On the design of Schur lattice digital filters

Robert G. Jenssen

September 6, 2025

Abstract

This article describes a computer-aided procedure for approximating an arbitrary transfer function by a tapped one-multiplier all pass lattice filter with integer coefficients. The all pass lattice has a simple stability criterion, that the lattice coefficients have magnitude less than unity, and, in addition, has good coefficient sensitivity and round-off noise properties. The procedure consists of finding an initial filter transfer function, performing constrained minimum-mean-squared-error optimisation of the filter response and searching for the fixed point filter coefficients.

1 Introduction

The Infinite Impulse Response (IIR) digital filter transfer function is a rational polynomial:

$$F(z) = \frac{B_N(z)}{A_N(z)} = \frac{b_0 + b_1 z^{-1} + \cdots + b_N z^{-N}}{1 + a_1 z^{-1} + \cdots + a_N z^{-N}} \tag{1}$$

Here z is a complex number and the response of the filter with respect to angular frequency, ω, is $F(e^{\imath\omega})$, where $\omega = 2\pi$ corresponds to the digital sampling frequency. This article describes a method for computer-aided design of the frequency response of $F(e^{\imath\omega})$. If the filter has $A_N(z) = 1$ then it is a Finite Impulse Response (FIR) filter and the optimisation problem is convex, has no constraints on the coefficients and is solved by well-known algorithms that minimise the peak error or the least-squared-error [7, 9, 10, 11, 12, 28, 30, 31]. Otherwise, the filter is an Infinite Impulse (IIR) filter. An IIR filter is stable if the roots, P_k, of the denominator polynomial, $A_N(z)$, lie within the unit circle, $|P_k| < 1$. *Lang* [18] applies Rouché's theorem to constrain the zeros of $A_N(z)$ during optimisation. *Lu* and *Hinamoto* [32] decompose $F(z)$ into a series connection of second-order sections and apply the "stabiliy triangle" to each section to ensure filter stability after optimisation. If the transfer function is represented in gain-pole-zero form:

$$F(z) = G\frac{\prod_{k=1}^{N}(z - Z_k)}{\prod_{k=1}^{N}(z - P_k)}$$

where G is the gain factor, the Z_k are the roots of $B_N(z)$ and the P_k are the roots of $A_N(z)$, then the constraints on the pole locations of $F(z)$ are $|P_k| < 1$. *Deczky* [1] and *Richards* [17] demonstrate the optimisation of stable IIR filter transfer functions expressed in gain-pole-zero form. Lattice filters are an alternative implementation of the IIR filter transfer function [22]. *Vaidyanathan et al.* [23, 21] show that a subset of the $F(z)$ transfer functions can be factored into the parallel sum of two all pass filters and that the resulting implementation has good coefficient sensitivity properties. They show lattice filter implementations of these all pass sections that are "structurally bounded" meaning that the section is all pass regardless of the values of the coefficients. *Gray* and *Markel* [2, 13] describe the implementation of $F(z)$ as a tapped all pass lattice filter based on the Schur polynomial decomposition algorithm [29, 8]. The Schur algorithm enables the decomposition of the denominator polynomial of the transfer function, $F(z)$, into a set of orthogonal polynomials. A result of this decompostion is a set of coefficients, often called "reflection coefficients", that have magnitude less than unity if-and-only-if the poles of the transfer function lie within the unit circle in the complex plane. Further, the numerator polynomial of $F(z)$ can be expressed as a weighted sum of the orthogonal polynomials. These coefficients and orthogonal polynomials correspond to a tapped all pass lattice filter implementation of the transfer function that has good round-off noise and coefficient sensitivity when the coefficients are truncated to integer values. This article reviews computer-aided techniques for designing tapped Schur lattice filters with integer coefficients : finding an initial IIR filter transfer function, constrained minimum-mean-squared-error optimisation of the filter frequency response in terms of the lattice coefficients and searching for integer filter coefficients. As an example, I demonstrate the design of a tapped Schur lattice implementation of a low pass differentiator filter. The coefficients and frequency response plots for this example are created with Octave [14, 19, 20]. The source code for this article is available at https://github.com/robertgj/DesignOfIIRFilters.

2 Schur lattice filters

2.1 The Schur polynomial decomposition

In 1918 *Issai Schur* [8] published a paper entitled "*On power series which are bounded in the interior of the unit circle*". From the abstract:

> The continued fraction algorithm introduced here very easily supplies an intrinsically important parametric representation for the coefficients of the power series to be considered.

Kailath [29] reviews the impact of this algorithm on modern signal processing. *Gray* and *Markel* describe the application of a similar algorithm to the autocorrelation analysis of speech [13] and to the synthesis of lattice digital filters [2]. In the following I use the inner product method of *Markel* and *Gray* [13] and *Parhi* [15, Chapter 12 and Appendix D].

Initialise an N'th order polynomial as:

$$\Phi_N(z) = A_N(z) = \sum_{n=0}^{N} \phi_n z^n$$

and define the reverse polynomial:

$$\hat{\Phi}_N(z) = z^N \Phi_N\left(z^{-1}\right)$$

The transfer function $\hat{\Phi}_N(z)/\Phi_N(z)$ represents an all pass filter. The Schur decomposition forms the polynomial $\Phi_{N-1}(z)$ as:

$$\Phi_{N-1}(z) = \frac{z^{-1}\left[\Phi_N(z) - k_N \hat{\Phi}_N(z)\right]}{s_N}$$

where s_N is a scaling constant and $k_N = \phi_0/\phi_N = \Phi_N(0)/\hat{\Phi}_N(0)$. The degree of $\Phi_{N-1}(z)$ is 1 less than that of $\Phi_N(z)$ since, by a change of variables, the numerator is:

$$\begin{aligned} z^{-1}\left\{\phi_N \Phi_N(z) - \phi_0 \hat{\Phi}_N(z)\right\} &= z^{-1} \sum_{n=0}^{N} \left\{\phi_N \phi_n z^n - \phi_0 \phi_{N-i} z^n\right\} \\ &= \sum_{n=1}^{N} \left\{\phi_N \phi_n - \phi_0 \phi_{N-i}\right\} z^{n-1} \end{aligned}$$

This degree reduction procedure is continued, resulting in the set of polynomials $\{\Phi_N(z), \Phi_{N-1}(z), \ldots, \Phi_0(z)\}$:

$$\Phi_{n-1}(z) = \frac{z^{-1}\left\{\Phi_n(z) - k_n \hat{\Phi}_n(z)\right\}}{s_n} \tag{2}$$

and

$$\hat{\Phi}_{n-1}(z) = \frac{\left\{\hat{\Phi}_n(z) - k_n \Phi_n(z)\right\}}{s_n}$$

