

New Ladder Filters Based on Wave Adapters

Jean H.F. Ritzerfeld

Technische Universiteit Eindhoven, Fac. Elektrotechniek

P.O. Box 513, 5600 MB Eindhoven, The Netherlands

Phone: +31 (0)40 247 3252 Fax: +31 (0)40 246 6508

E-mail: j.h.f.ritzerfeld@tue.nl

Abstract— In digital IIR filter design it is no easy matter to combine the demand of using the minimum number of multipliers (canonical design) with the demand of minimizing the effects of a finite wordlength (design for low noise, low sensitivity and freedom from limit cycles). Especially for higher order filters there are no easy solutions. Recent work [1] provided a sub-optimal solution that has low noise/sensitivity but is not free from limit cycles. Classical solutions like lattice-ladder filters [2] and wave digital filters [3], [4] can combine both demands, but do require some specialized knowledge on recursive ladders and analog filter design. Moreover, these classical solutions are not straightforward in terms of a state space description and both exhibit parameters that depend on all recursive multipliers.

In this paper a new canonical ladder filter design is given, where the ladder sections are not two-pair lattices (as in [2]) but are based on three-pair wave adapters (introduced in [4]). A simple Matlab program is provided that determines the ladder coefficients (the feedforward as well as the recursive multipliers), the state space description (scaled as well as unscaled) and the transformation matrix that generates this description via a similarity transform of the standard Direct Form II implementation of a general n -th order transfer function. It is shown quantitatively how close this ladder is to the optimal low noise solution that uses $(n+1)^2$ multipliers instead of the canonical $2n+1$. Design examples of narrow-band IIR low-pass filters (up to 12th order) demonstrate that sensitivity and noise are within 6dB of optimal and are independent of the cut-off frequency.

Keywords— Filter Design, Lattice Filters, Ladder Filters, Canonical, Wave Digital Filters.

I. INTRODUCTION

Lattice-ladder filters [2] and wave digital filters [3] both use the concept of losslessness, borrowed from analog filter design, to build digital filters that have low coefficient sensitivity when multiplier values are quantized. Low sensitivity also means low noise when signals are quantized (in the recursive loops of IIR filters) and the lossless property of the filter building blocks guarantees freedom from limit cycles and overflow oscillations (when energy decreasing signal quantization is used such as magnitude truncation). The losslessness of a two-pair digital filter

building block is thereby defined as

$$\mathbf{S}^T(z^{-1}) \mathbf{D} \mathbf{S}(z) = \mathbf{D}, \quad (1)$$

where \mathbf{S} is the 2×2 (scattering) matrix of the two-pair that relates its two output signals Y_1 and Y_2 to its two input signals U_1 and U_2 in matrix form $\mathbf{Y} = \mathbf{S} \mathbf{U}$ with column vectors \mathbf{Y} and \mathbf{U} , and \mathbf{D} is a positive diagonal matrix.

For example, the two-pair building block for a classical lattice-ladder filter is drawn in Fig. 1 and has a scattering matrix

$$\mathbf{S} = \begin{pmatrix} k & (1-k)z^{-1} \\ 1+k & -kz^{-1} \end{pmatrix}, \quad (2)$$

where the box with parameter k implements the relations

$$\begin{aligned} Y_1 &= k(U_1 - U'_2) + U'_2 \\ Y_2 &= k(U_1 - U'_2) + U_1 \end{aligned} \quad (3)$$

and is called a one-multiplier lattice, since (3) can be realized with a single multiplier k . Note that \mathbf{S} satisfies (1) with $\mathbf{D} = \begin{pmatrix} 1+k & 0 \\ 0 & 1-k \end{pmatrix}$, which is positive for $|k| < 1$.

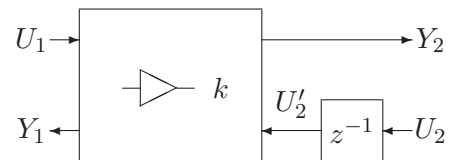


Fig. 1. Two-pair building block for ladder filter design

Incidentally, [2] also mentions the two-multiplier lattice element with

$$\mathbf{S} = \begin{pmatrix} k & 1-k^2 \\ 1 & -k \end{pmatrix} \quad \text{and} \quad \mathbf{D} = \begin{pmatrix} 1 & 0 \\ 0 & 1-k^2 \end{pmatrix}, \quad (4)$$

and even a four-multiplier two-pair with

$$\mathbf{S} = \begin{pmatrix} k & \sqrt{1-k^2} \\ \sqrt{1-k^2} & -k \end{pmatrix} \quad \text{and} \quad \mathbf{D} = \mathbf{I}, \quad (5)$$

but since we want to build canonical digital filters that use the minimum amount of multipliers we will not look into these lossless alternatives in this paper.

II. STATE SPACE DESCRIPTION OF LATTICE-LADDERS

In conventional lattice-ladder filter design n building blocks as in Fig. 1 are connected in a chain, or ladder, where the rightmost two-pair is closed off with a through connection ($U_2 = Y_2$) to create the n -th order denominator of an IIR transfer function. A general n -th order numerator is then realized by a linear combination of the n signals U_2 and the (overall) output signal, i.e. the output Y_1 of the leftmost two-pair. Here, we will deviate slightly from this conventional ladder and build the filter as in Fig. 2 where the delays are in the upper (left to right) row of the chain instead of the lower, and the numerator is created in the standard fashion by linear combination of input signal and state variables. We will get back to the conventional ladder at the end of this section.

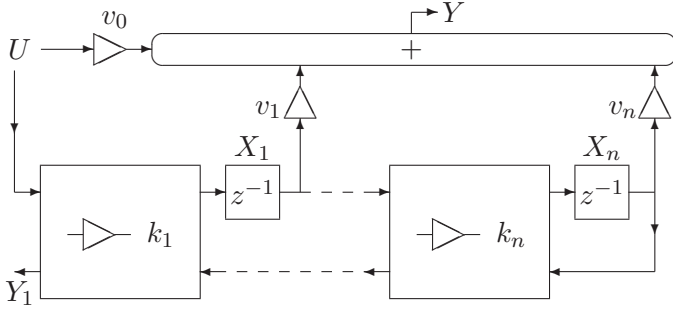


Fig. 2. Implementation of an n -th order lattice-ladder filter

The matrix \mathbf{S} of the building blocks in Fig. 2 is given by

$$\mathbf{S} = \begin{pmatrix} k & 1 - k \\ (1 + k)z^{-1} & -kz^{-1} \end{pmatrix}, \quad (6)$$

slightly different from (2). In order to do calculations on the ladder, however, it is more useful to describe the two-pair with its chain matrix \mathbf{R} :

$$\begin{pmatrix} U_1 \\ Y_1 \end{pmatrix} = \frac{1}{1+k} \begin{pmatrix} z & k \\ kz & 1 \end{pmatrix} \begin{pmatrix} Y_2 \\ U_2 \end{pmatrix} = \mathbf{R} \begin{pmatrix} Y_2 \\ U_2 \end{pmatrix}. \quad (7)$$

