SMOOTH AND SAFE PARAMETER INTERPOLATION OF BIQUADRATIC FILTERS IN AUDIO APPLICATIONS

Victor Kalinichenko

ASK Industries GmbH
Hauptstrasse 73, D-94559
Niederwinkling, Germany
KalinichenkoV@askgroup.de
Private: homekvn@yandex.ru

ABSTRACT

Parameter interpolation (PI) of biquadratic IIR filters (biquads) is widely used in audio applications to avoid very undesirable audible effects like plops, clicks etc. It is shown that although the simplest linear PI of numerator and denominator coefficients of the transfer function of a biquad guarantees the stability it, however, may sometimes lead to other audible side effects caused, for example, by the possible growth of the magnitude response at some frequencies in some intermediate states of PI procedure. To avoid audible transients and to make the PI easily portable into digital signal processor (DSP) a special technique is applied called here "sliding edges". The detailed description of this technique and their comparison is the goal of the present paper.

1. INTRODUCTION

Biquadratic IIR filters (biquads) are used in many audio applications; for example, for individual equalization of audio channels, or in some measurement algorithms to extract signal components within a certain bandwidth, etc. Sometimes it is necessary to change the parameters of one or several biquads together during a certain time with no undesirable audible transient effects that may occur in case of simple switching. The smooth PI is needed, for example, in case of changing the equalization filter-set in the cabin of a car from driver optimized setting to front seats.

A perfect PI must be smooth, safe and rather easy from the viewpoint of computations. The smoothness of PI may be well described at consideration of the behavior of magnitude to frequency response (MR) of the biquad during the PI procedure. It is naturally to require that MR had no sudden changes within the duration of PI. The term "safe" in this context means that during PI no instability or being very close to the stability boundaries may occur. The last demand is also of great importance as some audio systems may involve 100 and more biquads, 20-30 percents of which are to be interpolated in real-time. PI algorithm during rather short working time (usually 2-3 sec) requires additional computation power for all those filters. Using the complicated PI algorithm may be critical for the DSP installed in the audio system.

Different approaches of PI are described in [1, 2, 3, 4, 5, 6, 7, 8, 9]. The general classification of PI techniques is done in [2]. According to this classification the following types of PI are known: a) *Cross-fading method*; b) *Gradual variation of coefficients* [3]-[5]; c) *Intermediate coefficient matrix* [6]; d) *Input switching method* [7] and e) *Updating of the state vector* [8], [9]. Theoretically, all above methods are DSP-implementable. However, all of them have the restricted range of use: methods a) and d) require duplicating the processing power during PI; method b) may cause audible transients especially at harmonic signals; methods c) and e) imply the representation of filters in state-space form; method e) requires a lot of computations. We emphasize on the design of easy-to-implement approach that would operate for the case, when the filter, whose parameters are to be changed, is represented as a cascade of biquads. An attention to DSP-portability will be given.

2. PROBLEM STATEMENT

2.1. Notations and assumptions

Biquad may be described by its transfer function (TF) in z-domain:

$$W(z) = k \cdot \frac{1 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \qquad (1)$$

where a_i, b_i — coefficients being normally constants, k — gain, $z \in \mathbb{C}$. Using the form (1) of TF is usual in DSP applications. Let $W_{\text{src}}(z)$ and $W_{\text{tar}}(z)$ are the TFs of initial and final state of the biquad whose appearance corresponds to (1). Let T_{PI} is the time of *duration of PI*. In case the duration of PI is long enough so that it corresponds to rather large number of input signal samples PI may be done with the lower discreteness than the discreteness of the input signal, *i.e.* updating of coefficients of (1) may be done not every sample but every $2, 3, 4, ...$ sample. We denote the number of samples after which the next update is done as L_{block} . Then $T_{\text{PI}} = N_{\text{PS}} \cdot L_{\text{block}} \cdot T_{\text{S}}$, where T_{s} is the sampling period; the integer number N_{PI} is called the *length* of PI, *i.e.* the total number of updates to complete PI. The notations like $W_{[m]}(z)$, $a_{1[m]}$, etc $m = 0, 1, 2, ..., N_{\text{PI}}$ provided with the index within square brackets relates to the stage of PI, *i.e.* to the number of update. Note that $W_{[0]} = W_{\text{src}}(z), W_{[N_{\text{PI}}]}(z) = W_{\text{tar}}(z)$.