Markel and *Gray* [13, Equations 13a and 13b] define an inner product of two polynomials, $P(z)$ and $Q(z)$:

$$\langle P(z), Q(z)\rangle = \frac{1}{2\pi\imath} \oint_C \frac{P(z)\, Q\left(z^{-1}\right)}{A_N(z)\, A_N\left(z^{-1}\right)} \frac{dz}{z}$$

where all the zeros of $A_N(z)$ lie within the contour C. Two properties of $\Phi_n(z)$ under this inner product are:

1. The $\Phi_n(z)$ polynomials are orthogonal [15, Appendix D]:
$$\langle \Phi_m(z), \Phi_n(z)\rangle = 0 \text{ if } m \neq n$$
2. For $1 \leq n \leq N$, $|k_n| < 1$ if-and-only-if the zeros of $A_N(z)$ lie within the unit circle [13, pages 72 and 74]

In addition, if $s_n = \sqrt{1 - k_n^2}$, then the Φ_n polynomials are orthonormal [15, Appendix D]:

$$\langle \Phi_n(z), \Phi_n(z)\rangle = 1$$

2.2 The Schur one-multiplier lattice filter

In Equation 2, choose $s_n = 1 - \epsilon_n k_n$, where $\epsilon_n = \pm 1$, and initialise $\Lambda_N(z) = A_N(z)$. Then:

$$\Lambda_{n-1}(z) = \frac{z^{-1}\left\{\Lambda_n(z) - k_n \hat{\Lambda}_n(z)\right\}}{1 - \epsilon_n k_n} \tag{3a}$$

$$\hat{\Lambda}_{n-1}(z) = \frac{\left\{\hat{\Lambda}_n(z) - k_n \Lambda_n(z)\right\}}{1 - \epsilon_n k_n} \tag{3b}$$

where:

$$k_n = \frac{\Lambda_n(0)}{\hat{\Lambda}_n(0)} = \frac{\Phi_n(0)}{\hat{\Phi}_n(0)}$$

Rearranging Equation 3b and substituting it into Equation 3a gives:

$$z\Lambda_{n-1}(z) = (1 + \epsilon_n k_n)\Lambda_n(z) - k_n \hat{\Lambda}_{n-1}(z)$$
$$\hat{\Lambda}_n(z) = k_n \Lambda_n(z) + (1 - \epsilon_n k_n)\hat{\Lambda}_{n-1}(z)$$

Figure 1 shows the corresponding *all pass Schur one multiplier lattice* filter section.

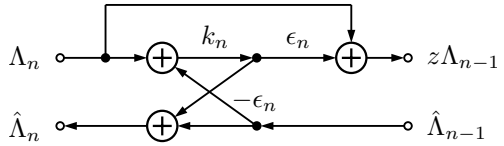


Figure 1: All pass Schur one multiplier lattice filter section

The expansion of the numerator polynomial of the transfer function, $F(z)$, in the orthogonal basis, $\Lambda_n(z)$, is:

$$B_N(z) = \sum_{n=0}^{N} c_n \Lambda_n(z)$$

Algorithm 2.1 shows the expansion of an arbitrary polynomial in a set of Schur orthogonal basis polynomials.

Algorithm 2.1 Schur polynomial expansion (see *Parhi* [15, Section 12.2.3]).

For any polynomial $N_m(z)$ of degree m, $(0 < m \leq N)$:

$Q(z) = N_m(z)$
$c_n = 0$, for $m < n \leq N$
for $n = m, m-1, \ldots, 0$ **do**
  $c_n = \frac{\hat{Q}(0)}{\hat{\Lambda}_n(0)}$
  $Q(z) = Q(z) - c_n \Lambda_n(z)$
end for

$\hat{Q}(z)$ and $\hat{\Lambda}_n(z)$ are the reverse polynomials of $Q(z)$ and $\Lambda_n(z)$.

Figure 2 shows the tapped Schur one multiplier lattice filter implementation of a second order transfer function.

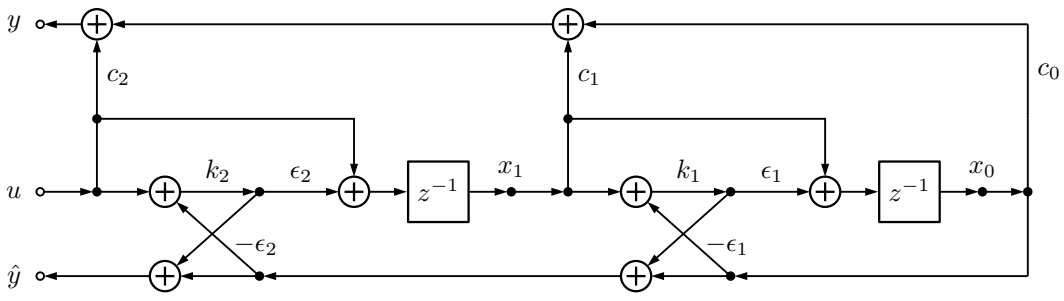


Figure 2: Second-order tapped Schur one-multiplier lattice filter

The relation between $\Lambda_n(z)$ and $\Phi_n(z)$ is:

$$\Lambda_N(z) = \Phi_N(z)$$
$$\Lambda_n(z) = \Phi_n(z)\sqrt{\frac{(1+\epsilon_N k_N)(1+\epsilon_{N-1}k_{N-1})\cdots(1+\epsilon_{n+1}k_{n+1})}{(1-\epsilon_N k_N)(1-\epsilon_{N-1}k_{N-1})\cdots(1-\epsilon_{n+1}k_{n+1})}}$$

The $\Lambda_n(z)$ polynomials are orthogonal but not orthonormal since:

$$\begin{aligned}\langle \Lambda_n(z), \Lambda_n(z)\rangle &= \langle \hat{\Lambda}_n(z), \hat{\Lambda}_n(z)\rangle \\ &= \frac{(1+\epsilon_N k_N)(1+\epsilon_{N-1}k_{N-1})\cdots(1+\epsilon_{n+1}k_{n+1})}{(1-\epsilon_N k_N)(1-\epsilon_{N-1}k_{N-1})\cdots(1-\epsilon_{n+1}k_{n+1})}\end{aligned}$$