For example, in the second-order case we have

$$\begin{pmatrix} U \\ Y_1 \end{pmatrix} = \frac{1}{(1+k_1)(1+k_2)} \begin{pmatrix} z & k_1 \\ k_1z & 1 \end{pmatrix} \begin{pmatrix} z & k_2 \\ k_2z & 1 \end{pmatrix} \begin{pmatrix} X_2 \\ X_2 \end{pmatrix}, \quad (8)$$

from which it follows that

$$\begin{aligned} \frac{Y_1}{U} &= \frac{k_1z^2 + k_2(1+k_1)z + 1}{z^2 + k_2(1+k_1)z + k_1} \\ \frac{X_2}{U} &= \frac{(1+k_1)(1+k_2)}{z^2 + k_2(1+k_1)z + k_1}. \end{aligned} \quad (9)$$

We see that Y_1/U is all-pass and X_2/U is all-pole. For any n , X_n/U is all-pole with numerator $(1 + k_1) \cdots (1 + k_n)$ and Y_1/U is necessarily all-pass, since the overall transfer must be lossless, i.e. $H_1(z)H_1(z^{-1}) = 1$.

The denominator coefficient vector $(1, k_2(1 + k_1), k_1)$ for the second-order case is found from the matrix product

$$(1 \quad k_2) \begin{pmatrix} 1 & 0 & k_1 \\ 0 & 1 + k_1 & 0 \end{pmatrix}, \quad (10)$$

which can be written as

$$(\mathbf{I}_{12} + k_2\mathbf{J}_{12})(\mathbf{I}_{23} + k_1\mathbf{J}_{23}), \quad (11)$$

where \mathbf{I}_{ij} is an $i \times j$ matrix with ones on the diagonal and zeros elsewhere, and likewise for \mathbf{J} with ones on the cross diagonal (through the corner furthest from upper left). For any n , the denominator coefficient vector is given by

$$\mathbf{a}^{(n)} = (\mathbf{I}_{12} + k_n\mathbf{J}_{12}) \cdots (\mathbf{I}_{n,n+1} + k_1\mathbf{J}_{n,n+1}), \quad (12)$$

where the superscript reminds us that this is the n -th order coefficient vector, subscripted 0 to n , where $\mathbf{a}_0^{(n)} = 1$.

From (12), we know the overall denominator given the lattice coefficients k_i . For design, we of course want to do the opposite, i.e. find k_i from $\mathbf{a}^{(n)}$. Initializing $k_1 = \mathbf{a}_n^{(n)}$, this can be done recursively by successively equating k_{i+1} to the highest coefficient of $\mathbf{a}^{(n-i)}$, the denominator that remains after taking away successive sections of the ladder (from left to right). The recursion is given by

$$\mathbf{a}^{(n-i)} = \mathbf{a}^{(n-i+1)} \frac{\mathbf{I}_{n-i+2, n-i+1} - k_i\mathbf{J}_{n-i+2, n-i+1}}{1 - k_i^2}, \quad (13)$$

holding for $i = 1, \dots, n$, where $\mathbf{a}^{(0)} = 1$. It can easily be checked that (13) is correct, since for example (take $i = 1$)

$$(\mathbf{I}_{n, n+1} + k_1\mathbf{J}_{n, n+1})(\mathbf{I}_{n+1, n} - k_1\mathbf{J}_{n+1, n}) = (1 - k_1^2)\mathbf{I}_{nn}, \quad (14)$$

where \mathbf{I}_{nn} is the identity matrix (of dimension $n \times n$). Note that since $k_{i+1} = \mathbf{a}_{n-i}^{(n-i)}$, which is the product of the roots of denominator $\mathbf{a}^{(n-i)}$, all lattice coefficients must satisfy $|k_i| < 1$ for stability, a condition that was also necessary for losslessness.

As a matter of interest, we mention that (12) can also be inverted explicitly (i.e. non-recursively) in closed form, such that k_i is expressed in the denominator coefficients $a_j = \mathbf{a}_j^{(n)}$ only. The result is found to be ($i = 0, \dots, n-1$)

$$k_{i+1} = \frac{\Delta_{i-1}^- \Delta_{i+1}^+ - \Delta_{i-1}^+ \Delta_{i+1}^-}{\Delta_{i-1}^- \Delta_{i+1}^+ + \Delta_{i-1}^+ \Delta_{i+1}^-}, \quad (15)$$

where $\Delta_{-1}^+ = \Delta_{-1}^- = \Delta_0^+ = \Delta_0^- = 1$, per definition, and where Δ_i^+ and Δ_i^- are the determinants of the matrices

$$\begin{bmatrix} a_0 & 0 & \cdots & 0 \\ a_1 & a_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{i-1} & a_{i-2} & \cdots & a_0 \end{bmatrix} \pm \begin{bmatrix} a_{n-i+1} & a_{n-i+2} & \cdots & a_n \\ \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & a_n & \cdots & 0 \\ a_n & 0 & \cdots & 0 \end{bmatrix}. \quad (16)$$

Although one would never calculate the lattice coefficients using (15), it is good to know that such closed form expressions exist and that the stability condition $|k_i| < 1$ follows directly from the conditions on the determinants Δ_i^\pm given in [5] to ensure that the roots of the polynomial with coefficient vector \mathbf{a} are within the unit circle. Specifically,

$$1 - k_{i+1}^2 = \frac{\Delta_{i-1}^- \Delta_{i+1}^+ \Delta_{i-1}^+ \Delta_{i+1}^-}{(\Delta_i^- \Delta_i^+)^2} > 0, \quad (17)$$

since $\Delta_i^- \Delta_i^+ > 0$ is necessary for stability, cf. [5].