2.2. Criteria for the parameter interpolation algorithm

The practical criteria for PI can be described in the following way:

1. No instability must occur while PI is done. In other words, for every m, $W_{[m]}(z)$ must not have any instable pole.
2. No remarkable audible side effects (like clicks, plops, ringing etc.) must occur during PI.
3. The complexity of PI algorithms must not be very high due to PI is done in real-time.

To illustrate the importance of the above criteria consider the simplest linear PI results drawn in Figure 2a. PI is done for $N_{\text{PI}} = 10$. It is clear that there are substantial drawbacks as PI procedure is not smooth because of great difference of MRs at two neighbor PI stages $m = 0$ and $m = 1$.

3. PROBLEM SOLUTION

The most important requirement while PI is in progress is to ensure the stability at all stages. As the stability condition does not depend on the numerator of (1) it is reasonable to make PI for numerator and denominator of the biquad's TF separately.

3.1. Parameter interpolation of the denominator

3.1.1. Linear PI

Fortunately, according to the investigations made in [10], the stability area of biquad is defined by the following condition:

$$|a_1| < a_2 + 1, \quad \text{and} \quad |a_2| < 1. \tag{2}$$

It shows that all boundaries of the stability region are straight lines and that in theory the linear PI

$$a_{i[m]} = a_{i[m-1]} + \Delta a_i, \quad \Delta a_i = \left(a_{i\ \mathrm{dst}} - a_{i\ \mathrm{src}}\right)/N_{\mathrm{PI}}, \tag{3}$$

(where $i = 1, 2$) will result in the stability at every m if and only if $W_{\mathrm{src}}(z)$ and $W_{\mathrm{dst}}(z)$ are stable. In practice the stability may be lost in case of inaccurate computations due to accumulated error at the summation in (3). To avoid this, the non-recurrent formula must be used:

$$a_{i[m]} = a_{i\ \mathrm{src}} + m \cdot \Delta a_i, \quad \Delta a_i = \left(a_{i\ \mathrm{dst}} - a_{i\ \mathrm{src}}\right)/N_{\mathrm{PI}} \tag{4}$$

It is easy to show that the summation error which is most important due to different order of the values a_i and Δa_i will not be accumulated in case (4). Note also that linear PI is applied to each coefficient independently and it requires minimal computations. Additional tricks must be made to ensure the stability at fixed-point computations.

3.1.2. Pole PI

Although the linear PI of denominator is very simple and provides the system stability it may result in undesirable behavior of poles which are responsible also for resonances in MR. The resonance may cause some audible side effect like "ringing" and in practice may even lead to "clipping" due to extremely high output values. Experiments have shown that in most practical cases it does not take place. However, if the source or the destination filters has the Q-factor more than 5–6, then in some cases high resonances may occur. If such cases a more sophisticated PI law should be applied.

The idea of the PI of poles is in step-by-step shifting of the poles and then recomputing new values of the coefficients a_i at each stage. Let the poles (the roots of denominator) of (1) are z_1 and z_2. The association between a_i and z_i is expressed in the following formulae:

$$a_1 = -(z_1 + z_2), \quad a_2 = z_1\ z_2, \tag{5}$$

$$z_{1,2} = 0.5 \cdot \left(-a \pm \sqrt{a_1^2 - 4a_2}\right). \tag{6}$$

Basing on (6) it is possible to calculate $z_{i\ \mathrm{src}}$ and $z_{i\ \mathrm{dst}}$. Depending on $a_{i\ \mathrm{src}}$ and $a_{i\ \mathrm{dst}}$ three cases of PI trajectories are possible:

1. Both couples of poles $z_{i\ \mathrm{src}}$ and $z_{i\ \mathrm{dst}}$ are real numbers;
2. Both couples of poles are complex[1] values;
3. One pole is complex and another one is real.

[1] In principle, $\mathbb{R} \subset \mathbb{C}$. However, in this context saying that a value x is complex, we imply that $x \in \mathbb{C} \setminus \mathbb{R}$.

