$ (or at $3f_{\text{Nyq}}/2$), such that no signal component aliases back into the useful frequency band at another location. (On the normalized scale where the Nyquist frequency is 1, $\omega = \pi \cdot f$ and $f_0 = 1/2$.) Using a normalized frequency axis, the FIR filter's amplitude response can be expressed

$$A(f) = d + 2a \cdot \cos(\pi f) + 2b \cdot \cos(2\pi f) + 2c \cdot \cos(3\pi f). \tag{4}$$

Filter Design

When applying the compensation filter we do not want to change the amplitude at f_0, the center of the frequency band of the signal. (If necessary, we adjust the overall gain outside of the filter.) For the frequency response compensation we specify the gains g_1 and g_2 of the filter at two other frequencies, say $f_1 = 1/4$ and $f_2 = 3/4$ at both sides of f_0. (They can be chosen closer or further away from f_0, according to the need to have more accurate compensation close to the center frequency or less accurate compensation over the whole band.) These represent three constraints, unity gain at f_0 and gains g_1 and g_2 at f_1 and f_2, for the filter response which is a function of four free coefficients. Using (4) we can express the three amplitude constraints as:

$$1 = d + 2a \cdot \cos\left(\frac{\pi}{2}\right) + 2b \cdot \cos(\pi) + 2c \cdot \cos\left(\frac{3\pi}{2}\right) \tag{5}$$

$$g_1 = d + 2a \cdot \cos\left(\frac{\pi}{4}\right) + 2b \cdot \cos\left(\frac{\pi}{2}\right) + 2c \cdot \cos\left(\frac{3\pi}{4}\right) \tag{6}$$

$$g_2 = d + 2a \cdot \cos\left(\frac{3\pi}{4}\right) + 2b \cdot \cos\left(\frac{3\pi}{2}\right) + 2c \cdot \cos\left(\frac{9\pi}{4}\right). \tag{7}$$

A very important additional requirement is that the filter must not have large ripple, i.e., its amplitude response must be smooth. This can be guaranteed if we mandate a fourth constraint to require the slope of the response curve at f_0 be the same as that of the secant line connecting the points $[f_1, g_1]$ and $[f_2, g_2]$. The slope of the curve, the derivative, tells how steep the response curve is in the neighborhood of a given point. It is the same as the slope of the tangent line of the filter response curve. The slopes of the secant lines $[f_1, g_1]$ and $[f_2, g_2]$ around a point $[f_0, 1]$ approximate the slope of the tangent $A'(f_0)$ of the (hopefully smooth) function A in (4). If we require an exact equality, it intuitively ensures some kind of smoothness. Of course, you can specify more complicated conditions, but in our case the following equality proved sufficient. That is:

$$\frac{g_1 - g_2}{f_1 - f_2} = A'(f_0) = \frac{d[A(f)]}{df}$$
$$= -2\pi a \cdot \sin(\pi f_0) -4\pi b \cdot \sin(2\pi f_0) -6\pi c \cdot \sin(3\pi f_0). \quad (8)$$

Because $f_1 - f_2 = -1/2$, and $f_0 = 1/2$, we have

$$g_1 - g_2 = \pi a \cdot \sin\left(\frac{\pi}{2}\right) + 2\pi b \cdot \sin(\pi) + 2\pi c \cdot \sin\left(\frac{3\pi}{2}\right). \quad (9)$$

This last requirement, constraint (9), now gives us a linear system of four equations for the four unknown coefficients of the filter. Evaluating (5) through (7), and (9), for the real values of the trigonometric functions gives

$$1 = d - 2b \quad (10)$$

$$g_1 = d + \sqrt{2}a - \sqrt{2}c \quad (11)$$

$$g_2 = d - \sqrt{2}a + \sqrt{2}c \quad (12)$$

$$g_1 - g_2 = \pi a - 3\pi c. \quad (13)$$

Solving those equations, we get a simple solution for our filter coefficients

$$a = (g_1 - g_2) \cdot \left(\frac{3\sqrt{2}}{8} - \frac{1}{2\pi}\right) \quad (14)$$

$$b = (g_1 + g_2) \cdot \frac{1}{4} - \frac{1}{2} \quad (15)$$

$$c = (g_1 - g_2) \cdot \left(\frac{\sqrt{2}}{8} - \frac{1}{2\pi}\right) \quad (16)$$

