IMPLEMENTATION OF RECURSIVE FILTERS HAVING DELAY FREE LOOPS

Aki Härmä

Helsinki University of Technology
Laboratory of Acoustics and Audio Signal Processing
P. O. Box 3000, 02015, Espoo, Finland
Aki.Harma@hut.fi

ABSTRACT

Certain types of recursive filters have been considered as non-realizable because they contain delayless recursive loops. Usually the problem is rather technical than theoretical. In this paper a method of implementing such filters is introduced. The general procedure is to split a delay free recursive filter to a non-delay free and a pure delay free structure. As a combination of these, the filter can be implemented directly and efficiently. In addition, following from the same formulation, a generic procedure to convert any such filter to an equivalent directly realizable structure is also given. As an example, a set of frequency warped all-pole filters is considered. The new warped all-pole lattice introduced in this paper completes the family of warped filters.

1. INTRODUCTION

The transfer function of a conventional FIR filter is given by

$$H(z) = 1 - \sum_{i=1}^{N} \alpha_i z^{-i}. \tag{1}$$

Its *inverse filter*, i.e., a recursive IIR filter has the form

$$G(z) = \frac{1}{H(z)} = \frac{1}{1 - \sum_i \alpha_i z^{-i}}. \tag{2}$$

The implementation of the filters is straightforward and determined by the two block diagrams represented in Fig. 1, where $A = z^{-1}$.

It is also possible to design transversal filters where $A \neq z^{-1}$. Several such filter structures have been presented in the past, e.g, in [6], [8]. In theory, any such transversal filter given by

$$H(z) = 1 - \sum_{i=1}^{N} \alpha_i A(z)^{-i}. \tag{3}$$

has an inverse filter

$$G(z) = \frac{1}{H(z)} = \frac{1}{1 - \sum_i \alpha_i A(z)^{-i}}. \tag{4}$$

The filter is always stable if the poles are inside a unit circle. However, problems may arise in trying to implement it if $A(z)$ contains delay free or *lag free* paths. A classical example is a *warped filter* [10][3], where

$$A(z) = \frac{z^{-1} - \lambda}{1 - \lambda z^{-1}}. \tag{5}$$

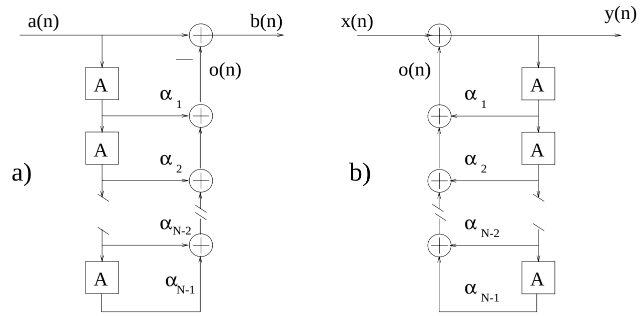

Figure 1: a.) A transversal FIR-type filter $H(z)$. b.) Its inverse filter $G(z)$

A direct implementation of (5) takes the following form

$$y(n) = x(n-1) + \lambda(y(n-1) - x(n)) \tag{6}$$

Applying this to Fig. 1b results in a structure with delayless paths, i.e., the signal $o(n)$ is needed to calculate $y(n)$, but $o(n)$ depends on the value of $y(n)$. By manipulating the difference equation of the filter it is possible to derive a new modified filter structure with a new set of coefficients (for a review, see [4]).

In this paper, a more general method of implementing certain filter structures with delay free loops without any modifications to the structure of the filter or its coefficients is introduced. In addition, a generic procedure to convert such a filter to a modified and directly realizable structure is also formulated.

Cancellation of delay free loops in digital filters have been studied extensively in the past, e.g., in [11], and especially in the case of digital filters derived from analog filters, e.g., recently in [9]. Those methods usually produce a new equivalent structure that can be implemented directly.

2. A GENERAL SOLUTION

A generalized version of a recursive filter discussed in this paper is shown in Fig. 2a. The method introduced in this paper separates the computation of the output of the filter $y(n)$ and updating of the inner states of the filter. During the computation of $y(n)$ the inner states are not updated.