In order to find the state description of the lattice-ladder, we look at the transfer functions X_i/U from input to state variables. The denominators of these are of course given by $\mathbf{a}^{(n)}$ and the numerator coefficient vectors are simply equal to $\mathbf{a}^{(n-i)}$ times a factor $(1+k_1) \cdots (1+k_i)$. If we lengthen these n vectors with $i-1$ leading zeros and put them as rows in an $n \times n$ matrix, we will have found the transformation matrix \mathbf{T} for a similarity transform of the lattice-ladder to Direct Form II. For example, for $n=2$

$$\mathbf{T} = \begin{pmatrix} 1 & k_2 \\ 0 & 1 \end{pmatrix} \cdot (1+k_1) \cdot (1+k_2)(1+k_1), \quad (18)$$

where the factors per row are written to the right of the matrix for clarity and to save space. For $n=3$ we have

$$\mathbf{T} = \begin{pmatrix} 1 & k_3(1+k_2) & k_2 \\ 0 & 1 & k_3 \\ 0 & 0 & 1 \end{pmatrix} \cdot (1+k_1) \cdot (1+k_2)(1+k_1) \cdot (1+k_3)(1+k_2)(1+k_1) \quad (19)$$

The state matrix of the lattice-ladder is then found from $\mathbf{A} = \mathbf{T}\mathbf{A}_d\mathbf{T}^{-1}$, where \mathbf{A}_d is the state matrix of Direct Form II which has the coefficients $-a_1$ to $-a_n$ on the first row. So the second-order lattice-ladder has a state matrix

$$\mathbf{T} \begin{pmatrix} -k_2(1+k_1) & -k_1 \\ 1 & 0 \end{pmatrix} \mathbf{T}^{-1} = \begin{pmatrix} -k_1 k_2 & -k_1(1-k_2) \\ 1+k_2 & -k_2 \end{pmatrix}, \quad (20)$$

as indeed can be checked directly from Fig. 2. For $n=3$

$$\mathbf{A} = \begin{pmatrix} -k_1 k_2 & -k_1(1-k_2)k_3 & -k_1(1-k_2)(1-k_3) \\ 1+k_2 & -k_2 k_3 & -k_2(1-k_3) \\ 0 & 1+k_3 & -k_3 \end{pmatrix}, \quad (21)$$

and for any n the logical structure of \mathbf{A} with respect to the lattice coefficients k_i can be continued as above. Note that \mathbf{A} is upper triangular apart from a single diagonal below the principal one with entries $1+k_2, \dots, 1+k_n$. Given \mathbf{T} , the \mathbf{B} , \mathbf{C} and \mathbf{D} matrices of the state space description are also easy to determine from Direct Form II as

$$\mathbf{B} = \mathbf{T}\mathbf{B}_d, \quad \mathbf{C} = \mathbf{C}_d\mathbf{T}^{-1}, \quad \mathbf{D} = \mathbf{D}_d, \quad (22)$$

where \mathbf{B}_d is the unit vector \mathbf{I}_{n1} and \mathbf{C}_d and \mathbf{D}_d follow directly from the numerator of the transfer function to be implemented. Note that $\mathbf{B} = (1+k_1)\mathbf{I}_{n1}$, the first column of \mathbf{T} , and $[\mathbf{D} \ \mathbf{C}] = \mathbf{v}$, the vector of multipliers v_0, \dots, v_n to create the output signal in Fig. 2. Incidentally, if we extend \mathbf{T} with a first row $\mathbf{a}^{(n)}$ to create an $(n+1) \times (n+1)$ matrix \mathbf{V} , we can also write more concisely

$$\mathbf{v} = \mathbf{b}^{(n)}\mathbf{V}^{-1}, \quad (23)$$

where $\mathbf{b}^{(n)}$ is the numerator coefficient vector of the transfer function to be implemented. For example, for $n=2$

$$\mathbf{V} = \begin{pmatrix} 1 & k_2(1+k_1) & k_1 \\ 0 & 1 & k_2 \\ 0 & 0 & 1 \end{pmatrix} \cdot (1+k_1) \cdot (1+k_2)(1+k_1) \quad (24)$$

Having found \mathbf{k} from (13), with $k_{i+1} = a_{n-i}^{(n-i)}$, and \mathbf{v} from (23), the design of the lattice-ladder is complete.

Returning to the conventional lattice-ladder design mentioned in the first paragraph of this section, we note that in this case not only are the delays in the lower row of the chain in Fig. 2, but also that the state variables X_i , the lattice coefficients k_i and the output multipliers v_i are numbered from right to left. As a result, the matrices \mathbf{T} and \mathbf{V} are rotated over 180 degrees, but the state description and output vector \mathbf{v} are still found from (22) and (23). Note that \mathbf{A} will still be upper triangular apart from a single diagonal below the principal one (but now with entries $1-k_1, \dots, 1-k_{n-1}$) and that \mathbf{B} will no longer be a factor times a unit vector. For example, in the second-order case

$$\mathbf{A} = \begin{pmatrix} -k_1 & -k_2(1+k_1) \\ 1-k_1 & -k_1 k_2 \end{pmatrix}, \quad \mathbf{B} = (1+k_2) \begin{pmatrix} 1+k_1 \\ k_1 \end{pmatrix}. \quad (25)$$

In passing we mention that the ladder with two-multiplier lattices cf. (4), often used in linear prediction, has matrices \mathbf{T} and \mathbf{V} without the factors per row as in (18) and (24), or rotated versions thereof in the conventional case.

In all, we can write the following Matlab function that has as input arguments the numerator and denominator coefficient vectors \mathbf{b} and \mathbf{a} of the desired transfer function and as output arguments the state description of the lattice-ladder and the vectors of multipliers \mathbf{k} and \mathbf{v} .

```

function [A,B,C,D,k,v,p,T]=lattice1(b,a,nb,s)
%
% As optional input arguments one can specify the number of bits (nb)
% to be used for the multiplier values, and choose the string variable
% s to be '1' (for the conventional one-multiplier lattice-ladder),
% '2' (for the conventional two-multiplier lattice-ladder), and/or any
% combination of 'p' (for a plot of the amplitude response with and
% without coefficient quantization), 'o' (for the value of the optimal
% noise gain with signal quantization, given a transfer function b,a),
% 'n' (for the value of the best feasible noise gain of the lattice-
% ladder after scaling) and 's' (to actually scale the implemented
% lattice-ladder with power-of-two scaling multipliers (shifts over
% p bits), and for the value of the corresponding noise gain).
%
n=length(a)-1; k=zeros(1,n); K=zeros(n,n); K2=zeros(n,n);
[H,w]=freqz(b,a,4096); V=zeros(n+1); P=zeros(n+1); P(1,:)=a;
b=[b,zeros(1,n+1-length(b))]; if nargin<4, s=''; end
for i=1:n
    k(i)=P(i,n+1); I=eye(n-i+2,n-i+1); J=flipud(I);
    P(i+1,i+1:n+1)=P(i,i:n+1)*(I-k(i)*J)/(1-k(i)^2);
end
if nargin>2, k=2^-(nb-1)*round(2^(nb-1)*k); end
for i=n:-1:1
    I=eye(n-i+1,n-i+2); J=flipr(I);
    P(i,i:n+1)=P(i+1,i+1:n+1)*(I+k(i)*J);
    V(i+1,:)=P(i+1,:)*prod(1+k(1:i));
    K(i,i)=prod(1+k(1:i))/prod(1-k(1:i));
    K2(i,i)=1/prod(1-k(1:i).^2);
end
p=ceil(log2(diag(K)')/2); p2=ceil(log2(diag(K2)')/2);
if findstr(s,'s'), V=diag(2.^[0,-p])*V; P=diag(2.^[0,-p2])*P; end
T=V(2:n+1,2:n+1); aq=P(1,:); V(1,:)=aq;
if findstr(s,'1'), k=flipr(k); V=rot90(V,2); T=V(1:n,1:n); end
if findstr(s,'2'), k=flipr(k); V=rot90(P,2); T=V(1:n,1:n); p=p2; end
v=b/V; if nargin>2, v=2^-(nb-1)*round(2^(nb-1)*v); end
bq=v*V; Hq=freqz(bq,aq,4096); [A,B,C,D]=tf2ss(bq,aq);
A=T*A/T; B=T*B; C=C/T; for i=3:n, A(i,1:i-2)=0; end
K=diag(diag(dgram(A,B))); W=dgram(A',C');
if findstr(s,'p'), plot(w/pi,20*log10(abs([H,Hq]))); grid; end
if findstr(s,'o'), Go=sum(sqrt(eig(K*W)))/2/n; display(Go); end
if findstr(s,'n'), Gn=diag(K)*diag(W); display(Gn); end
if findstr(s,'s'), Gs=sum(diag(W)); display(Gs); end