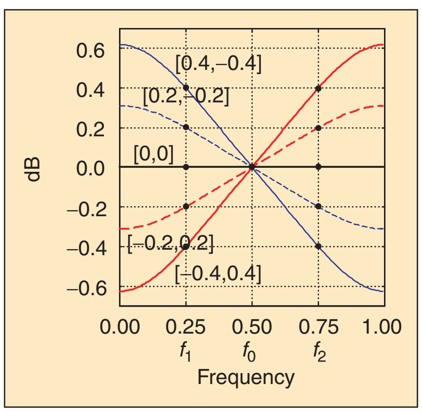
$$d = (g_1 + g_2) \cdot \frac{1}{2}. \quad (17)$$

So, based on predetermined test instrument calibration data stored as an array of g_1 and g_2 values versus center frequency, the seven-tap FIR filter's (13) through (16) coefficients are computed and used in real-time as new center frequencies are assigned during test instrument operation.

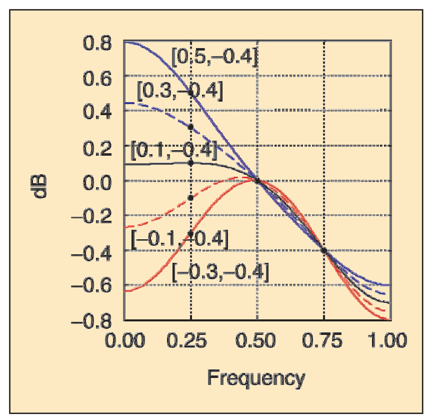
MATLAB Simulation

The function in Table 1, in MATLAB code, calculates the desired filter coefficients based on the desired gains at the frequencies $f_0 = 1/2$, $f_1 = 1/4$, and $f_2 = 3/4$.

Using the code in Table 1 to compute and plot filter responses of linear compensation curves with decibel gains of [0.4,–0.4], [0.2,–0.2], [0,0], [–0.2,0.2], [–0.4,0.4], we have those shown in Figure 1. The desired gain compensation values are indicated by the dots. Example curves with decreasing gains of [0.5,–0.4], [0.3,–0.4], [0.1,–0.4], [-0.1,–0.4], [-0.3,–0.4], in decibels, are provided in Figure 2. Finally, example curves with increasing gains of [0.3,0.4], [0.1,0.4], [–0.1,0.4], [–0.3,0.4], [–0.5,0.4], in dB, are shown in Figure 3. They agree completely with the gain compensation we wanted.



▲ 1. Example gain compensation curves for the N = 7 FIR filter.



▲ 2. Example gain compensation curves for decreasing gain versus frequency.

Table 1

```
function v = relatflt(dB1,dB2)
%relatflt(dB1,dB2) len=7 FIR filter of response:
% 0 dB gain at f0 = 1/2
% dB1 = gain compensation in dB at f1 = 1/4
% dB2 = gain compensation in dB at f2 = 3/4
g1 = 10^(dB1/20); % convert dB gain to linear
g2 = 10^(dB2/20); % convert dB gain to linear
a = (g1-g2)*0.37117514279802; % 3sqrt(2)/8 - 1/2/pi
b = (g1+g2)*0.25 - 0.5;
c = (g1-g2)*0.01762175220474; % sqrt(2)/8 - 1/2/pi
d = (g1=g2) *0.5;
v = [c b a d a b c];
```

Implementation in C

Calculation of the FIR filter coefficients is straightforward in the C language as shown in Table 2. The floating point variables **d1** and **d2** specify the desired filter gains in decibels at $f_1 = 1/4$ and $f_2 = 3/4$. The filter coefficients are stored after calculation in the array **Filt[Len]**, with **Len** = 7.

Extensions

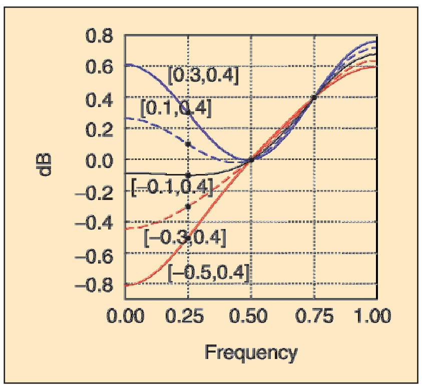
The designed filters work well even beyond a ±6 dB compensation range. These large flatness errors, however, should not occur. They indicate serious test instrument hardware faults. If the shape of the compensation curve needs to be more complex, we can easily add to the constraints a second pair of frequencies with specified gains. We should request the slope of the response curve at the innermost frequency points be equal to the slope of the secant going through the surrounding specified curve points. We need now a 15-tap filter of the form: $a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7$.