If the input signal is random and white with unit power then the average power at an internal node, x_n, is $\langle \Lambda_n(z), \Lambda_n(z)\rangle$. The magnitude of $\langle \Lambda_n(z), \Lambda_n(z)\rangle$ can be adjusted by choosing the sign parameters, $\epsilon_N, \epsilon_{N-1}\ldots, \epsilon_{n+1}$. *Gray* and *Markel* [2, p. 496] suggest that one criterion for choosing the sign parameters is to require that the node associated with the largest k_n parameter in magnitude have the largest amplitude. The sign parameters are found recursively by requiring that the amplitudes at other nodes be as large as possible without exceeding the maximum value. If the maximum occurs for k_l then the recursion proceeds for $n = l-1, l-2, \ldots, 0$ and again for $n = l+1, l+2, \ldots, N$. The recursion is simple because:

$$\frac{\langle \Lambda_n(z), \Lambda_n(z)\rangle}{\langle \Lambda_{n+1}(z), \Lambda_{n+1}(z)\rangle} = \frac{1+\epsilon_{n+1}k_{n+1}}{1-\epsilon_{n+1}k_{n+1}}$$

By changing the sign parameter, this ratio can always be made smaller or larger than one. Algorithm 2.2 shows the method used to assign the sign parameters described by *Gray* and *Markel* [2, p. 496].

Algorithm 2.2 One multiplier lattice sign assignment [2, p. 496].

Assume that k_l has the largest magnitude of the k_n for $n = 1, 2, \ldots, N$. Define the quantities

$$Q_n = \frac{\langle \Lambda_n(z), \Lambda_n(z)\rangle}{\langle \Lambda_l(z), \Lambda_l(z)\rangle}$$
$$q_n = \frac{1+|k_n|}{1-|k_n|}$$

so that $Q_l = 1$. Each Q_n should be as large as possible without exceeding Q_l. Successive ratios are:

$$\frac{Q_n}{Q_{n+1}} = \begin{cases} q_n & if\ \epsilon_n = \ \ \operatorname{sign}(k_n) \\ 1/q_n & if\ \epsilon_n = -\operatorname{sign}(k_n) \end{cases} \tag{4}$$

Now assign the ϵ_n:

for $n = l-1, l-2, \ldots, 1$ **do**
  if $Q_{n+1} < 1/q_n$ **then**
    $\epsilon_n = -\operatorname{sign}(k_n)$
  else
    $\epsilon_n = \operatorname{sign}(k_n)$
  end if
end for
for $n = l+1, l+2, \ldots, N$ **do**
  if $Q_n < 1/q_n$ **then**
    $\epsilon_n = \operatorname{sign}(k_n)$
  else
    $\epsilon_n = -\operatorname{sign}(k_n)$
  end if
end for

2.3 State variable descriptions of the Schur one multiplier lattice filter

The state variable representation of a filter is:

$$\begin{bmatrix} \mathbf{x}' \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{u} \end{bmatrix} \tag{5}$$

where $\mathbf{u}$ is the input, $\mathbf{x}$ is the filter state, for convenience $\mathbf{x}' = z\,\mathbf{x}$, and $\mathbf{y}$ is the filter output. $\mathbf{u}$ and $\mathbf{y}$ may be vectors. $\mathbf{A}$ is called the state transition matrix. Examination of Figure 1 and Figure 2 suggests that the calculation of the all pass output, $\hat{y}$, of the tapped Schur lattice filter can be represented as a matrix product:

$$\begin{bmatrix} \hat{y}_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ u \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & & \cdots & 0 \\ 0 & 1 & & & & \vdots \\ \vdots & & \ddots & & & \\ & & & \ddots & & \vdots \\ 0 & & & & 1 & 0 \\ 0 & \cdots & & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ u \end{bmatrix}$$

$$\begin{bmatrix} x'_0 \\ \hat{y}_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ u \end{bmatrix} = \begin{bmatrix} -k_1 & (1+k_1\epsilon_1) & 0 & \cdots & \cdots & 0 \\ (1-k_1\epsilon_1) & k_1 & 0 & & & \vdots \\ 0 & 0 & 1 & & & \\ \vdots & & & \ddots & & \vdots \\ 0 & & & & 1 & 0 \\ 0 & & \cdots & & 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{y}_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ u \end{bmatrix}$$

$$\begin{bmatrix} x'_0 \\ x'_1 \\ \hat{y}_2 \\ \vdots \\ x_{N-1} \\ u \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & -k_2 & (1+k_2\epsilon_2) & 0 & & \vdots \\ 0 & (1-k_2\epsilon_2) & k_2 & 0 & & \\ \vdots & & & \ddots & & \vdots \\ 0 & & & & 1 & 0 \\ 0 & \cdots & & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} x'_0 \\ \hat{y}_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ u \end{bmatrix}$$

$$\vdots$$

$$\begin{bmatrix} x'_0 \\ x'_1 \\ \vdots \\ x'_{N-2} \\ \hat{y}_{N-1} \\ u \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & & & \cdots & 0 \\ 0 & 1 & & & & & \vdots \\ \vdots & & \ddots & & & & \vdots \\ 0 & & & -k_{N-1} & (1+k_{N-1}\epsilon_{N-1}) & & 0 \\ 0 & & & (1-k_{N-1}\epsilon_{N-1}) & k_{N-1} & & 0 \\ 0 & \cdots & \cdots & 0 & 0 & & 1 \end{bmatrix} \begin{bmatrix} x'_0 \\ \vdots \\ x'_{N-3} \\ \hat{y}_{N-2} \\ x_{N-1} \\ u \end{bmatrix}$$

$$\begin{bmatrix} x'_0 \\ x'_1 \\ x'_2 \\ \vdots \\ x'_{N-1} \\ \hat{y} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \cdots & & \cdots & 0 \\ 0 & 1 & & & & \vdots \\ \vdots & & \ddots & & & \vdots \\ 0 & & & 1 & 0 & 0 \\ 0 & & & 0 & -k_N & (1+k_N\epsilon_N) \\ 0 & \cdots & & 0 & (1-k_N\epsilon_N) & k_N \end{bmatrix} \begin{bmatrix} x'_0 \\ x'_1 \\ \vdots \\ x'_{N-2} \\ \hat{y}_{N-1} \\ u \end{bmatrix}$$

The construction of the state variable description of the Schur one multiplier lattice filter is summarised in Algorithm 2.3.

Algorithm 2.3 Construction of a state variable description of the Schur one multiplier lattice filter.