In case of complex poles z_1 and z_2 they must be conjugant, *i. e.* $z_1 = z_2^* \iff \Re(z_1) = \Re(z_2), \Im(z_1) = -\Im(z_2)$. Depending on the type of poles there may be different but linear PI trajectories; see Figure 1, a-c. In case 3 the trajectory of poles consists of two straight-line pieces. The shift of poles at each straight piece is done according to linear law:

$$z_{i\ [m]} = z_{i\ [0]} + m \cdot \Delta z_i. \tag{7}$$

With a small adaptation the formula (7) is valid even if PI trajectories are of type 3. Note that formulae (5) and (7) are also very simple from the viewpoint of computation and thus satisfies the criterion 3.

3.2. Parameter interpolation of the numerator. "Sliding edges" algorithm

The algorithm for the readjustment of the numerator coefficients of TF is based on the idea which can be called "sliding edges". This term relates to two specific points of frequency response: namely, $\omega_1 = 0$ and $\omega_2 = \pi$ (*i.e.* Nyquist frequency). Let the source biquad has the edge points equal to

$$\begin{aligned} g_{1\ [0]} &= A_{[0]}(\omega_1) = A_{\mathrm{src}}(\omega_1), \\ g_{2\ [0]} &= A_{[0]}(\omega_2) = A_{\mathrm{src}}(\omega_2), \end{aligned} \tag{8}$$

where

$$A_{[m]}(\omega) = \left|W_{[m]}(z)\right|_{z=\exp(j\omega)}. \tag{9}$$

Thus, g_1 and g_2 are non-negative. Let the destination biquad has the edge points

$$\begin{aligned} g_{1\ [N_{\mathrm{PI}}]} &= A_{[N_{\mathrm{PI}}]}(\omega_1) = A_{\mathrm{dst}}(\omega_1), \\ g_{2\ [N_{\mathrm{PI}}]} &= A_{[N_{\mathrm{PI}}]}(\omega_2) = A_{\mathrm{dst}}(\omega_2). \end{aligned} \tag{10}$$

Then, moving the edges:

$$g_{1\ [m]} := A_{[m]}(\omega_1), \quad g_{2\ [m]} := A_{[m]}(\omega_2) \tag{11}$$

use of some smooth law (for example, linear law) will be assumed to result in the appropriate intermediate MRs through all transitions. The above idea exploits the assumption based on the analysis of experimental data that the biggest "jump" between two neighbor stages of MR may occur namely because of big jump at the edge values. Then, the natural way to "impel" MRs not to "jump" very strongly is to move their edges along vertical axis $\omega_1 = 0$ and $\omega_2 = \pi$ using some smooth law. As hinted before the linear law

$$\begin{aligned} &g_{i\ [m]} = g_{i\ [m-1]} + \Delta g_i = g_{i\ \mathrm{src}} + m \cdot \Delta g_i, \\ &\Delta g_i = \left(g_{i\ \mathrm{dst}} - g_{i\ \mathrm{src}}\right)/N_{\mathrm{PI}}, \quad i = \{1, 2\} \end{aligned} \tag{12}$$

is chosen. Let us establish the correspondence between edge values g_i and the coefficients a_i, b_i and k. Note that as it follows from (9) the value of argument at the edges $\omega = 0$ and $\omega = \pi$ corresponds to $z = 1$ and $z = -1$ respectively. Taking into account this fact and also (11) we may conclude that:

$$\left|W_{[m]}(1)\right| = g_{1\ [m]}, \quad \left|W_{[m]}(-1)\right| = g_{2\ [m]}. \tag{13}$$

Unwrapping (13) and taking into account (2) will result in:

$$\begin{aligned} \frac{\left|1+b_{1\ [m]}+b_{2\ [m]}\right|}{\left(1+a_{1\ [m]}+a_{2\ [m]}\right)} \cdot \left|k_{[m]}\right| &= g_{1\ [m]}, \\ \frac{\left|1-b_{1\ [m]}+b_{2\ [m]}\right|}{\left(1-a_{1\ [m]}+a_{2\ [m]}\right)} \cdot \left|k_{[m]}\right| &= g_{2\ [m]}, \end{aligned} \tag{14}$$