2.0.1. Computation of $y(n)$

If P contains delay free paths the problem is that $o(n)$ is a function of $y(n)$ and *vice versa*. A solution is to divide the computation of

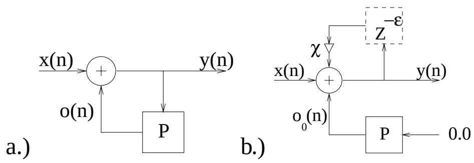

Figure 2: a.) A recursive delay free filter b.) equivalent circuit where $\epsilon \to 0$

$y(n)$ to two distinct steps. First, we may calculate an output of P so that P is temporarily disconnected from $y(n)$. In Fig. 2b this is denoted $o_0(n)$. A practical way to do that is to feed 0.0 into P and read its output. After that $y_0(n) = x(n) + o_0(n)$ is fed to a *pure delay free structure* (the upper loop in Fig. 2b). Pure delay free structure may be obtained from the original filter structure by disconnecting all its inner unit delay elements. Usually this reduces to a single coefficient χ.

The output of the disconnected part $o_0(n)$ depends only on the previous samples while the output of the pure delay free structure is a function of $x(n)$ and $o_0(n)$.

To derive a formula for $y(n)$ it is convenient to study first an approximation of the filter structure where there is a fractional delay element in the loop, i.e., in Fig. 2b, $\varepsilon = 1/M$, where M is a large positive integer. Equivalently one might say that the sampling rate in the *approximately delay free* loop is Mf_s, higher than that of the original structure. At time instant n the sum $y_0(n) = x(n) + o_0(n)$ is formed and fed to the delay free loop. At time instant $n + \varepsilon$ the output value is

$$y(n+\varepsilon) = y_0(n) + \chi y_0(n). \tag{7}$$

After $M - 1$ small time steps the output is

$$y(n+(M-1)\varepsilon) = y_0(n) + y_0(n)\sum_{i=1}^{M-1}\chi^i. \tag{8}$$

Now we immediately see that if $\varepsilon \to 0$, or $M \to \infty$ the final output $y(n)$ is a sum of a power series given by

$$y(n) = y_0(n)\sum_{i=0}^{\infty}\chi^i = \frac{y_0(n)}{1-\chi} = \frac{x(n)+o_0(n)}{1-\chi}. \tag{9}$$

2.0.2. Update the inner states

Once the current value of $y(n)$ is found the inner states of P must be updated. Usually intermediate results from computation of $o_0(n)$ and χ can be used efficiently at this phase.

We may now introduce the following algorithm as an implementation of any recursive filter:

Algorithm 1

1. At a given time step, compute $o_0(n)$ using the disconnected structure shown in Fig. 2b.
2. The value of χ may be computed in advance or obtained as an output for P, where the inner states are set to zeros, using 1.0 as an output. It is trivial to show that $|\chi| < 1$ and the sum in (9) converge if the filter is stable, i.e., the poles are within the unit circle.

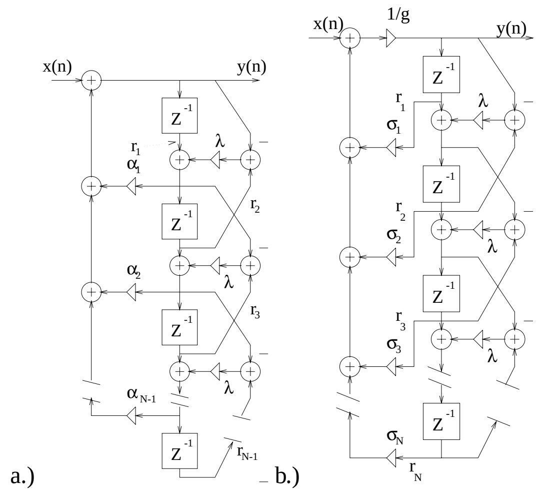

Figure 3: a.) A warped IIR structure b.) A modified and directly realizable warped IIR structure [2]

3. The final output $y(n)$ of the delay free recursive filter is now given by

$$y(n) = \frac{x(n)+o_0(n)}{1-\chi}. \tag{10}$$

4. Update the inner states of P using $y(n)$.
5. $n \to n+1$ and go to step 1.

3. APPLICATIONS

3.1. Warped all-pole filters

The block diagram of a warped all-pole filter is shown in Fig. 3a. The outputs of the unit delay elements are denoted by r_i. Fig. 3b shows the modified structure first proposed in [5] and later, independently, in [4].