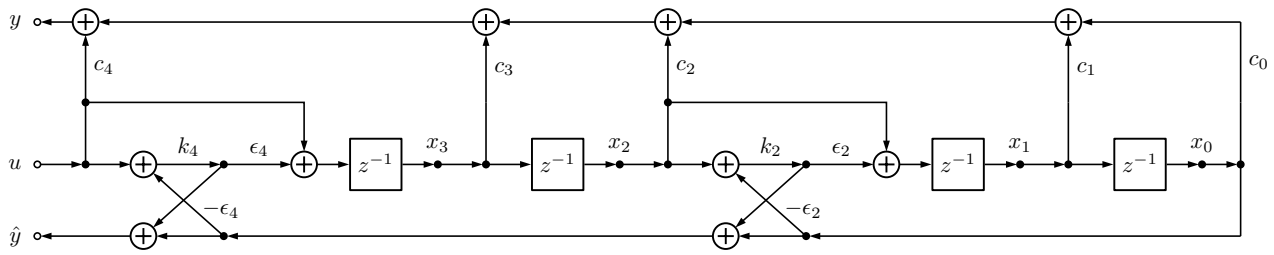
Given $\{k_1, k_2, \ldots, k_N\}$, $\{\epsilon_1, \epsilon_2, \ldots, \epsilon_N\}$ and $\{c_0, c_1, \ldots, c_N\}$:

$\hat{y}_0 = x_0$
for $n = 1, \ldots, N-1$ **do**
  $x'_{n-1} = -k_n \hat{y}_{n-1} + (1 + k_n \epsilon_n)\, x_n$
  $\hat{y}_n = (1 - k_n \epsilon_n)\, \hat{y}_{n-1} + k_n x_n$
end for
$x'_{N-1} = -k_N \hat{y}_{N-1} + (1 + k_N \epsilon_N)\, u$
$\hat{y} = (1 - k_N \epsilon_N)\, \hat{y}_{N-1} + k_N u$
$y = c_0 x_0 + c_1 x_1 + \cdots + c_{N-1} x_{N-1} + c_N u$

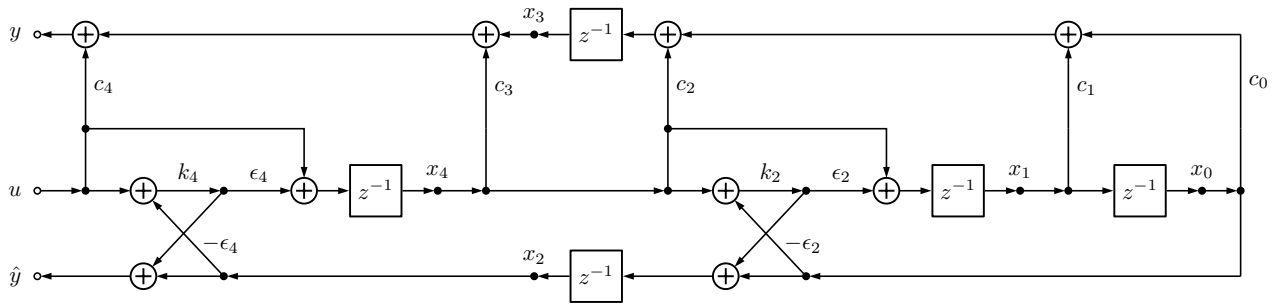
The state variable matrix calculations for a second order tapped Schur one muliplier lattice filter are:

$$\begin{bmatrix} x'_0 \\ x'_1 \\ \hat{y} \\ y \end{bmatrix} = \begin{bmatrix} -k_1 & 1 + \epsilon_1 k_1 & 0 \\ -k_2 (1 - \epsilon_1 k_1) & -k_1 k_2 & 1 + \epsilon_2 k_2 \\ (1 - \epsilon_1 k_1)(1 - \epsilon_2 k_2) & k_1 (1 - \epsilon_2 k_2) & k_2 \\ c_0 & c_1 & c_2 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ u \end{bmatrix}$$

The latency in the arithmetic calculation of y and $\hat{y}$ increases with the filter order. *Parhi* [15, Chapter 4 and Section 12.8] describes the retiming and pipelining of lattice filters for reduced latency. One method is to design the lattice filter with coefficients of the denominator polynomial, $A_N(z)$, only in z^{-2n}, where N is even and $n = 1, \ldots, \frac{N}{2}$. *Deczky* [1] and *Richards* [17] describe such filters as $R = 2$. Figure 3 shows such a fourth order tapped Schur one multiplier lattice filter with and without pipelining.



(a) Without pipelining



(b) With pipelining

Figure 3: Fourth-order tapped Schur one-multiplier lattice filter with denominator coefficients only for z^{-2n}

The state variable matrix calculations for a pipelined fourth order tapped Schur one-muliplier lattice filter with denominator polynomial coefficients only for z^{-2n} are:

$$\begin{bmatrix} x'_0 \\ x'_1 \\ x'_2 \\ x'_3 \\ x'_4 \\ \hat{y} \\ y \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ -k_2 & 0 & 0 & 0 & 1 + \epsilon_2 k_2 & 0 \\ 1 - \epsilon_2 k_2 & 0 & 0 & 0 & k_2 & 0 \\ c_0 & c_1 & 0 & 0 & c_2 & 0 \\ 0 & 0 & -k_4 & 0 & 0 & 1 + \epsilon_4 k_4 \\ 0 & 0 & 1 - \epsilon_4 k_4 & 0 & 0 & k_4 \\ 0 & 0 & 0 & 1 & c_3 & c_4 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ u \end{bmatrix}$$

The construction of the state variable description of the pipelined tapped Schur one multiplier lattice filter with denominator polynomial coefficients only for z^{-2n} is summarised in Algorithm 2.4.

Algorithm 2.4 Construction of a state variable description of the pipelined tapped Schur one multiplier lattice filter with denominator polynomial coefficients only for z^{-2n}.