Assuming that for most practical applications k and the values in the module brackets are non-negative, we can express the numerator values:

$$\begin{aligned} b_{1\,[m]} &= \frac{(g_{1\,[m]}-g_{2\,[m]})(1+a_{2\,[m]})+(g_{1\,[m]}+g_{2\,[m]})a_{1\,[m]}}{2k_{[m]}}, \\ b_{2\,[m]} &= \frac{(g_{1\,[m]}+g_{2\,[m]})(1+a_{2\,[m]})+(g_{1\,[m]}-g_{2\,[m]})a_{1\,[m]}}{2k_{[m]}-1}. \end{aligned} \quad (15)$$

Thus, applying the linear or pole PI for denominator, linear PI for $k_{[m]}$ and the formula (12), one can obtain from (15) the numerator coefficients b_i automatically.

3.3. Aspects of parameter interpolation for cascaded biquads

If it is necessary to do PI in a cascade of biquads all the above formulae are valid. The only difficulty may occur due to the circumstance that the gains k of each cascade are normally not stored in the memory due to storing only the entire overall gain being a multiplicative product of gains of all cascades. Unfortunately direct PI of overall gain may generally lead to extremely big or small values in intermediate MRs. It is easy to see from the following fact:

$$k_{1\,[m]} \times k_{2\,[m]} \times ... \times k_{L\,[m]} \neq k_{\text{overall}\,[m]} \quad (16)$$

where $k_{l\,[m]} = k_{l\,\text{src}} + m \cdot \Delta k_l$ is the gain of cascade number $l = \{1, 2, ..., L\}$ at the PI stage number m; L is the number of cascades; overall $k_{\text{overall}\,[m]} = k_{\text{overall src}} + m \cdot \Delta k_{\text{overall}}$ is the overall gain at the PI stage number m. Thus, two variants are possible: either to store and then interpolate the gain of each cascade individually, or to use the multiplicative PI for overall gain, *i.e.*

$$k_{\text{overall}\,[m]} = k_{\text{overall src}} \cdot \delta^m, \quad \delta = (k_{\text{overall dst}}/k_{\text{overall src}})^{1/m}. \quad (17)$$

The inequality (16) in the last case would turn to the equality. This, however, may cause different problems if $k_{\text{overall src}} \leq 0$.

3.4. Capability to start new parameter interpolation while previous interpolation is still in progress

Practical usage of filters with PI may require the ability to start new PI while the previous PI is still in progress. This feature must be implemented in case when it is not 100% priory defined that the next service command used for starting PI will not interrupt the previous one. To start new PI it is necessary to store current values of $a_{i\,[m]}$, $b_{i\,[m]}$, $k_{[m]}$ and to consider them as source values. All the above mentioned PI methods allow us to do this procedure.

4. RESULTS OF MODELLING

To show the quality of each PI algorithm the modeling is done at the number of PI stages $N_{\text{PI}} = 10$. For the synthesis of biquads the formulae from [10] are used. The computation of all types of biquads is based on four parameters: a) filter type: LPF1, LPF2, HPF1, HPF2, Notch, Peak, BPF, APF1, APF2; b) f_0 — cut-off frequency; c) L — pass band level and d) Q — Q-factor. First two letters in the filter type abbreviation denote: LP — low-pass, HP — high-pass, BP — band-pass, AP — all-pass; the third letter "F" denotes "filter"; the number denotes the order of filter. Three variants of PI are compared: 1) linear denominator + linear numerator (LDLN) PI; 2) linear denominator + sliding-edge numerator (LDSEN) PI; 3) pole denominator + sliding-edge numerator (PDSEN) PI. The results of modeling are represented in Figures 2-3, a-c.

5. ANALYSIS OF THE RESULTS AND CONCLUSIONS

Let us analyze the suitability of the modelling results to the criteria 1–3 introduced in the section 2.2. LDLN PI satisfies fully the criteria 1 and 3. However, in some cases the criterion 2 is failed. LDSEN PI also satisfies the criteria 1 and 3; a set of potential cases leading to the failure of the criteria 2 is reduced in comparison with LDLN. PDSEN PI satisfy the criteria 1-3 (the criterion may probably fail only in some "exotic" cases). Its drawback is the relative complexity of program implementation due to separate consideration of three types of trajectories described in the subsection 3.1.2. But, it does not mean that PDSEN will require a lot of processing power in real-time mode: only the program code will grow.