```

III. LADDER DESIGN BASED ON WAVE ADAPTERS

Although the lattice-ladder is canonical, limit cycle stable and low noise, it has the disadvantage that the entries of its state matrix are products of several recursive multipliers (A_{1n} and $A_{1,n-1}$ even depend on all n lattice coefficients). This can be a problem in time-critical applications, since intermediate results on non-state nodes are only found sequentially in time and it may therefore take too long for the new state, especially $x_1[k+1]$, to compute. The problem can be alleviated by a factor 2 if we were to use second-order two-pairs, but still lossless and canonical, as sections in the ladder. One way to create a second-order two-pair is drawn in Fig. 3, where the delays are intrinsically lossless and the delay-free three-pair should satisfy $S^T D S = D$. So can the three-pair, which must now necessarily have two degrees of freedom, be implemented with only two multipliers and still meet the condition for losslessness?

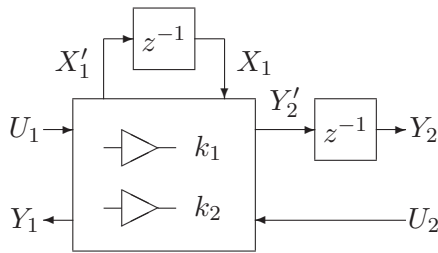


Fig. 3. Second-order two-pair section for ladder filter design

In a Wave Digital Filter (WDF) [3] a port, or signal-pair, can only be connected to another port via so-called wave adapters which realize the series or parallel connection of the components of the analog lossless two-port from which the WDF is derived. Fettweis introduces m -pair series and parallel adapters which are delay-free and contain $m-1$ multipliers that are directly related to the values of the analog components they connect. These adapters are canonical and intrinsically lossless. For example, a three-pair parallel adapter has a scattering matrix

$$S = \begin{pmatrix} \alpha_1 - 1 & \alpha_2 & \alpha_3 \\ \alpha_1 & \alpha_2 - 1 & \alpha_3 \\ \alpha_1 & \alpha_2 & \alpha_3 - 1 \end{pmatrix}, \quad (26)$$

with $\alpha_1 + \alpha_2 + \alpha_3 = 2$ for it to have only two degrees of freedom and $\alpha_i > 0$ for stability and losslessness. For our purposes we redefine the α 's for S to more resemble the scattering matrix of the lattice two-pair element. Here, the three-pair of Fig. 3 is defined with

$$\begin{pmatrix} X'_1 \\ Y'_2 \\ Y_1 \end{pmatrix} = \begin{pmatrix} k_1 & 1 - k_2 & -k_1 + k_2 \\ 1 + k_1 & -k_2 & -k_1 + k_2 \\ 1 + k_1 & 1 - k_2 & -1 - k_1 + k_2 \end{pmatrix} \begin{pmatrix} X_1 \\ U_2 \\ U_1 \end{pmatrix}, \quad (27)$$

which meets the condition for losslessness with

$$D = \begin{pmatrix} 1 + k_1 & 0 & 0 \\ 0 & 1 - k_2 & 0 \\ 0 & 0 & -k_1 + k_2 \end{pmatrix}, \quad (28)$$

so for stability the coefficients k_1 and k_2 should satisfy

$$-1 < k_1 < k_2 < 1, \quad (29)$$

which is a nice compact condition to replace $-1 < k < 1$ for the original lattice two-pair. Note that the three-pair is also canonical since (27) can be implemented with only two multipliers via

$$\begin{aligned} X'_1 &= k_1(X_1 - U_1) + k_2(U_1 - U_2) + U_2 \\ Y'_2 &= k_1(X_1 - U_1) + k_2(U_1 - U_2) + X_1, \\ Y_1 &= k_1(X_1 - U_1) + k_2(U_1 - U_2) + X_1 + U_2 - U_1 \end{aligned} \quad (30)$$

where the first two lines tell us how $x_1[k+1]$ and $x_2[k+1]$ of the new state are computed. From this we see that the state matrix of a ladder built from second-order sections has a diagonal $A_{11} = k_1$, $A_{22} = k_2(1 + k_3 - k_4)$, $A_{33} = k_3$, $A_{44} = k_4(1 + k_5 - k_6)$, etcetera. In Fig. 4 the implementation of the n -th order ladder is drawn for even n , where the last section is closed off with an inverter, so $A_{nn} = k_n$.

In order to do calculations on the ladder, we need to describe the building block of Fig. 3 as a two-pair with a

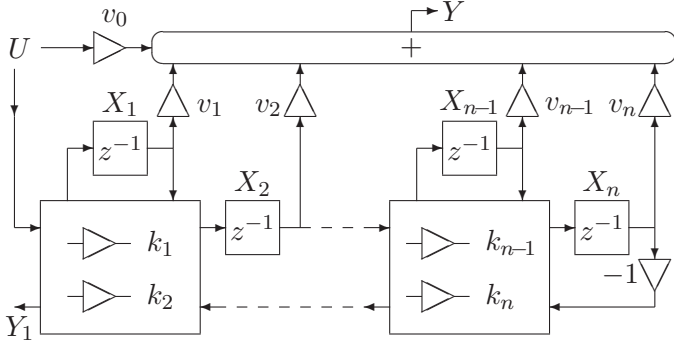


Fig. 4. Implementation of an even-order wave-adapter-ladder

2×2 scattering matrix \mathbf{S} or chain matrix \mathbf{R} . Using $X'_1 = zX_1$ and $Y'_2 = zY_2$, and eliminating X_1 from (30) yields

$$\mathbf{S} = \frac{\begin{pmatrix} -(1+k_1-k_2)z^2+k_2z & (1-k_2)(z^2+z) \\ (k_2-k_1)(z+1) & -k_2z+(1+k_1-k_2) \end{pmatrix}}{z(z-k_1)} \quad (31)$$

$$\mathbf{R} = \frac{\begin{pmatrix} z(z-k_1) & k_2z-(1+k_1-k_2) \\ -(1+k_1-k_2)z^2+k_2z & -k_1z+1 \end{pmatrix}}{(k_2-k_1)(z+1)}, \quad (32)$$

so for a ladder of a single section (with $Y_2 = -U_2 = X_2$)

$$\begin{aligned} \frac{Y_1}{U} &= -\frac{(1+k_1-k_2)z^2 - (k_1+k_2)z + 1}{z^2 - (k_1+k_2)z + (1+k_1-k_2)} \\ \frac{X_2}{U} &= \frac{(k_2-k_1)(z+1)}{z^2 - (k_1+k_2)z + (1+k_1-k_2)}. \end{aligned} \quad (33)$$