Given N even, $\{k_2, k_4 \ldots, k_N\}$, $\{\epsilon_2, \epsilon_4, \ldots, \epsilon_N\}$ and $\{c_0, c_1, \ldots, c_N\}$:

$x'_0 = x_1$
$x'_1 = -k_2 x_0 + (1 + k_2\epsilon_2)\, x_4$
$x'_2 = (1 - k_2\epsilon_2)\, x_0 + k_2 x_4$
$x'_3 = c_0 x_0 + c_1 x_1 + c_2 x_4$
for $n = 2, \ldots, \frac{N}{2} - 1$ **do**
  $x'_{3n-2} = -k_{2n} x_{3n-4} + (1 + k_{2n}\epsilon_{2n})\, x_{3n+1}$
  $x'_{3n-1} = (1 - k_{2n}\epsilon_{2n})\, x_{3n-4} + k_{2n} x_{3n+1}$
  $x'_{3n} = x_{3n-3} + c_{2n-1} x_{3n-2} + c_{2n} x_{3n+1}$
end for
$x'_{3\frac{N}{2}-2} = -k_N x_{3\frac{N}{2}-4} + (1 + k_N\epsilon_N)\, u$
$\hat{y} = (1 - k_N\epsilon_N)\, x_{3\frac{N}{2}-4} + k_N u$
$y = x_{3\frac{N}{2}-3} + c_{N-1} x_{3\frac{N}{2}-2} + c_N u$

2.4 Frequency response of the tapped Schur one multiplier lattice filter

Eliminating the filter state, $\mathbf{x}$, from Equation 5 gives:

$$\mathbf{F}(z) = \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} \tag{6}$$

Thiele [16] uses the identity:

$$\begin{aligned} \mathbf{R}\mathbf{R}^{-1} &= \mathbf{I} \\ \frac{\partial \mathbf{R}}{\partial \chi}\mathbf{R}^{-1} + \mathbf{R}\frac{\partial \mathbf{R}^{-1}}{\partial \chi} &= 0 \\ \frac{\partial \mathbf{R}}{\partial \chi} &= -\mathbf{R}\frac{\partial \mathbf{R}^{-1}}{\partial \chi}\mathbf{R} \end{aligned}$$

Theile shows the sensitivity functions of $\mathbf{F}(z)$ with respect to the components α, β, γ and δ of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ respectively:

$$\begin{aligned} \frac{\partial \mathbf{F}}{\partial z} &= -\mathbf{C}(z\mathbf{I} - \mathbf{A})^{-2}\mathbf{B} \\ \frac{\partial \mathbf{F}}{\partial \alpha} &= \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\frac{\partial \mathbf{A}}{\partial \alpha}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} \\ \frac{\partial \mathbf{F}}{\partial \beta} &= \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1} \\ \frac{\partial \mathbf{F}}{\partial \gamma} &= (z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} \\ \frac{\partial \mathbf{F}}{\partial \delta} &= \mathbf{I} \end{aligned}$$

Substituting $z = e^{\imath\omega}$ into Equation 6 gives the complex frequency response of the state variable filter on the unit circle. In the following I will use $\mathbf{R} = (e^{\imath\omega}\mathbf{I} - \mathbf{A})^{-1}$. The components α of $\mathbf{A}$, etc. may themselves be functions of other variables (for example, the c and k coefficients of a tapped Schur one multiplier lattice filter) that I will represent by χ. The sensitivity functions of the complex frequency response $\mathbf{F}(e^{\imath\omega})$ with respect to ω and χ are:

$$\begin{aligned} \frac{\partial \mathbf{R}}{\partial \omega} &= -\imath e^{\imath\omega}\mathbf{R}\mathbf{R} \\ \frac{\partial \mathbf{F}}{\partial \omega} &= -\imath e^{\imath\omega}\mathbf{C}\mathbf{R}\mathbf{R}\mathbf{B} \\ \frac{\partial \mathbf{F}}{\partial \chi} &= \frac{\partial \mathbf{C}}{\partial \chi}\mathbf{R}\mathbf{B} + \mathbf{C}\mathbf{R}\frac{\partial \mathbf{A}}{\partial \chi}\mathbf{R}\mathbf{B} + \mathbf{C}\mathbf{R}\frac{\partial \mathbf{B}}{\partial \chi} + \frac{\partial \mathbf{D}}{\partial \chi} \\ \frac{\partial^2 \mathbf{F}}{\partial \chi \partial \omega} &= -\imath e^{\imath\omega}\left[\frac{\partial \mathbf{C}}{\partial \chi}\mathbf{R}\mathbf{R}\mathbf{B} + \mathbf{C}\mathbf{R}\mathbf{R}\frac{\partial \mathbf{A}}{\partial \chi}\mathbf{R}\mathbf{B} + \mathbf{C}\mathbf{R}\frac{\partial \mathbf{A}}{\partial \chi}\mathbf{R}\mathbf{R}\mathbf{B} + \mathbf{C}\mathbf{R}\mathbf{R}\frac{\partial \mathbf{B}}{\partial \chi}\right] \end{aligned}$$

The complex frequency response of the filter is:

$$\mathbf{F} = \Re\mathbf{F} + \imath\Im\mathbf{F}$$

The squared-magnitude response of the filter is:

$$|\mathbf{F}|^2 = \Im\mathbf{F}^2 + \Re\mathbf{F}^2$$

The gradients of the squared-magnitude response of the filter with respect to the Schur lattice filter coefficients are:

$$\frac{\partial |\mathbf{F}|^2}{\partial \chi} = 2\left[\Im\mathbf{F}\Im\frac{\partial \mathbf{F}}{\partial \chi} + \Re\mathbf{F}\Re\frac{\partial \mathbf{F}}{\partial \chi}\right]$$

The gradient of the squared-magnitude response of the filter with respect to the angular frequency is:

$$\frac{\partial |\mathbf{F}|^2}{\partial \omega} = 2\left[\Im\mathbf{F}\Im\frac{\partial \mathbf{F}}{\partial \omega} + \Re\mathbf{F}\Re\frac{\partial \mathbf{F}}{\partial \omega}\right]$$

so that:

$$\frac{\partial^2 |\mathbf{F}|^2}{\partial \omega \partial \chi} = 2\left[\Im\frac{\partial \mathbf{F}}{\partial \chi}\Im\frac{\partial \mathbf{F}}{\partial \omega} + \Im\mathbf{F}\Im\frac{\partial^2 \mathbf{F}}{\partial \omega \partial \chi} + \Re\frac{\partial \mathbf{F}}{\partial \chi}\Re\frac{\partial \mathbf{F}}{\partial \omega} + \Re\mathbf{F}\Re\frac{\partial^2 \mathbf{F}}{\partial \omega \partial \chi}\right]$$

The phase response, P, of the filter is:

$$P = \arctan\frac{\Im\mathbf{F}}{\Re\mathbf{F}}$$

The gradients of the phase response of the filter with respect to the Schur lattice filter coefficients are given by:

$$|\mathbf{F}|^2 \frac{\partial P}{\partial \chi} = \Re\mathbf{F}\Im\frac{\partial \mathbf{F}}{\partial \chi} - \Im\mathbf{F}\Re\frac{\partial \mathbf{F}}{\partial \chi}$$

