Proceedings of the 1998 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'98), vol. 6, pp. 3569–3572, Seattle, Washington, May 12–15, 1998

SUPPRESSION OF TRANSIENTS IN TIME-VARYING RECURSIVE FILTERS FOR AUDIO SIGNALS

Vesa Välimäki[1] and Timo I. Laakso[2]

[1]Helsinki University of Technology, Laboratory of Acoustics and Audio Signal Processing
P.O. Box 3000, FIN-02015 HUT, Finland
E-mail: Vesa.Valimaki@hut.fi, URL: http://www.acoustics.hut.fi/~vpv/

[2]Helsinki University of Technology, Laboratory of Telecommunications Technology
P.O. Box 3000, FIN-02015 HUT, Finland
E-mail: Timo.Laakso@hut.fi

ABSTRACT

A new method for suppressing transients in time-varying recursive filters is proposed. The technique is based on modifying the state variables when the filter coefficients are changed so that the filter enters a new state smoothly without transient attacks, as originally proposed by Zetterberg and Zhang. In this contribution we modify the Zetterberg–Zhang algorithm to render it feasible for efficient implementation. We explain how to determine an optimal transient suppresser to cancel the transients down to a desired level at the minimum complexity of implementation. The application of the method to time-varying all-pole and direct-form II filter structures is studied. The algorithm may be generalized for any recursive filter structure. The transient suppression technique finds applications in audio signal processing where the characteristics of a recursive filter needs to be changed in real time, such as in music synthesis, auralization, and equalization.

1. INTRODUCTION

Due to the recursive nature of IIR filters, abrupt changes in filter coefficients cause disturbances to values of internal state variables and thus result in transients at the filter output. These transients may cause serious trouble for practical applications, such as clicks in audio signals, and they are a critical problem in the implementation of time-varying recursive filters. Many different approaches have been proposed for suppressing transients in time-varying recursive filters: a cross-fading method [1], [7], gradual variation of coefficients using interpolation [2], intermediate coefficient matrix [3], and updating of the state vector [8].

The most general approach to transient suppression is the *state-variable update technique* introduced by Zetterberg and Zhang [8]. They state that every change in filter coefficients should be accompanied by an appropriate change in the internal state variables. The Zetterberg–Zhang (ZZ) method can completely eliminate the transients but it does require that all the past input samples are known. This makes the approach impractical as such but provides a fruitful starting point for more efficient approximate algorithms. In this paper we build on the ZZ method.

The motivation for our work has been to find a practical way to update the state variables of a recursive filter in real time when the filter coefficients are changed abruptly. We present a solution for transient suppression that gives an acceptable performance at the minimum implementation complexity. In this paper we show how the new technique is used with the all-pole filter structure. The transient cancellation method may be generalized for any IIR filter, including cascade and parallel structures.

2. TIME-VARYING RECURSIVE FILTERS

2.1 Output-Switching Method

Let us consider a recursive Nth-order filter with transfer function

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + \ldots + b_N z^{-N}}{1 + a_1 z^{-1} + \ldots + a_N z^{-N}} \qquad (1)$$

where b_k and a_k are its numerator and denominator coefficients, respectively (k = 0, 1, ..., N). Assuming a causal implementation, the input-output relation of this filter may be expressed as

$$y(n) = \sum_{k=0}^{N} b_k x(n-k) - \sum_{m=1}^{N} a_m y(n-m), \quad \text{for } n \ge 0 \qquad (2)$$

where $x(n)$ and $y(n)$ are the input and output signal of the filter, respectively, and they are assumed to be stationary signals.

In order to understand what the change of the filter characteristics means for the filter output, we consider a single change of the coefficient set at time index $n = n_c$. Ideally, the filter should instantly reach its *steady state* and there would not be any disturbances in the output signal after the change. This can be achieved by running two filters, $H_1(z)$ and $H_2(z)$, in parallel as shown in Fig. 1. The output signals of these two filters are

$$y_i(n) = x(n) * h_i(n) = \sum_{k=-\infty}^{\infty} x(k) h_i(n-k) \qquad (3)$$

where i = 1, 2 and the asterisk denotes discrete-time convolution. The output is switched at time index $n = n_c > 0$ and the output of the system can be expressed as

$$y_{\mathrm{id}}(n) = \begin{cases} y_1(n) & \text{for } 0 \le n < n_c \\ y_2(n) & \text{for } n \ge n_c \end{cases} \qquad (4)$$

We call this the *output-switching method* for implementing time-varying recursive filtering, and it represents the ideal case where the change in filter coefficients does not introduce any transients.