As expected, Y_1/U is all-pass though X_2/U is no longer all-pole. For any even n , X_n/U is found to have a numerator $(k_2-k_1) \cdots (k_n-k_{n-1})(z+1)^{n/2}$.

The denominator coefficient vector for the second-order case can be written in matrix form as (cf. Section II)

$$\mathbf{I}_{13} - k_1 \mathbf{I}'_{13} - k_2 \mathbf{J}'_{13} + (1+k_1-k_2) \mathbf{J}_{13}, \quad (34)$$

where \mathbf{I} and \mathbf{J} are defined as before and \mathbf{I}' has ones on a diagonal set off one from (and with the same length as) the principal diagonal and zeros elsewhere, and likewise for \mathbf{J}' with respect to the cross diagonal. This expression cannot be factored as in (10), and unfortunately it also has no easy pseudo-inverse as in (14). We can however write the denominator vector $\mathbf{a}^{(n)}$ for the general even-order case as a matrix product $\mathbf{M}_n \mathbf{M}_{n-2} \cdots \mathbf{M}_2$ as in (12), where

$$\mathbf{M}_i = \mathbf{I} - k_{i-1} \mathbf{I}' - k_i \mathbf{J}' + (1+k_{i-1}-k_i) \mathbf{J} \quad (35)$$

with dimension $(n-i+1) \times (n-i+3)$ for $i=2, 4, \dots, n$.

Now if we know the initial values k_1 and k_2 from $\mathbf{a}^{(n)}$, we can start the recursion to calculate the coefficient vectors $\mathbf{a}^{(n-i)}$ of the denominators that remain after taking away successive sections of the ladder (from left to right):

$$\mathbf{a}^{(n-i)} = \mathbf{a}^{(n-i+2)} \text{pinv}(\mathbf{M}_i), \quad (36)$$

holding for $i=2, 4, \dots, n$, where $\mathbf{a}^{(0)} = 1$ and the pseudo-inverse of \mathbf{M}_i is such that $\mathbf{M}_i \cdot \text{pinv}(\mathbf{M}_i)$ is equal to the identity matrix of dimension $(n-i+1) \times (n-i+1)$. Luckily, the built-in Matlab function `pinv` provides a fast and accurate routine for the calculation, so we do not really need a closed-form inverse of (35). For the recursion, however, we do need to express k_{i+1} and k_{i+2} in the coefficients of $\mathbf{a}^{(n-i)}$ in closed form, starting with k_1 and k_2 . The result, which is not trivial, is found to be

$$\begin{aligned} k_1 &= \frac{(a_0^{(n)} - a_n^{(n)}) \sum_{j=0}^{n/2} (-1)^j a_j^{(n)} / a_0^{(n)}}{\sum_{j=0}^{n/2} (-1)^j (n-2j) (a_j^{(n)} - a_{n-j}^{(n)})} - 1 \\ k_2 &= k_1 + 1 - a_n^{(n)} / a_0^{(n)}, \end{aligned} \quad (37)$$

with similar expressions for k_{i+1} and k_{i+2} if we replace all the n 's in (37) by $n-i$.

In order to find the state description of the wave-adapter-ladder, we again need to determine the numerator coefficient vectors of the transfer functions X_i/U and put them as rows in a matrix \mathbf{T} for a similarity transform of the ladder to Direct Form II, as before (cf. Section II). We find

$$\mathbf{t}^{(i)} = (k_2 - k_1) \cdots (k_i - k_{i-1}) \mathbf{a}^{(n-i)} * \mathbf{c}^{(i/2)} \quad (38)$$

for the numerators of the even states, where $*$ denotes convolution and the vector $\mathbf{c}^{(i/2)}$ contains the coefficients of the polynomial $(z+1)^{i/2}$, i.e. row $i/2$ of Pascal's triangle. Note that, appended with $i/2-1$ leading zeros, $\mathbf{t}^{(i)}$ are the even rows of \mathbf{T} . The odd rows are found with

$$\mathbf{t}^{(i-1)} = \sum_{j \geq i}^n (k_2 - k_1) \cdots (k_j - k_{j-1}) \mathbf{a}^{(n-j)} * \mathbf{c}^{(j/2-1)} * [1, -1] \quad (39)$$

where the summation is over all even rows j below the odd row we want to determine. We can also write equivalently:

$$\mathbf{t}^{(i-1)} = \sum_{j \geq i}^n \mathbf{t}^{(j)} *^{-1} [1, 1] * [1, -1], \quad (40)$$

where $*^{-1}$ denotes deconvolution which, like convolution, is a standard Matlab function (`conv` and `deconv`). Note that row $i-1$ has $i/2-1$ leading zeros, the same as the row below it. Having found \mathbf{T} , we can again extend it with a first row $\mathbf{t}^{(0)} = \mathbf{a}^{(n)}$, the numerator of the transfer function $X_0/U = U/U = 1$, to make an $(n+1) \times (n+1)$ matrix \mathbf{V}

and determine the vector \mathbf{v} of output multipliers with (23). Although \mathbf{T} and \mathbf{V} are no longer upper triangular, the state matrix \mathbf{A} will still contain only one full diagonal below the principal one, and on the diagonal below that only entries in the even columns. The main difference compared to the lattice-ladder lies in the fact that \mathbf{A} has entries that are at most a product of $n/2$ ladder coefficients k_j instead of n , and that the transfer functions to the states have $i/2$ extra zeros. All of these are at $z = -1$ for the even state of section $i/2$, whereas one zero is flipped to $z = +1$ for the odd state of that section.