The group delay response, T, of the filter is:

$$T = -\frac{\partial P}{\partial \omega}$$

so that:

$$|\mathbf{F}|^2 T = -\left[\Re\mathbf{F}\Im\frac{\partial \mathbf{F}}{\partial \omega} - \Im\mathbf{F}\Re\frac{\partial \mathbf{F}}{\partial \omega}\right]$$

The gradients of the group delay response with respect to the Schur lattice filter coefficients are given by:

$$\frac{\partial |\mathbf{F}|^2}{\partial \chi} T + |\mathbf{F}|^2 \frac{\partial T}{\partial \chi} = -\left[\Re\frac{\partial \mathbf{F}}{\partial \chi}\Im\frac{\partial \mathbf{F}}{\partial \omega} + \Re\mathbf{F}\Im\frac{\partial^2 \mathbf{F}}{\partial \chi \partial \omega} - \Im\frac{\partial \mathbf{F}}{\partial \chi}\Re\frac{\partial \mathbf{F}}{\partial \omega} - \Im\mathbf{F}\Re\frac{\partial^2 \mathbf{F}}{\partial \chi \partial \omega}\right]$$

3 Computer-Aided-Design of Schur lattice filters

The filter design procedure commences with the specification of the required filter amplitude, phase and group delay responses. The optimisation of an IIR frequency response with respect to the coefficients is not a convex problem. I hope to find a "good enough" design rather than a global optimum.

One formulation of the filter optimisation problem is to minimise the *weighted squared error* of the frequency response of $F(z)$:

$$\begin{aligned} \textbf{minimise} \quad & \mathcal{E}_F(\chi) = \int W(\omega) |F(\omega) - F_d(\omega)|^2 \, d\omega \\ \textbf{subject to} \quad & F(\omega) \text{ is stable} \end{aligned} \tag{7}$$

where χ is the coefficient vector of the filter, $F(z)$, $\mathcal{E}_F(\chi)$ is the weighted sum of the squared error, $F(\omega)$ is the filter frequency response, $W(\omega)$ is the frequency weighting and $F_d(\omega)$ is the desired filter complex frequency response. The integrand of Equation 7 could be the weighted sum of one or more of the squared-magnitude, phase or group delay responses.

The filter response optimisation problem can also be expressed as a *mini-max* optimisation problem:

$$\begin{aligned} \textbf{minimise} \quad & \max |F(\omega) - F_d(\omega)| \\ \textbf{subject to} \quad & F(\omega) \text{ is stable} \end{aligned} \tag{8}$$

I begin by finding an initial filter design that approximates the desired response, then optimising the minimum mean-squared-error of the calculated response and, if desired, finding a mini-max approximation to the desired response. Finally, I describe two methods of searching for integer valued filter coefficients: a depth-first branch-and-bound search and a successive relaxation method.

3.1 Finding an initial filter

The initial filter could be an FIR filter or a "classical" Butterworth, Chebyshev, etc. IIR filter design. *Deczky* [1] and *Richards* [17] begin with an initial filter consisting of a "by-eye" arrangement of the filter poles and zeros. *Tarczynski et al.* [4] propose minimising the filter response mean-squared-error, $\mathcal{E}_F$, weighted with a barrier function:

$$(1-\lambda)\,\mathcal{E}_F + \lambda \sum_{t=T+1}^{T+M} h^2(t) \tag{9}$$

where λ, T and M are suitable constants and $h(t)$ is the impulse response of the filter $H(z) = \frac{1}{A_N(z)}$. The barrier function (the second part of Equation 9) is intended to be small when the filter, $F(z)$, is stable and increase rapidly when the poles approach the unit circle. *Tarczynski et al.* provide heuristics for selecting λ, T and M. Typically, $\lambda \in \left[10^{-10}, 10^{-3}\right]$, $T \in [100, 500]$ and $M = NR$. *Roberts and Mullis* [24, Section 8.3] show that the state space description of the direct-form implementation of $H(z)$ is:

$$\begin{bmatrix} x(t+1) \\ y(t) \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} x(t) \\ u(t) \end{bmatrix}$$

where:

$$\begin{aligned} A &= \begin{bmatrix} 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ -a_N & -a_{N-1} & \cdots & -a_1 \end{bmatrix} \\ B &= \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \end{bmatrix}^\top \\ C &= \begin{bmatrix} -a_N & \cdots & -a_1 \end{bmatrix} \\ D &= 1 \end{aligned}$$

The corresponding impulse response is [24, Equation 8.3.21]:

$$h(t) = \begin{cases} 0 & t < 0 \\ D & t = 0 \\ CA^{t-1}B & t > 0 \end{cases}$$

3.2 Minimum-Mean-Squared-Error optimisation of IIR filters

Minimisation of the mean squared error of the filter response proceeds by successive solution of the following linear approximation problem for Δ_χ:

$$\begin{aligned} \textbf{minimise} \quad & \epsilon + \beta \\ \textbf{subject to} \quad & \|\mathcal{E}_F(\chi) + \Delta_\chi \nabla \mathcal{E}_F(\chi)\| \le \epsilon \\ & \|\Delta_\chi\| \le \beta \\ & F(\omega) \text{ is stable} \end{aligned} \tag{10}$$

After each step the new coefficient vector is $\chi + \Delta_\chi$.

In fact I replace $\mathcal{E}_F$ with an approximation to the sum at discrete frequencies of the weighted squared error of the real valued squared magnitude, phase, group delay and the gradient of the squared magnitude with respect to angular frequency. For example, the first order approximation to the squared error of the group delay, T, is:

$$\bar{\mathcal{E}}_T(\chi + \Delta_\chi) \approx \sum_\omega W_T(\omega) \left[T(\chi,\omega) + \Delta_\chi \nabla_\chi T(\chi,\omega) - T_d(\omega)\right]^2$$

with gradient with respect to χ:

$$\nabla_\chi \bar{\mathcal{E}}_T(\chi) \approx 2 \sum_\omega W_T(\omega) \left[T(\chi,\omega) - T_d(\omega)\right] \nabla_\chi T(\chi,\omega)$$

The required gradients of the state variable filter frequency response were derived in Section 2.4.