Although the proposed LDSEN and PDSEN PI techniques are not the ideal solutions they are the good alternatives to the intuitive linear PI LDLN. Anyway the solution on what type of PI to use follows from the prior information about possible type of the filter to be interpolated.

6. REFERENCES

[1] R. J. Clark, "Investigation into digital audio equalizer systems and the effects of arithmetic and transform errors on performance," Ph.D. dissertation, Univ. of Plymouth, 2001, [Online] http://www.tech.plym.ac.uk/spmc/pdf/audio/RobClarkPhD.pdf.

[2] V. Välimäki, "Discrete-time modeling of acoustic tubes using fractional delay filters," Ph.D. dissertation, Helsinki University of Technology, 1995, [Online] http://www.acoustics.hut.fi/~vpv/publications/vesan_vaitos.

[3] J. N. Mourjopoulos, E. D. Kyriakis-Bitzaros, and C. E. Goutis, "Theory and real-time implementation of time-varying digital audio filters," *J. Audio Eng. Soc.*, vol. 38, no. 7/8, pp. 523–536, Jul./Aug. 1990.

[4] U. Zölzer, B. Redmer, and J. Bucholtz, "Strategies for switching digital audio filters," in *95th Conv. Audio Eng. Soc.,* New York, USA, Oct. 1993, preprint 3714.

[5] S. S. Nikolaidis, J. N. Mourjopoulos, and C. E. Goutis, "A dedicated processor for time-varying digital audio filters," *IEEE Trans. Circuits and Systems—II: Analog and Digital Sig. Proc.*, vol. 40, no. 7, pp. 452–455, July 1993.

[6] R. Rabenstein, "Minimization of transient signals in recursive time-varying digital filters," *Circuits, Systems, and Sig. Proc.*, vol. 7, no. 3, pp. 345–359, 1988.

[7] W. Verhelst and P. Nilens, "A modified-superposition speech synthesizer and ist applications," in *Proc. IEEE Int. Conf. Acoust., Speech, and Sig. Proc. (ICASSP'86),* Tokyo, Japan, vol. 3, 1986, pp. 2007–2010.

[8] L. H. Zettenberg and Q. Zhang, "Elimination of transients in adaptive filters with application to speech coding," *Signal Processing*, vol. 15, no. 4, pp. 419–428, Dec. 1988.

[9] V. Välimäki and T. I. Laakso, "Suppression of transients in time-varying recursive filters for audio signals," in *Proc. IEEE Int. Conf. Acoust., Speech, and Sig. Proc. (ICASSP'98),* Seattle, USA, vol. 6, May 1998, pp. 3569–3572, [Online] http://www.acoustics.hut.fi/~vpv/publications/icassp98-trel.pdf.

[10] K.-D. Kammeyer and K. Kroschel, *Digitale Signalverarbeitung. Filterung und Spektralanalyse mit MATLAB-Übungen*, 5th ed. B. G. Teubner, 2002.