Before we can write the Matlab function to calculate the ladder coefficients k_j from (37) and the recursion (36), we need to deal with the odd-order case. The simplest way to do this is by extending the ladder on the left with a first-order lattice section, since we can take its coefficient k_1 equal to $a_n^{(n)}$ and we know how to find the transfer function of the remaining even-order ladder if we remove it. We will deviate slightly from this solution by choosing $k_1 = -a_n^{(n)}$. As we will see in some examples, this has the advantage that all the ladder coefficients k_j are positive for low-pass filters (with cut-off frequency below $\pi/2$ such that all poles are in the right-half of the unit circle). The first step in the recursion then is as (13) for $i = 1$ and reversed sign for k_1 :

$$\mathbf{a}^{(n-1)} = \mathbf{a}^{(n)}(\mathbf{I}_{n+1,n} + k_1 \mathbf{J}_{n+1,n}) / (1 - k_1^2). \quad (41)$$

Next, we can put $\mathbf{a}^{(n-1)}$ into (36), since $n - 1$ is even. In the following Matlab program the wave-adapted-ladder is calculated from a given transfer function $\mathbf{b}^{(n)}$, $\mathbf{a}^{(n)}$.

```
function [A,B,C,D,k,v,p,T]=ladder1(b,a,nb,s)
n=length(a)-1; r=rem(n,2); H=freqz(b,a,4096);
k=zeros(1,n); K=zeros(n,n); V=zeros(n+1); V(1,:)=a;
b=[b,zeros(1,n+1-length(b))]; if nargin<4, s=''; end
if r==1
    I=eye(n+1,n); J=flipud(I);
    k(1)=-a(n+1); a=a*(I+k(1)*J)/(1-k(1)^2);
end
for i=2:r:2:n
    m=length(a)-1; num=(-1)^(0:m)*a*(a(1)-a(m+1));
    den=(-1)^(0:m/2)*(m:-2:0)*(a(1:m/2+1)-a(m+1:-1:m/2+1))';
    k(i-1)=num/den-1; k(i)=k(i-1)+a(1)-a(m+1); I=eye(m-1,m+1);
    J=flipplr(I); I1=I(:,[m+1,1:m]); J1=flipplr(I1);
    a=a*pinv(I+(1+k(i-1)-k(i))*J-k(i-1)*I1-k(i)*J1); a=a/a(1);
end
if nargin>2, k=2^-(nb-1)*round(2^(nb-1)*k); end
for i=n:-2:2+r
    c=round(2^((i-r)/2)*binopdf(0:(i-r)/2,(i-r)/2,1/2));
    V(i+1,(i+r)/2+1:n+1)=prod(k(2+r:2:i)-k(1+r:2:i-1))*conv(a,c);
    V(i+1,:)=V(i+1,:)*(1-r*k(1)); K(i,i)=(1-r*k(1))/(1+r*k(1));
    V(i,:)=conv(deconv(sum(V([i,i+1:2:n+1],:)),[1 1]),[1 -1]);
    K(i,i)=K(i,i)*prod(k(2+r:2:i)-k(1+r:2:i-1))/prod(1-k(2+r:2:i));
    K(i-1,i-1)=(1-k(i))/(1+k(i-1))*K(i,i); I=eye(m-1,m+1);
    J=flipplr(I); I1=I(:,[m+1,1:m]); J1=flipplr(I1);
    a=a*(I+(1+k(i-1)-k(i))*J-k(i-1)*I1-k(i)*J1); m=length(a)+1;
end
if r==1
    V(2,2:n+1)=(1-k(1))*a; K(1,1)=(1-k(1))/(1+k(1));
    I=eye(n,n+1); J=flipplr(I); a=a*(I-k(1)*J);
end
p=ceil(log2(diag(K)')/2);
if findstr(s,'s')
    V=diag(2.^[0,-p])*V; K=diag(diag(K)'.*2.^(-2*p));
end
V(1,:)=a; A=[-a(2:n+1);eye(n-1,n)]; v=b/V;
if nargin>2, v=2^-(nb-1)*round(2^(nb-1)*v); end
T=V(2:n+1,2:n+1); A=T*A/T; A(find(abs(A)<eps))=0; B=T(:,1);
C=v(2:n+1); D=v(1); W=dgram(A','C'); [Hq,w]=freqz(v*V,a,4096);
if findstr(s,'p'), plot(w/pi,20*log10(abs([H,Hq]))); grid; end
if findstr(s,'o'), Go=sum(sqrt(eig(K*W)))^2/n; display(Go); end
if findstr(s,'n'), Gn=diag(K)'+diag(W); display(Gn); end
if findstr(s,'s'), Gs=sum(diag(W)); display(Gs); end
```

IV. SCALING AND NOISE GAIN OF LADDER FILTERS

We already know that the ladders of Figs. 2 & 4 are low noise with signal quantization (and low sensitivity with coefficient quantization) due to their being built with lossless sections, but exactly how low noise are they? The easiest way to do quantization noise calculations is by introducing the gramians \mathbf{K} and \mathbf{W} for controllability and observability. These were already used in the two given Matlab programs `lattice1` and `ladder1`, and can be calculated with the built-in function `dgram`. They are defined as

$$\mathbf{K} = \sum_{j=0}^{\infty} (\mathbf{A}^j \mathbf{B})(\mathbf{A}^j \mathbf{B})^T = \mathbf{A} \mathbf{K} \mathbf{A}^T + \mathbf{B} \mathbf{B}^T$$

$$\mathbf{W} = \sum_{j=0}^{\infty} (\mathbf{C} \mathbf{A}^j)^T (\mathbf{C} \mathbf{A}^j) = \mathbf{A}^T \mathbf{W} \mathbf{A} + \mathbf{C}^T \mathbf{C}, \quad (42)$$

where the recursive definition is used for calculation. On the diagonals of these gramians are the power gains from input to state and from state to output respectively, so \mathbf{K} can be used for scaling and \mathbf{W} can be used to calculate the noise gain, i.e. the factor by which the quantization noise power $q^2/12$ is amplified at the output. After scaling, which can be viewed as a diagonal similarity transform $\mathbf{T}_s = \text{diag}(\sqrt{K_{jj}})$, the noise gain is given by $\sum_1^n K_{jj} W_{jj}$. Using `snr=6 (nb-1) dB` as a rule of thumb to determine the signal-to-noise ratio of a signal quantizer with `nb` bits and quantization step size $q = 2^{-(nb-1)}$, a filter with noise gain G_n simply has `snr=6 (nb-1) - 10 log10 G_n dB`.