3.3 Peak-Constrained-Least-Squares optimisation of IIR filters

Linear phase FIR filter design algorithms [5, 31, 9, 11] commonly employ an "exchange algorithm" that, given an initial approximation to the desired amplitude response, finds the extremal frequencies of that response and interpolates the coefficients to obtain a new set of filter coefficients that achieves the desired ripple values at all of those frequencies. The process repeats until a satisfactory mini-max amplitude response is found. I have successfully solved the IIR mini-max problem of Equation 8 with a modified version of the "Peak-Constrained-Least-Squares" (PCLS) algorithm of *Selesnick*, *Lang* and *Burrus* [11, p.498]. They describe the exchange of amplitude response, $A(\omega)$, constraints as follows:

> After each iteration, the algorithm checks the values of $A(\omega)$ over the previous constraint set frequencies. ... However, if it is found that $A(\omega)$ violates the constraints at some frequency belonging to the previous constraint set, R, then i) that frequency where the violation is greatest is appended to the current constraint set, S, and ii) the same frequency is removed from the record of previous constraint set frequencies, R.

Figure 4 shows a flow-chart of the modified *Selesnick*, *Lang* and *Burrus* exchange algorithm.

Similarly to $\bar{\mathcal{E}}_T$, the PCLS constraints for the group delay response, $T(\chi, \omega)$, are approximated to first order by:

$$T_l(\omega) \leq T(\chi, \omega) + \Delta_\chi \nabla_\chi T(\chi, \omega) \leq T_u(\omega)$$

where $T_l(\omega)$ and $T_u(\omega)$ are the lower and upper constraints on the group delay.

The modified PCLS mini-max optimisation is:

$$\begin{aligned} \textbf{minimise} \quad & \epsilon + \beta \\ \textbf{subject to} \quad & \|\bar{\mathcal{E}}_F(\chi) + \Delta_\chi \nabla \bar{\mathcal{E}}_F(\chi)\| \leq \epsilon \\ & \|\Delta_\chi\| \leq \beta \\ & F(\omega) \text{ is stable} \\ & \text{PCLS response constraints are satisfied} \end{aligned} \tag{11}$$

3.4 Second Order Cone Programming

Alizadeh and *Goldfarb* [6] describe Second Order Cone Programming (SOCP) as the solution of a class of convex optimisation problems in which a linear function is minimised subject to a set of conic constraints. In the following example I use the *SeDuMi* (*Se*lf-*Du*al-*Mi*nimisation) SOCP solver originally written by *Jos Sturm* [27]. *Lu* [33, Section III] provides an example of expressing an optimisation problem of Equation 11 in the form accepted by SeDuMi. In *Lu*'s notation the problem is:

$$\textbf{minimise} \quad \boldsymbol{b}^\top \boldsymbol{x} \tag{12a}$$

$$\textbf{subject to} \quad \|\boldsymbol{A}_i^\top \boldsymbol{x} + \boldsymbol{c}_i\| \leq \boldsymbol{b}_i^\top \boldsymbol{x} + d_i \quad \text{for } i = 1, \ldots, q \tag{12b}$$

$$\boldsymbol{D}^\top \boldsymbol{x} + \boldsymbol{f} \geq \boldsymbol{0} \tag{12c}$$

where $\boldsymbol{x} \in \mathbb{R}^{m\times 1}$, $\boldsymbol{b} \in \mathbb{R}^{m\times 1}$, $\boldsymbol{A}_i \in \mathbb{R}^{m\times(n_i-1)}$ [1], $\boldsymbol{c}_i \in \mathbb{R}^{(n_i-1)\times 1}$, $\boldsymbol{b}_i \in \mathbb{R}^{m\times 1}$, $d_i \in \mathbb{R}$ for $1 \leq i \leq q$, $\boldsymbol{D} \in \mathbb{R}^{m\times p}$ and $\boldsymbol{f} \in \mathbb{R}^{p\times 1}$. The problem is cast into SeDuMi format by defining:

$$\begin{aligned} \boldsymbol{A}_t &= \begin{bmatrix} -\boldsymbol{D} & \boldsymbol{A}_t^{(1)} \ldots & \boldsymbol{A}_t^{(q)} \end{bmatrix} \\ \boldsymbol{A}_t^{(i)} &= -\begin{bmatrix} \boldsymbol{b}_i & \boldsymbol{A}_i \end{bmatrix} \\ \boldsymbol{b}_t &= -\boldsymbol{b} \\ \boldsymbol{c}_t &= \begin{bmatrix} \boldsymbol{f}; & \boldsymbol{c}_t^{(1)}; & \ldots & \boldsymbol{c}_t^{(q)} \end{bmatrix} \\ \boldsymbol{c}_t^{(i)} &= \begin{bmatrix} d_i; & \boldsymbol{c}_i \end{bmatrix} \end{aligned}$$

For Equation 11, $\boldsymbol{x} = [\epsilon, \beta, \chi]$, Equation 12b applies the conic constraints on the linear approximation to the minimum mean squared error of the frequency response and Equation 12c applies the linear constraints on both the reflection coefficients, $|k_n| < 1$, and the PCLS frequency response constraints.

[1] $n_i - 1$ to allow for the column b_i in the matrix $\boldsymbol{A}_t^{(i)}$ and d_i in the column vector $\boldsymbol{c}_t^{(i)}$