In a practical situation where multiple coefficient changes occur, realization of a time-varying filter using the output-switching

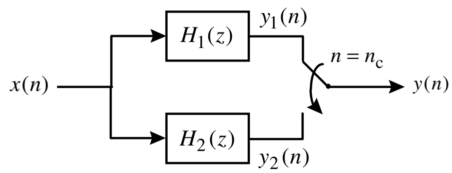


Figure 1. The output-switching method.

method of Fig. 1 grows increasingly complex. For example, if it is needed to switch between one hundred different filter coefficient sets in a given application, the transient-free implementation requires 100 filters running in parallel. This also implies that the filter coefficient sets must be known beforehand.

2.2 Transient versus Discontinuity

The output signal of a time-varying recursive filter whose coefficients are changed at time $n = n_c$ may be expressed as

$$y(n) = \begin{cases} y_1(n) & \text{for } 0 \le n < n_c \\ y_2(n) + y_t(n) & \text{for } n \ge n_c \end{cases} \tag{5}$$

where $y_1(n)$ and $y_2(n)$ are the steady-state responses of the filter before and after, respectively, the change in the coefficient set, and $y_t(n)$ is the *transient signal*. It is defined as the difference between the actual (5) and ideal output signals (4), that is

$$y_t(n) = y(n) - y_{id}(n) \quad \text{for } n \ge n_c \tag{6}$$

Although the output-switching method is ideal in the sense that no transient will occur, there will be a *discontinuity* in output signal $y_{id}(n)$ at time n_c. The interpretation of the discontinuity is that the values of output signal $y_{id}(n)$ at time instants $n_c - 1$ and n_c are results of different filtering processes. If the discontinuity is a problem, one should introduce smaller changes in the filter coefficients [1], [2] or crossfade the outputs of the two filters [7]—the transient cancellation method discussed subsequently in this paper will not help that problem by itself.

3. STATE-VARIABLE FORMULATION

3.1 Transient in the State Variables

A recursive filter can be expressed in state-variable form as

$$\mathbf{v}(n+1) = \mathbf{F}\mathbf{v}(n) + \mathbf{q}x(n) \tag{7a}$$

$$y(n) = \mathbf{g}^{\mathrm{T}}\mathbf{v}(n) + g_0 x(n) \tag{7b}$$

The dimensions and values of the matrices and vectors used in (7a) and (7b) depend on the realization structure of the filter. According to (7a), the state-variable vector $\mathbf{v}(n)$ can be expressed as a function of the input signal $x(n)$ and coefficient matrices when the coefficients have been changed at time n_c [8]

$$\mathbf{v}(n) = \begin{cases} \mathbf{F}_1^n \mathbf{v}(0) + \sum_{k=0}^{n-1} \mathbf{F}_1^{n-k-1}\mathbf{q}x(k), & 0 < n \le n_c \\ \mathbf{F}_2^{n-n_c}\mathbf{v}(n_c) + \sum_{k=n_c}^{n-1} \mathbf{F}_2^{n-k-1}\mathbf{q}x(k), & n > n_c \end{cases} \tag{8a, b}$$

where $\mathbf{v}(0)$ is the initial state of the filter, and $\mathbf{F}_1$ and $\mathbf{F}_2$ are the coefficient matrices before and after the coefficient change, respectively. In the following we assume that $n_c >> 0$ so that the decaying *initial transient* $\mathbf{F}_1^n \mathbf{v}(0)$ can be neglected. At the time of the change $(n = n_c)$, the state vector can be expressed as

$$\mathbf{v}(n_c) = \sum_{k=0}^{n_c-1} \mathbf{F}_1^{n_c-k-1}\mathbf{q}x(k) \tag{9}$$

and by substituting (9) into (8b) we obtain the state vector after the coefficients have been changed (that is, $n > n_c$)

$$\mathbf{v}(n) = \mathbf{F}_2^{n-n_c} \sum_{k=0}^{n_c-1} \mathbf{F}_1^{n_c-k-1}\mathbf{q}x(k) + \sum_{m=n_c}^{n-1} \mathbf{F}_2^{n-m-1}\mathbf{q}x(m) \tag{10}$$

This form can be elaborated in the following way to explicitly show the cause of the transient in the state vector

$$\mathbf{v}(n) = \mathbf{F}_2^{n-n_c}\Delta\mathbf{v}(n_c) + \sum_{k=0}^{n-1} \mathbf{F}_2^{n-k-1}\mathbf{q}x(k), \quad n > n_c \tag{11}$$

with

$$\Delta\mathbf{v}(n_c) = \sum_{k=0}^{n_c-1} \left(\mathbf{F}_1^{n_c-k-1} - \mathbf{F}_2^{n_c-k-1}\right)\mathbf{q}x(k) \tag{12}$$

The first term in (11) represents the *transient in the state vector* and the second term is the steady-state response of the filter to the input after the parameters have changed.