A typical property of ladder filters is that the matrix \mathbf{K} is diagonal. Since we know the matrices \mathbf{A} and \mathbf{B} in terms of the ladder coefficients, we should be able to express \mathbf{K} in k_j as well. For the lattice-ladder we find

$$K_{jj} = \frac{(1 + k_1) \cdots (1 + k_j)}{(1 - k_1) \cdots (1 - k_j)} \quad (43)$$

and $K_{ij} = 0$ for $i \neq j$. The wave-adapted-ladder yields

$$K_{ii} = \frac{(k_2 - k_1) \cdots (k_i - k_{i-1})}{(1 - k_2) \cdots (1 - k_i)}, \quad K_{i-1,i-1} = \frac{1 - k_i}{1 + k_{i-1}} K_{ii} \quad (44)$$

for even n and $i = 2, 4, \dots, n$. In case n is odd,

$$K_{11} = \frac{1 - k_1}{1 + k_1}, \quad K_{ii} = \frac{(k_3 - k_2) \cdots (k_i - k_{i-1})}{(1 - k_3) \cdots (1 - k_i)} K_{11} \quad (45)$$

for $i = 3, 5, \dots, n$, with $K_{i-1,i-1}$ as in (44). For both the even and odd-order case $K_{ij} = 0$ off the diagonal. That \mathbf{K} can be given in terms of such closed-form expressions is quite surprising, as is the fact that it is diagonal. As a result, we can perform the scaling operation without having to determine \mathbf{K} numerically with (42).

Scaling can be done with the diagonal similarity transform $\mathbf{T}_s = \text{diag}(\sqrt{K_{jj}})$ if we allow that the operation takes n extra multipliers to perform: every state variable is scaled down with $1/\sqrt{K_{jj}}$ and all output multipliers v_j are scaled up with the inverse (the latter taking no extra hardware). In practice, and certainly in our case where we want the ladder to remain canonical, scaling is often done with shift operations, i.e. power-of-two multipliers. So instead of $\sqrt{K_{jj}}$ we take the nearest larger integer power of two, such that after scaling the entries on the diagonal of \mathbf{K}_s will be less than unity instead of equal to one (which in our case would mean $\mathbf{K}_s = \mathbf{I}$). As a result, the noise gain G_s with power-of-two scaling will be larger than G_n with exact scalars:

$$G_s = \text{trace}(\mathbf{W}_s) = \text{trace}(\mathbf{T}'_s \mathbf{W} \mathbf{T}'_s), \quad (46)$$

where $\mathbf{T}'_s = \text{diag}(2^{p_j})$ with $p_j = \text{ceil}(\log_2 \sqrt{K_{jj}})$.

Returning to the question posed at the beginning of this section: from the fact that the gramian \mathbf{K} of a ladder filter is diagonal we can show quantitatively how close the implementation is to the optimal low noise solution that uses $(n+1)^2$ multipliers (all the degrees of freedom in an n -th order state space description). To see why this is true, we first note that the sum $\sum K_{jj} W_{jj}$ is equal to the trace of the product matrix $\mathbf{K} \mathbf{W}$ if one (or both) of the matrices is diagonal. So the noise gain (with exact scalars) is

$$G_n = \sum_{j=1}^n K_{jj} W_{jj} = \text{trace}(\mathbf{K} \mathbf{W}) = \sum_{j=1}^n \mu_j^2, \quad (47)$$

where μ_j^2 are the (positive) eigenvalues of the product matrix. From system theory, on the other hand, we know that the optimal solution has a noise gain $G_o = (\sum_{j=1}^n \mu_j)^2 / n$. So we can conclude

$$G_o \leq G_n < nG_o, \quad (48)$$

where the lower bound (the ladder is optimal) holds if all the eigenvalues are equal, whereas the upper bound corresponds to the case where the eigenvalues are very dissimilar, i.e. one pronounced μ_p is much larger than all others. In practice (for standard filter types) that case never occurs. In our examples (up to 12th order) G_n is never over $2G_o$, so we seldom loose more than 3 dB of SNR with the use of n^2 fewer multipliers. Of course the noise gain G_s with power-of-two scaling is larger (in theory up to 4 times) but nominally only about 3 dB, so in total we could loose 1 bit (= 6 dB) of accuracy in practice. In the next section we will also look at coefficient quantization (with the same number of bits) and find that the best feasible stop-band of our filters is round about the SNR in dB down as a rule, which means that the ladder is almost perfect in that respect too.

V. DESIGN EXAMPLES

As a first example we design a 6th order elliptic filter with passband ripple 0.1 dB, stopband -100 dB and cut-off frequency $\pi/5$, and compare it to the same filter with a cut-off frequency of $\pi/50$. We note the following:

```
>> [b,a]=ellip(6,.1,100,.2); [A,B,C,D,k,v,p]=ladder1(b,a,18,'ons');
Go = 1.1700    Gn = 2.0002    Gs = 4.4597
>> k
k = 0.1841    0.8410    0.8384    0.9295    0.8442    0.9049
>> p
p = 0     2     -1     2     -1     1
>> [b,a]=ellip(6,.1,100,.02); [A,B,C,D,k,v,p]=ladder1(b,a,18,'ons');
Go = 1.1725    Gn = 2.0020    Gs = 4.4288
>> k
k = 0.8958    0.9975    0.9983    0.9993    0.9984    0.9991
>> p
p = -2     3     -2     3     -2     3
```

1. the noise gain is independent of the cut-off frequency;
2. with exact scaling we loose less than 3 dB of SNR compared to the optimum and close to 6 dB with power-of-two;
3. all ladder coefficients are positive (LPF with $\theta_c < \pi/2$);
4. the powers p (the numbers of bits across which to shift) increase the more narrow-band the filter becomes (positive p_j means shift to the right and negative is shift to the left). Due to the choice $k_1 = -a_n^{(n)}$ for odd-order ladders, all k_j remain positive if we change the filter to 7th order:

```
>> [b,a]=ellip(7,.1,100,.02); [A,B,C,D,k,v,p]=ladder1(b,a,18,'k');
k = 0.8994    0.9965    0.9991    0.9987    0.9994    0.9985    0.9990
```

Note that now the coefficients satisfy $-1 < k_{i-1} < k_i < 1$ for odd i , whereas they do for even i in the even-order case.

The effect that the noise seems to depend slightly on θ_c is only due to the finite precision of 18 bits we chose because that should be about enough for a stopband of -100 dB. To prove the point, we look at a second example of a 12th order elliptic filter with a cut-off frequency of $\pi/10$.

```
>> [b,a]=ellip(12,.1,100,.1); [A,B,C,D,k,v,p]=ladder1(b,a,18,'pons');
Go = 2.4147    Gn = 4.4178    Gs = 8.4708
```

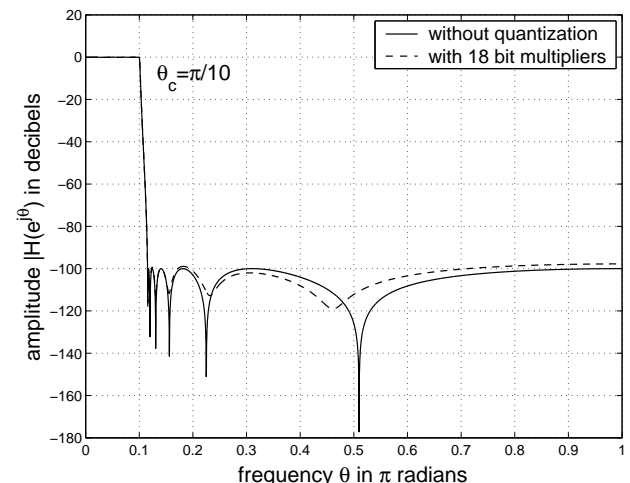


Fig. 5. Amplitude response of twelfth-order elliptic filter

Indeed SNR and feasible stopband with coefficient quantization are related as expected from our rule of thumb.