Let $f_k = k\pi/6$ be the five frequencies where the gain will be specified, thus $k = 1,2,\ldots,5$, and the corresponding gain values are g_k normalized at the center with $g_3 = 1$. We have five equations describing the correction gains, similar in form to (5) through (7):

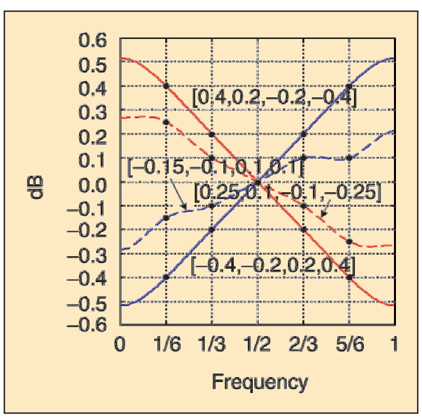
$$g_k = a_0 + 2\sum_{i=1}^{7} a_I \cos(i\pi/6), \quad k = 1,\ldots,5. \tag{18}$$

And another three, requiring the correct slopes, similar in form to (9):

$$g_k - g_{k+2} = (2\pi/3)\sum_{i=1}^{7} a_I \sin(i\pi/6), \quad k = 1,2,3. \tag{19}$$



▲ 3. Example gain compensation curves for increasing gain versus frequency.



▲ 4. Example gain compensation curves for a 15-tap FIR filter.

Table 2

```
#define Len 7
#define C ((Len-1)/2)
/* from dB to ratio: */
float g1 = pow( 10.0, d1 / 20.0);
float g2 = pow( 10.0, d2 / 20.0);
float Filt[Len];
/* 3*sqrt(2)/8 - 1/2/pi?= 0.37117514279802 */
/* sqrt(2)/8 - 1/2/pi?= 0.01762175220474 */
Filt[C-1] = Filt[C+1] = (g1-g2)*0.37117514279802;
Filt[C-2] = Filt[C+2] = (g1+g2)*0.25 - 0.5;
Filt[C-3] = Filt[C+3] = (g1-g2)*0.01762175220474;
Filt[ C ] = (g1+g2)*0.5;
```

Table 3

```
function v = flt15(dB1,dB2,dB4,dB5)
%flt15(dB1,dB2,dB4,dB5) len=15 FIR filter response
% dB1 = gain in dB at f1 = 1/6
% dB2 = gain in dB at f2 = 2/6
% 0 dB gain at f3 = 3/6
% dB4 = gain in dB at f4 = 4/6
% dB5 = gain in dB at f5 = 5/6
g1 = 10^(dB1/20); g4 = 10^(dB4/20);
g2 = 10^(dB2/20); g5 = 10^(dB5/20);
a0 = 0.27883*(g1+g5) + 0.25000*(g2+g4) -0.05767;
a1 = 0.15735*(g1-g5) + 0.27254*(g2-g4);
a2 = 0.19550*(g1+g5) -0.39100;
a3 = 0.01301*(g1-g5) + 0.02254*(g2-g4);
a4 = 0.02883*(g1+g5) - 0.05767;
a5 =-0.08647*(g1-g5) + 0.18169*(g2-g4);
a6 =-0.02725*(g1+g5) + 0.12500*(g2+g4) -0.19550;
a7 =-0.04486*(g1-g5) + 0.09085*(g2-g4);
v =[a7 a6 a5 a4 a3 a2 a1 a0 a1 a2 a3 a4 a5 a6 a7];
```

The solution of the corresponding linear system of equations goes similarly as before [see (20)-(27) at the bottom of the page].

It is more complex, requiring a little more processor power for the run-time design, but the work is still manageable. The MATLAB code found in Table 3 calculates the desired 15-tap filter coefficients with the like terms in (20) through (27) evaluated and $g_3 = 1$.