3.2 Zetterberg–Zhang Method

As stated by Zetterberg and Zhang [8], one way to completely eliminate the transient caused by the change of coefficients is to *subtract* term $\Delta\mathbf{v}(n_c)$ from the state vector at time $n = n_c$:

$$\begin{aligned} &\mathbf{v}(n_c) - \Delta\mathbf{v}(n_c) \\ &= \sum_{k=0}^{n_c-1} \mathbf{F}_1^{n_c-k-1}\mathbf{q}x(k) - \sum_{k=0}^{n_c-1} \left(\mathbf{F}_1^{n_c-k-1} - \mathbf{F}_2^{n_c-k-1}\right)\mathbf{q}x(k) \\ &= \sum_{k=0}^{n_c-1} \mathbf{F}_2^{n_c-k-1}\mathbf{q}x(k) \end{aligned} \tag{13}$$

This is the *Zetterberg–Zhang (ZZ) method* for the elimination of transients. The ZZ method implements the output-switching method introduced in Section 2. This is seen to be true since in (13) the subtraction of the correction term from the state vector effectively switches the state vector of coefficient set 1 to that of set 2, exactly as suggested by (4) where the state vectors of filters $H_1(z)$ and $H_2(z)$ are updated all the time. The drawbacks of the ZZ method are thus those of the output-switching method. Next we propose modifications to this method and introduce an efficient suppression method that does not have these problems.

4. THE NEW SUPPRESSION METHOD

4.1 Modifications to the ZZ Method

Equation (13) suggests that the ZZ method of transient suppression can equivalently be implemented by replacing the state vector (at the time of change from coefficient set 1 to 2) with the following *transient cancellation vector* (TCV)

$$\mathbf{v}_{\mathrm{tc}} = \sum_{k=0}^{n_c-1} \mathbf{F}_2^{n_c-k-1}\mathbf{q}x(k) \tag{14}$$

which simply contains the steady-state vector obtained when the coefficient matrix $\mathbf{F}_2$ is used from the beginning. As discussed above, it is impractical to compute the state vector for all filter coefficient sets all the time. Instead, we suggest approximating transient cancellation vector $\mathbf{v}_{\mathrm{tc}}$ with a *truncated sum* as [4], [5]

$$\hat{\mathbf{v}}_{\mathrm{tc}}(N_{\mathrm{a}}) = \sum_{k=n_c-N_a}^{n_c-1} \mathbf{F}_2^{n_c-k-1}\mathbf{q}x(k) = \sum_{k=1}^{N_a} \mathbf{F}_2^{k-1}\mathbf{q}x(n_c-k) \tag{15}$$

where parameter N_{a} is called the *advance time*. It expresses the number of samples of the input signal that are used for computing the state vector in advance of the coefficient change. If approximation $\hat{\mathbf{v}}_{\mathrm{tc}}$ is updated recursively (at the same sample rate) in parallel with the filtering operation, advance time N_{a} also represents the time lag (in samples) that is required before the estimate $\hat{\mathbf{v}}_{\mathrm{tc}}$ is available. The main advantage of this technique is that now the computation of the transient cancellation vector only takes finite time and need not be updated all the time in parallel with the filtering operation.

The use of a finite number of samples for computing $\hat{\mathbf{v}}_{\mathrm{tc}}$ is motivated by the fact that the impulse response of a stable recursive filter decays exponentially and can thus be regarded as finite-length. This implies that any input sample of the filter contributes to the state vector for a finite time. Thus, the knowledge of the *effective length of the impulse response from the input to the state vector* helps to estimate how many past input samples need to be taken into account in updating the transient cancellation vector (i.e., how many values of this impulse response are observably nonzero for the application). This principle may be applied to all recursive discrete-time filter structures.