As a third and final example we compare the recursive parts (given by \mathbf{A} , \mathbf{B}) of two general second-order filters, the one implemented with two sections of a lattice-ladder, and the other with one section of the wave-adapter-ladder. Given the denominator vector $\mathbf{a}^{(n)} = [1, a_1, a_2]$ of the transfer function, the lattice filter is defined by

$$\mathbf{A} = \begin{pmatrix} -k_1 k_2 & -k_1(1-k_2) \\ 1+k_2 & -k_2 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 1+k_1 \\ 0 \end{pmatrix} \quad (49)$$

and $k_1 = a_2, \quad k_2 = a_1/(1+a_2),$

whereas the wave-adapter filter is given by

$$\mathbf{A} = \begin{pmatrix} k_1 & -(1-k_2) \\ 1+k_1 & k_2 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} k_2-k_1 \\ k_2-k_1 \end{pmatrix} \quad (50)$$

and $k_1 = -\frac{1}{2}(1+a_1-a_2), \quad k_2 = \frac{1}{2}(1-a_1-a_2).$

While both create the denominator with $\text{trace}(\mathbf{A}) = -a_1$ and $\det(\mathbf{A}) = a_2$, the filters are virtually equal in terms of noise gain and coefficient sensitivity with any numerator. They differ only in the relation between $\mathbf{a}^{(n)}$ and \mathbf{k} , which is still linear for the wave-adapter filter in this second-order case. The fact can also be recognized in the transformation matrix \mathbf{T} given by (18) for the lattice filter and by

$$\mathbf{T} = (k_2 - k_1) \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} \quad (51)$$

for the wave-adapter filter, the latter similarity transform simply being a (scaled) rotation. Indeed, from [6] we know that the best canonical second-order filter implementation (less than 3 dB from optimal) can be found with a 45 degree rotation of Direct Form II as in (51) without the factor $k_2 - k_1$. The resulting structure is called a WDF in [6] since its recursive part is a wave digital adapter. Of course dropping the factor $k_2 - k_1$ is not fundamental since we have to still scale the filter anyway. It just changes \mathbf{B} into $[1, 1]^T$ and the entries of the (diagonal) scaling gramian \mathbf{K} into $K_{11} = 1/(k_2 - k_1)(1 + k_1)$ and $K_{22} = 1/(k_2 - k_1)(1 - k_2)$, i.e. (44) for $i = 2$, divided by $(k_2 - k_1)^2$. As a matter of interest we also look at the similarity transform for $n = 4$,

$$\mathbf{T} = (k_2 - k_1) \begin{bmatrix} 1 & -1 - 2k_3 & 1 + 2k_3 & -1 \\ 1 & 1 - k_3 - k_4 & 1 - 2k_4 & 1 + k_3 - k_4 \\ 0 & k_4 - k_3 & 0 & k_3 - k_4 \\ 0 & k_4 - k_3 & 2(k_4 - k_3) & k_4 - k_3 \end{bmatrix}, \quad (52)$$

and see that it is still rather simple in view of the fact that the transform for the lattice-ladder by now would contain products of four k_j instead of two (or three instead of one if we drop the common factors in both cases).

Redoing the first two design examples for the lattice-ladder yields virtually the same results in terms of sensitivity and noise, as expected. We mention one exception: the non-canonical two-multiplier lattice-ladder is less intrinsically scaled than its canonical counterpart due to the fact that it lacks the multiplications $1 + k_i$ per section, i.e. the factors per row in (18) and (19). When we redo the 12th order elliptic filter with `lattice1(b, a, 18, 'p2')` for a plot of the amplitude response, we note that 18 bits do not suffice. After scaling (type `'ps2'`) there is no problem, however. Looking at `p` we see that the state variables are shifted up to 29 bits versus only 6 for the canonical ladder.

VI. CONCLUSION

We have presented an easy way to design IIR digital filters that are close to optimal (in terms of finite wordlength effects) and canonical at the same time by combining classical ladder design with wave digital design. One could argue (as indeed Fettweis does) that all lossless ladders are in fact WDF's and could have been designed as such. It is not directly clear however, apart from the second-order case, which analog reference filters would lead to the ladder filters treated here. More importantly, it was just our goal to design canonical, near-optimal filters (whether they are WDF's or not) without having to resort to their analog counterparts and do the tedious translation (for a designer not familiar with analog techniques) to digital filters with all the advantages that WDF's undeniably have.

As a closing remark we mention that the pseudo-inverse of (35) can be given in closed form [7], but using it instead of `pinv` does not improve the accuracy of `ladder1`.

REFERENCES

- [1] J.H.F. Ritzerfeld, "Design of canonical higher order digital filters," *Proc. ProRISC 2006, 17th Annual Workshop on Circuits, Systems and Signal Processing*, Veldhoven, The Netherlands, 23-24 November 2006, ISBN 90-73461-44-8; STW, Technology Foundation, Utrecht, The Netherlands, pp. 184-191, 2006.
- [2] A.H. Gray Jr. and J.D. Markel, "Digital lattice and ladder filters synthesis," *IEEE Trans. Audio & Electroacoust.*, vol. AU-21, pp. 491-500, 1973.
- [3] A. Fettweis, "Wave digital filters: Theory and practice," *Proc. IEEE*, vol. 74, pp. 270-327, 1986.
- [4] A. Fettweis, "Digital filters related to classical filter networks," *Arch. Elektron. & Übertragungstechn.*, vol. AEÜ-25, pp. 79-89, 1971.
- [5] J.H.F. Ritzerfeld, "On stability tests for continuous and discrete-time linear systems," *Proc. ProRISC 2005, 16th Annual Workshop on Circuits, Systems and Signal Processing*, Veldhoven, Netherlands, 17-18 November 2005, ISBN 90-73461-50-2; STW, Technology Foundation, Utrecht, The Netherlands, pp. 668-673, 2005.
- [6] J.H.F. Ritzerfeld, "Noise gain expressions for low noise second-order digital filter structures," *IEEE Trans. Circuits Syst. II - Analog Digit. Signal Process.*, vol. 52, no. 4, pp. 223-227, 2005.
- [7] J.H.F. Ritzerfeld, private communication, e-mail: @tue.nl, 2008.