Figure 4 shows a few example compensation response curves of 15-tap FIR filters designed using the code in Table 3. Again, the desired gain compensation values are indicated by the dots.

Calibration Tables

Usually the input signal gets attenuated and/or amplified before the analog to digital conversion to assure maximum digital resolution. These attenuator-amplifier circuits affect each other to a different degree, dependent on the selected attenuation. In theory, we need a frequency response calibration table for each possible attenuation value. If an instrument needs to be calibrated at several different attenuation levels over the whole frequency range, it will consume a lot of time and cost. However, proper design reduces the number of necessary calibration runs. In practice, without significant effort in the hardware design the frequency characteristics change only at smaller attenuation values, because higher attenuations provide good decoupling between otherwise interfering circuit parts. The affects of decoupled, cascaded, attenuation devices are cumulative (additive when measured in dB) which further reduces the number of necessary tables. Temperature compensation can be incorporated, too. The measured ambient or internal temperature represents another dimension for the family of tables. If the temperature dependency is smooth (linear or close to that), and does not change the shape of the frequency response curve, one extra table is enough for high precision compensation. The correction can be done by a temperature dependent multiplication factor (additive in decibels).

Laszlo Hars is a mathematician by education but otherwise quite a normal person. He is now a researcher at Seagate Technology. He has lectured at universities (math and computer science), led the signal processing work for measurement instrument designs, and worked on many different research topics like large scale optimizations, chip wiring, digital rights management, content protection, digital watermarking, random number generation testing, and cryptographic hardware and system designs.

Further Reading

[1] A. Poularikas, *Fomulas and Tables for Signal Processing*. New York: CRC Press - IEEE Press, 1999.

[2] N. Kalouptsidis, *Signal Processing Systems: Theory and Design*. New York: Wiley, 1997.

[3] J. Proakis and D. Manolakis: *Digital Signal Processing: Principles, Algorithms and Applications*, 3rd ed. Englewood Cliffs, NJ: Prentice Hall, 1996.

[4] N. Fliege, *Multirate Digital Signal Processing: Multirate Systems, Filter Banks, Wavelets*. New York: Wiley, 1995.

[5] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery: *Numerical Recipes in C*. Cambridge, MA: Cambridge Univ. Press, 1992.

[6] L. Rabiner and B. Gold: *Theory and Application of the Digital Signal Processing*. Englewood Cliffs, NJ: Prentice Hall, 1975.

[7] A. Oppenheim and R. Schafer: *Digital Signal Processing* Englewood Cliffs, NJ: Prentice Hall, 1975.

$$a_0 = \frac{-3\sqrt{3}(g_1 - 2g_3 + g_5) + (5g_1 + 3g_2 - 4g_3 + 3g_4 + 5g_5)\pi}{12\pi} \tag{20}$$

$$a_1 = \frac{\left[3(g_2 - g_4) + \sqrt{3}(g_1 - g_5)\right](-2 + \pi)}{4\pi} \tag{21}$$

$$a_2 = \frac{(g_1 - 2g_3 + g_5)\left(-3\sqrt{3} + 4\pi\right)}{12\pi} \tag{22}$$

$$a_3 = \frac{\left[3(g_2 - g_4) + \sqrt{3}(g_1 - g_5)\right](-3 + \pi)}{6\pi} \tag{23}$$

$$a_4 = \frac{(g_1 - 2g_3 + g_5)\left(-3\sqrt{3} + 2\pi\right)}{12\pi} \tag{24}$$

$$a_5 = \frac{72g_4 + 27\sqrt{3}g_5 + 36g_2(-2 + \pi) - 36g_4 - 5\sqrt{3}g_5 + \sqrt{3}g_1(-27 + 5\pi)}{72\pi} \tag{25}$$

$$a_6 = \frac{-3\sqrt{3}(g_1 - 2g_3 + g_5) + (g_1 + 3g_2 - 8g_3 + 3g_4 + g_5)\pi}{24\pi} \tag{26}$$

$$a_7 = \frac{-3\left[3g_1 + 4\sqrt{3}(g_2 - g_4) - 3 - g_5\right] + \left[g_1 + 6\sqrt{3}(g_2 - g_4) - g_5\right]\pi}{24\sqrt{3}\pi} \tag{27}$$