4.2 Application to All-Pole and DF II Structures

Let us consider the application of the transient suppression method to all-pole and direct-form (DF) II recursive structures. A key observation is to understand how the contents of the state vector of these filters are produced. The state vector contains the N latest output samples, that is

$$\mathbf{v}(n) = \begin{bmatrix} y(n-1) & y(n-2) & \cdots & y(n-N) \end{bmatrix}^{\mathrm{T}} \tag{16}$$

On the other hand, the output signal $y(n)$ is the convolution of the impulse response of the filter with the input signal (see (3)). Thus, in the case of all-pole and DF II structures, it is necessary to determine the *effective length of the impulse response of the filter*, say N_P, to know how many past input samples effectively contribute to the first value of the state vector $v_1(n) = y(n-1)$. After N sample cycles, this value disappears from the state vector. Thus, the advance time may then be set equal to

$$N_{\mathrm{a}} = N_P + N \tag{17}$$

where N_P and N are the effective length of the impulse response and the order of the filter, respectively. This choice of N_{a} ensures that the updated state vector suffers sufficiently little from the truncation of the input signal in (15), according to the same criterion that was used to determine N_P. In practice, it is desirable to choose N_{a} to be the smallest integer that yields sufficient suppression, since this minimizes the implementation costs of the transient cancellation algorithm.

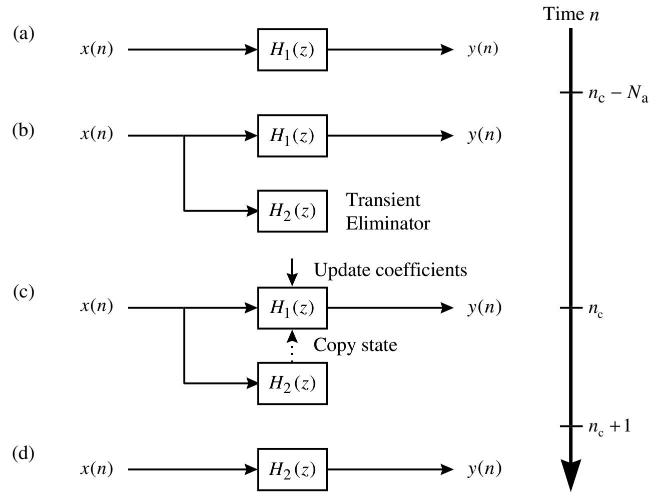


Figure 2. The different phases of the new transient suppression scheme for a single change of filter coefficients at time index $n = n_c$.

4.3 Implementation of the New Algorithm

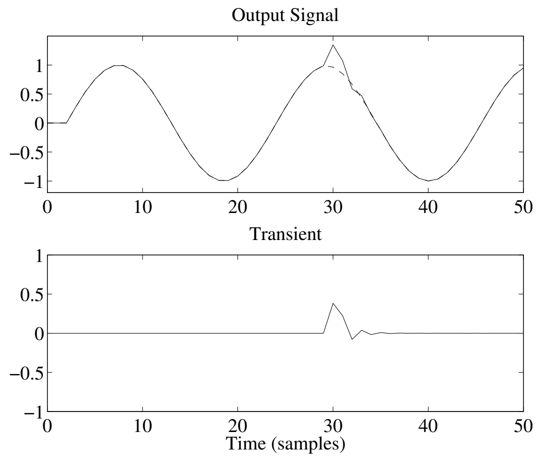
The transient elimination algorithm is implemented as depicted in Fig. 2. Initially, the IIR filter $H_1(z)$—called the signal filter—processes the input signal (Fig. 2(a)). N_{a} samples before the coefficient change, the input signal $x(n)$ is fed into two systems, filter $H_1(z)$ and the *transient eliminator* that has the new transfer function $H_2(z)$ (Fig. 2(b)). At time $n = n_c$, the coefficients of the signal filter are updated and the state vector (TCV) is copied from the transient eliminator to the signal filter's state as shown in Fig. 2(c). The transient eliminator is now turned off. Finally, the new coefficient set is used for filtering the input signal (Fig. 2(d)). As a result, the transient will be sufficiently suppressed if the value of parameter N_{a} is large enough.

It is seen that for a single coefficient change, the algorithm requires that two filters run in parallel for N_{a} sample intervals. Thus, when multiple changes are required and it is fast enough to update filter coefficients at every N_{a}th sample interval, there is no need to run more than two filters in parallel at any time.