The σ_i-coefficients and $1/g$ of the modified structure or the output $o_0(n)$ may be calculated using the following algorithm [4]:

Algorithm 2

1. $\sigma_{N+1} = \lambda\alpha_N$; $S_N = \alpha_N$; $o_0(n) = \sigma_{N+1}r_{N+1}$
2. for $i = N, N-1, \cdots, 2$,

$$S_{i-1} = \alpha_{i-1} - \lambda S_i;$$

$$\sigma_i = \lambda S_{i-1} + S_i;\; o_0(n) = o_0(n) + \sigma_i r_i;$$

   end

3. $\sigma_1 = S_1$; $1/g = \sigma_0 = 1 - \lambda S_1$; $o_0(n) = o_0(n) + \sigma_1 r1$;

The direct computation of the value of $o_0(n)$ using step 1 in algorithm 1 takes the form:

$$o_0(n) = \alpha_1(r_1 + \lambda r_2) + \alpha_2(r_2 + \lambda r_3 - \lambda(r_1 + \lambda r_2)) \tag{11}$$
$$+\alpha_3(r_3 + \lambda r_4 - \lambda(r_2 + \lambda r_3 - \lambda(r_1 + \lambda r_2))) + \cdots$$

Notice that if the equation is collected in the form: $o_0(n) = \sigma_1 r_1 + \sigma_2 r_2 + \sigma_3 r_3 + \cdots$ the result is exactly the same modified IIR structure as above and the coefficients are the same as those given by the algorithm 2.

Equation (11) may also be computed recursively by the following algorithm:

Algorithm 3

1. $S_1 = r_1 + \lambda r_2$; $o_0(n) = \alpha_1 S_1$;
2. for $i = 2, 3, 4, \cdots$

$$S_i = r_i + \lambda(r_{i+1} - S_{i-1});$$

$$o_0(n) = o_0(n) + \alpha_i S_i;$$

   end

If coefficients α_i of the system and parameters λ do not change often it is convenient to calculate the value of χ in advance. In this structure it takes the following form:

$$\chi = \sum_{i=1}^{N} \alpha_i(-\lambda)^i. \tag{12}$$

Obviously, the term $1/(1-\chi)$ takes exactly the same form as the $1/g$ term used in the modified structure.

If the coefficients are not updated at each sample the most efficient implementation is the use of the modified and directly realizable structure (Fig. 3b), where all the coefficients are calculated in advance. Numerical simulations (using MATLAB) by the author showed that in using the algorithm 1, the number of floating point operations per one filter stage per sample were 11 FLOPS [1]. In the modified filter (Fig. 3b), where all the coefficients were computed in advance it was 6.5 FLOPS. However, if the coefficients are updated at each sample the amount of FLOPS in the modified filter was 13.2, while in the algorithm 1 it was 13 FLOPS. This suggests that in this case, the use of the algorithm 1 is justified only if the coefficients of the filter change continuously.

3.2. Warped all-pole lattice

A warped IIR lattice may be represented in the same form as a general recursive filter of Fig. 2a by using a structure called *warped lattice predictor*, where the unit delay elements of the conventional structure are replaced by first order allpass elements given by (5). The author is not aware of any implementation of a warped IIR lattice prior to the following formulation.

The block diagram of a warped recursive lattice is shown in Fig. 4a. The k_i coefficients are reflection coefficients that can be derived from α_i coefficients of the corresponding transversal filter using a simple recursion (see e.g., [7].)