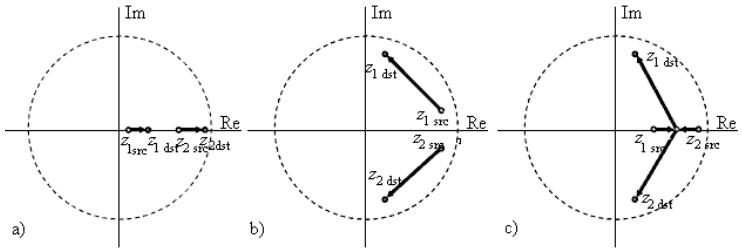
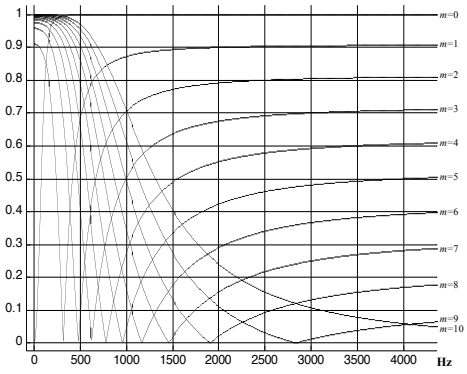
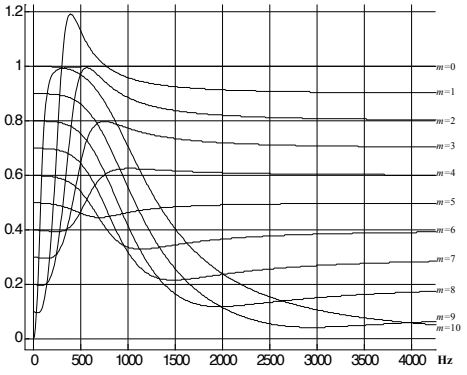


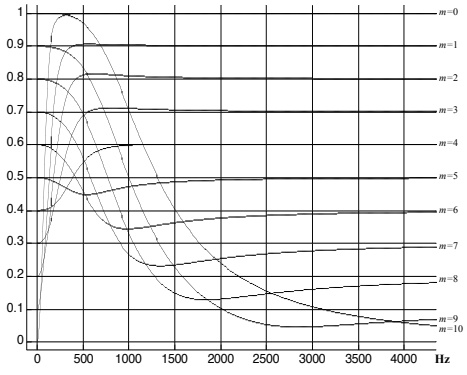
Figure 1: *Three variants of pole moving for source and destination biquad.*



(a)

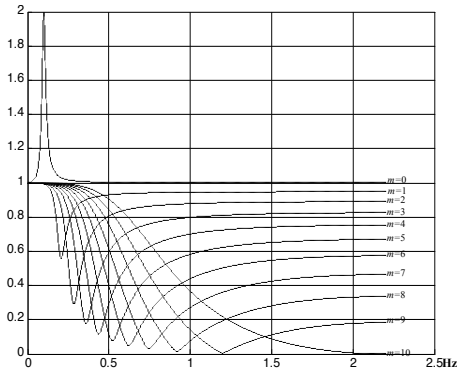


(b)

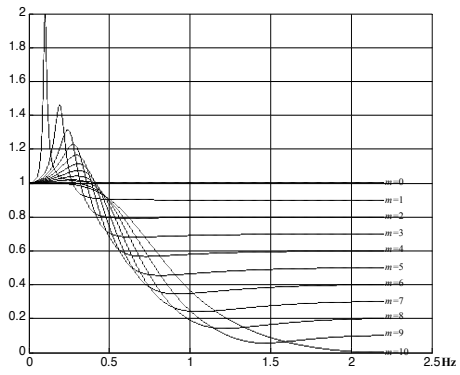


(c)

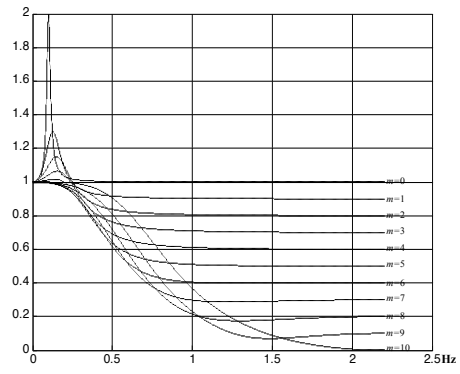
Figure 2: *PI examples of a filter with the parameters: source: type: HPF2, $f_0 = 100$ Hz, $L = 0$ dB, $Q = 0.707$; destination: type: LPF2, $f_0 = 1$ kHz, $L = 0$ dB, $Q = 0.707$; a) LDLN, b) LDSEN, c) PDSEN.*



(a)



(b)



(c)

Figure 3: *PI examples of a filter with the parameters: source: type: PEAK, $f_0 = 1$ kHz, $L = 6$ dB, $Q = 3.0$; destination: type: LPF2, $f_0 = 7$ kHz, $L = 0$ dB, $Q = 0.707$; a) LDLN, b) LDSEN, c) PDSEN.*