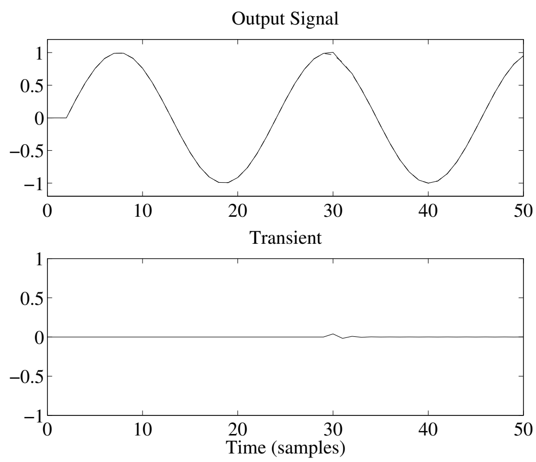
A major advantage of all-pole and DF II structures is that only the feedback coefficients affect the state vector. In the case of the DF II structure this implies that the suppression scheme only requires implementation of 1.5 filters at any time: the pole-zero signal filter and an all-pole transient eliminator.

5. EXAMPLE

We present an example that illustrates the transient suppression method. We filter a low-frequency sine wave (0.0454 times the sampling frequency f_s) with a second-order allpass filter (direct-form II) that approximates a constant group delay. Initially, the



(a)



(b)

Figure 3. The output signal waveform of a second-order allpass filter (DF II structure) when the coefficients are changed at time index n = 30 (a) with no transient elimination and (b) with the proposed transient suppresser with N_a = 4. The difference (transient) of the actual and ideal (dashed line) output signals is also displayed.

filter coefficients are $a_1 = 0$ and $a_2 = 0$ (corresponding to a constant delay of 2 samples) and at time index 30 they are changed to values $a_1 = 0.4$ and $a_2 = -0.028571$, which gives a group delay of 1.5 samples at low frequencies. We present the output and transient signals of the filter in two cases: without transient cancellation and when the cancellation method is used with parameter value $N_a = 4$. These output signals are compared with the "ideal" output signal (dashed line in Fig. 3) which has been computed using the output-switching method (see Fig. 1) by running two filters in parallel and changing the output at time n = 30. The transient signal shown in the lower part of the figures in both cases is the difference of the output signals of the time-varying and ideal filter. Obviously, in Fig. 3(b) (N_a = 4) the maximum amplitude of the transient has been suppressed with respect to Fig. 3(a). More suppression can be achieved by using a larger value for N_a. Other examples on the use of the algorithm are available in [4], [5], and [6].

6. CONCLUSIONS

A novel and efficient transient elimination technique for time-varying recursive filters was introduced. The technique updates the state variables at the time of the filter coefficient change. A finite number of input samples (described by the advance time parameter) is used for computing new values for the state vector. The advance time of the transient eliminator should be determined so that it will result in the required transient suppression at minimum implementation costs. The proposed transient elimination method can be used with all recursive digital filters. However, it can be most efficiently implemented when used with filter structures whose state contains the delayed output sample values, as in the case of all-pole or direct form II filter structures. Then the transient eliminator is an all-pole filter.

The new transient cancellation method is useful especially in real-time audio signal processing where the properties of recursive filters need to be changed while filtering a signal. Examples of such applications are music synthesis with physical models, auralization, and equalization.

7. ACKNOWLEDGMENTS

The authors wish to thank Dr. Jean–Marc Jot, Dr. Jonathan Mackenzie, Prof. Paavo Alku, and Prof. Matti Karjalainen for inspiring discussions on time-varying recursive filters.

8. REFERENCES

[1] J.-M. Jot, V. Larcher, and O. Warusfel, "Digital signal processing issues in the context of binaural and transaural stereophony," presented at the *98th AES Convention*, Paris, France, preprint no. 3980, Feb. 1995.

[2] 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 Signal Processing*, vol. 40, no. 7, pp. 452–455, July 1993.

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

[4] V. Välimäki, T. I. Laakso, and J. Mackenzie, "Elimination of transients in time-varying allpass fractional delay filters with application to digital waveguide modeling," in *Proc. Int. Computer Music Conf.*, Banff, AB, Canada, pp. 327–334, Sept. 1995.

[5] V. Välimäki, *Discrete-Time Modeling of Acoustic Tubes Using Fractional Delay Filters*. Dr. Tech. thesis. Espoo, Finland: Helsinki University of Technology, Laboratory of Acoustics and Audio Signal Processing, Dec. 1995. 193 p.

[6] V. Välimäki and T. I. Laakso, "Suppression of transients in variable recursive digital filters with a novel and efficient cancellation method," unpublished manuscript, Apr. 1997.

[7] S. A. Van Duyne, D. A. Jaffe, G. P. Scandalis, and T. S. Stilson, "A lossless, click-free, pitchbend-able delay line loop interpolation scheme," in *Proc. Int. Computer Music Conf.*, Thessaloniki, Greece, pp. 252–255, Sept. 1997.

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