Denoting $\nu = 1 - \lambda^2$, the first step in algorithm 1 is now

$$\begin{aligned} o_0(n) = k_1(\nu r_1) + k_2(\nu r_2 - \lambda(\nu r_1)) \quad (13)\\ +k_3(\nu r_3 - \lambda(\nu r_2 - \lambda(\nu r_1) + k_2 k_1(\nu r_1)))\\ +k_4(\nu r_4 - \lambda(\nu r_3 - \lambda(\nu r_2 - \lambda(\nu r_1) + k_2 k_1(\nu r_1))\\ +k_3 k_1(\nu r_1)) + k_3 k_2(\nu r_2 - \lambda(\nu r_1))) \cdots \end{aligned}$$

From (13), an efficient implementation is given by the following algorithm:

Algorithm 4

1. $S_i = 0\,\forall i$ and $k_i, p_i = 0\,\forall i \leq 0$
2. for $i = 1, 2, 3, \cdots, M$

$$S_i = \nu r_3 - \lambda(S_{i-1} + k_{i-1} p_{i-2});$$

$$o_0(n) = o_0(n) + k_i S_i;\ p_i = o_0(n);$$

   end

[1] χ was calculated in advance

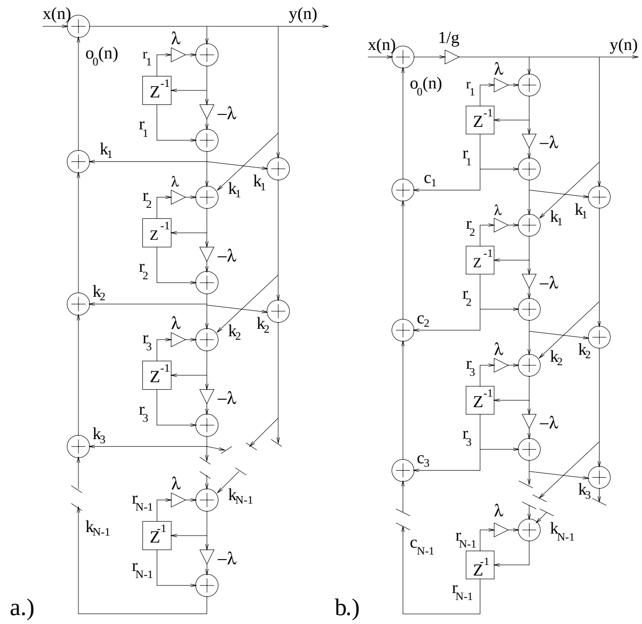

Figure 4: a.) A warped all-pole lattice structure as a combination of a lattice predictor and a recursive feedback loop b.) A modified lattice structure

As above, a directly realizable structure (Fig. 4b) is available. If (13) is reorganized in terms of the signals r_i as was done in deriving the modified structure of Fig. 3b, the resulting equation for $o_0(n)$ becomes very complicated because the lattice structure contains significantly more delay free paths. However, we may use lattice-type computation for the coefficients c_i of the new structure. The algorithm may be understood so that the input to the lattice structure is first disconnected from $y(n)$. Next, set for each c_i, $r_i = 1$ and all the other signals $r_{j \neq i} = 0$. After that, 0.0 is fed into the lattice structure and as a result $c_i = o_0(n)$.

As an algorithm, this may be reduced to

Algorithm 5

1. for $j = 1, 2, 3, \cdots, M$
2. $f = 0$; $b = 0$; $c_j = 0$; $s = \nu r_k$;
3. for $i = j, j+1, \cdots, M$

$$c_j = c_j + k_i s;$$

$$b = s + k_i f$$

$$f = f + k_i s;$$

$$s = -\lambda b;$$

   end

4. end

The χ coefficient may be calculated using a similar approach. Descriptively, according to the step 2 in algorithm 1, the inner states are set to zeros and 1.0 is fed into P. As a result, $\chi = o_0(n)$. An efficient computation of χ takes the following steps:

Algorithm 6

1. $b = 1.0$; $f = 1.0$; $\chi = 0$; $s = -\lambda$;

2. for $i = 1, 2, 3, \cdots, M$

$$\chi = \chi + k_i s;$$
$$b = s + k_i f$$
$$f = f + k_i s;$$
$$s = -\lambda b;$$

end

Numerical simulations show that an optimized implementation of the system using algorithm 1, so that χ is computed offline, requires approximately 13 floating point operations per sample per filter stage. In the modified lattice structure, where the coefficients c_i and χ are computed beforehand, the computational load is 12 FLOPS. If the coefficients are updated continuously, the corresponding numbers of FLOPS for algorithm 1 and the modified lattice structure of Fig. 4b are 23 and 82.3–130 FLOPS [2], respectively. Therefore, the modified structure of Fig. 4b is more efficient than the algorithm 1 only if the coefficients are held constant over several hundred sample periods.

4. MODIFIED AND DIRECTLY REALIZABLE STRUCTURES

From the two examples above, we may now formulate a generic procedure to design a modified and directly realizable structure and its coefficients from any filter having the structure of Fig. 2a and containing delay free loops. As a difference equation the modified structure takes the following form:

$$y(n) = \frac{1}{1-\chi}(x(n) + \sum_{i=1}^{M} c_i r_i), \qquad (14)$$

where χ is the output of system P so that all the inner states of system P are set to zero and 1.0 is fed into the filter.

The signals r_i are selected so that they are outputs of unit delay elements of the structure and do not depend on the passing value of $y(n)$. Coefficient c_i is calculated by first setting $r_i = 1$ and all the other signals $r_{j \neq i} = 0$. After that coefficient c_i is given as an output of P for a zero input. This is repeated for every coefficient c_i.

5. CONCLUSIONS AND FUTURE WORK

A method of implementing any recursive filter having a delayless feedback loop was introduced in this paper. It was also shown that the same formulation of the problem may be used to characterize a technique of converting any recursive filter to a new modified filter that can be implemented directly.

In this paper, the technique was applied to the implementation of frequency warped all-pole filters, i.e., filters where the unit delays of a conventional structure are replaced with first order allpass elements. In the case of a direct form 1 filter an alternative method of implementing the filter was found. The warped all-pole lattice and the corresponding modified filter are new usable alternatives for applications where warped all-pole filters are used.

The author has been working with new techniques of audio coding based on *warped linear prediction* [10][1] and complex valued warped linear prediction [2]. The warped all-pole lattice introduced in this paper is going to have a central role in developing and optimizing the coding scheme because with the new lattice structure it is now possible to apply several efficient techniques developed for conventional linear predictive coding, e.g., backward adaptive lattice, to warped linear predictive coding.

6. ACKNOWLEDGMENT

This work has been partially supported by the Academy of Finland. The author is grateful to Professor M. Karjalainen, U. K. Laine, and V. Välimäki for useful discussions and help in developing this text.

7. REFERENCES

[1] A Härmä, U K Laine, and M Karjalainen. Warped linear prediction in audio coding. In *Proc. NORSIG'96*, Helsinki, 1996.

[2] A Härmä, U K Laine, and M Karjalainen. An experimental audio codec based on warped linear prediction of complex valued signals. In *ICASSP 97*, volume 1, pages 323–327, Munich, 1997.

[3] M Karjalainen, A Härmä, J Huopaniemi, and U K Laine. Warped filters and their audio applications. In *IEEE Workshop on ASPAA 97*, Mohonk, New York, 1997.

[4] M Karjalainen, A Härmä, and U K Laine. Realizable warped IIR filters and their properties. In *ICASSP'97*, volume 3, pages 2205–2209, Munich, 1997.

[5] T Kobayashi, S Imai, and Y Fukuda. Mel-generalized log spectral approximation filter. *Trans. IECE*, 68:610–611, 1985.

[6] Y W Lee. *Statistical theory of communication*. Wiley, New York, 1960.

[7] J G Proakis and D G Manolakis. *Digital Signal Processing: Principles, Algorithms, and Applications*. Macmillan, 2 edition, 1992.

[8] W Schüssler. Variable digital filters. *Arch. Elek. Übertragung*, 24:524–525, 1970.

[9] S Signell and L Harnefors. Analysis of bilinear ladder filters. In *Proc. of ISCAS 1995*, volume 1, pages 69–72, Seattle, 1995. IEEE.

[10] H W Strube. Linear prediction on a warped frequency scale. *J. of the Acoust. Soc. Am.*, 68(4):1071–1076, 1980.

[11] J Szczupak and S K Mitra. Detection, location, and removal of delay-free loops in digital filter configurations. *IEEE Trans. on ASSP*, 23(6):558–562, 1975.

[2] In the modified structure the computational load is a function filter dimension. The numbers correspond to 10- to 20-tap lattice filters