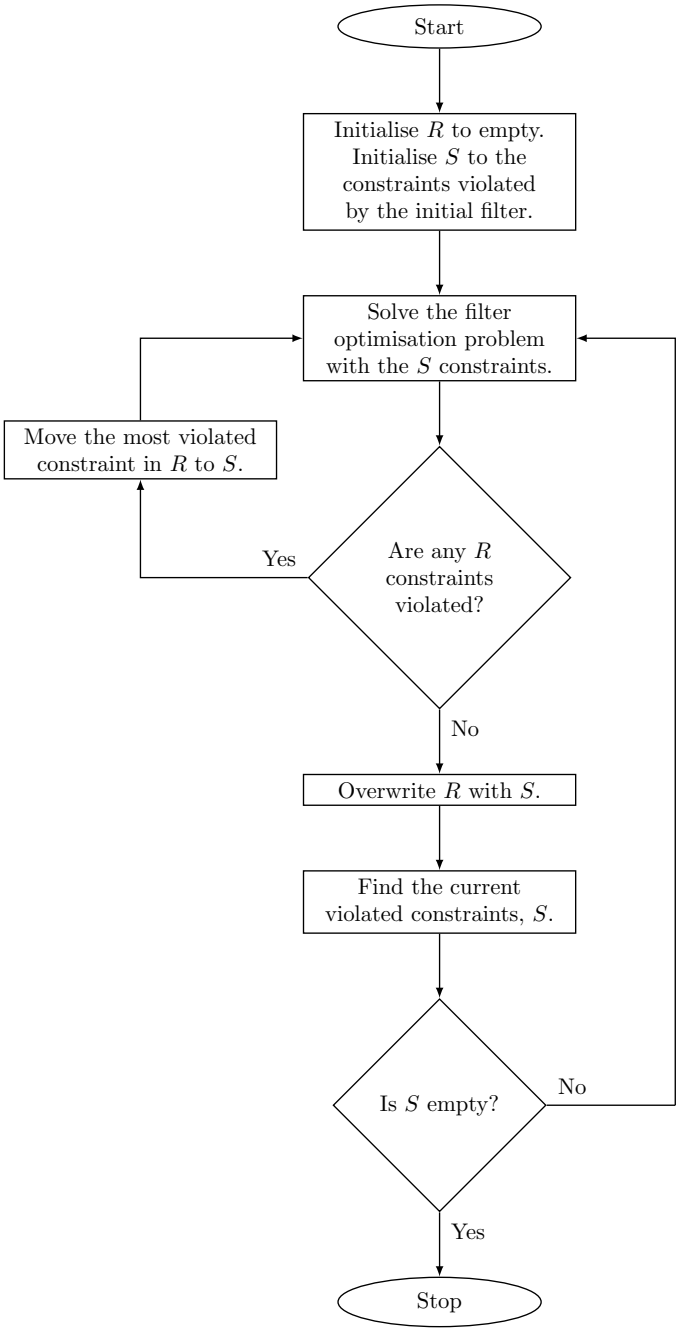


Figure 4: Modified Selesnick, Lang and Burrus exchange algorithm [11, p.498].

4 Design of digital filters with signed-digit coefficients

4.1 The canonical signed-digit representation of filter coefficients

Parhi [15, Section 13.6] lists the following properties of the *canonical* binary signed-digit (CSD) representation of a binary signed-digit number $A = a_{W-1}a_{W-2}\cdots a_1a_0$ where each $a_i \in \{-1, 0, 1\}$:

- no 2 consecutive bits in a CSD number are non-zero
- the CSD representation of a number contains the miniumum possible number of non-zero bits, thus the name *canonic*
- the CSD representation of a number is unique

The first property breaks the carry chain when adding two CSD numbers. *Parhi* [15, Section 13.6.1] shows an algorithm that calculates the *canonical* binary signed-digit representation from the two's complement representation, reproduced here as Algorithm 4.1.

Algorithm 4.1 Conversion of 2's complement numbers to the canonical signed-digit representation (*Parhi* [15, Section 13.6.1]).

Denote the two's complement representation of the number A as $A = \hat{a}_{W-1}\hat{a}_{W-2}\cdots\hat{a}_1\hat{a}_0$.
Denote the CSD representation of A as $A = a_{W-1}a_{W-2}\cdots a_1a_0$.

$\hat{a}_{-1} = 0$
$\gamma_{-1} = 0$
$\hat{a}_W = \hat{a}_{W-1}$
for $k = 0, \ldots, W-1$ **do**
  $\theta_i = \hat{a}_i \oplus \hat{a}_{i-1}$
  $\gamma_i = \bar{\gamma}_{i-1}\theta_i$
  $a_i = (1 - 2\hat{a}_{i+1})\,\gamma_i$
end for

Lim et al. [34] describe a method of allocating a limited number of signed power-of-two digit terms to the fixed point coefficients of a digital filter. The method is based on the belief that "*allocating the SPT terms in such a way that all the coefficient values have the same quantisation step-size to coefficient sensitivity ratio will lead to a good design*".

Lim et al. first prove properties of the canonical signed-digit representation [34, Section II]. In particular:

Property 1: Define S_Q as the set of contiguous integers that can be represented by up to Q signed digits. The largest integer in S_Q is $J_Q = \sum_{l=0}^{Q-1} 2^{2l+1}$.

Property 5: On average, $0.72Q$ signed-digits are required to represent the integers in S_Q.

Lim et al. show that an estimate of the number of signed digits, Q, required to represent J_Q is:

$$Q \approx \frac{1}{2}\log_2 J_Q + 0.31$$

Replacing J_Q by an integer $n \in S_Q$, an estimate of the average number of terms, Q_A, required to represent n is:

$$\begin{aligned} Q_A &\approx 0.72Q \\ &\approx 0.36\log_2 n + 0.22 \end{aligned}$$

Now suppose that D signed digits are available to represent two positive integers n_1 and n_2. If $n_1 \approx n_2$ then each integer is allocated $\frac{D}{2}$ bits. If $n_1 > n_2$ then *Lim et al.* argue that the number of additional signed-digits, Q_E, required to represent n_1 is:

$$Q_E \approx 0.36\log_2 \lfloor \frac{n_1}{n_2} \rfloor$$

where $\lfloor\ \rfloor$ represents the integer part. In general, Q_E is not an integer.

Lim et al. go on to consider the allocation of signed digits to the coefficients of a symmetric FIR filter. The change in the frequency response, $F(\omega)$, of a filter due to a change $\Delta\chi_n$ in coefficient χ_n is:

$$\Delta F(\omega, n) \approx \frac{\partial F(\omega)}{\partial \chi_n}\Delta\chi_n$$

Lim et al. use the average of the coefficient sensitivity to define a cost for the n'th coefficient:

$$cost_n = 0.36 \log_2 |\chi_n| + 0.36 \log_2 \int_0^\pi \left| \frac{\partial F(\omega)}{\partial \chi} \right| d\omega$$

Given a total of D signed-digits, *Lim et al.* assign a single signed-digit at a time to the coefficient with the largest cost. After a coefficient is given a signed-digit, its cost is decreased by one. The process is repeated until all D digits have been allocated.

4.2 Branch-and-bound search for the filter coefficients

Given the K floating point coefficients, χ, of a filter, an exhaustive search of the upper and lower bounds on the integer or signed-digit approximations to these coefficients would require $\mathcal{O}\left(2^K\right)$ comparisons of the corresponding filter approximation error. *Branch-and-bound* [3], [25, p.627] is a heuristic for reducing the number of branches searched in a binary decision tree. At each branch of the binary tree the solution is compared to the estimated lower bounds on the cost of the full path proceeding from that branch. If the cost of that full path is greater than that of the best full path found so far then further search on that path is abandoned. Figure 5 shows a flow diagram of an implementation of the algorithm using a stack. The floating point filter coefficients, χ, are approximated by the signed-digit coefficients, $\bar{\chi}$. Each coefficient, χ_n and $\bar{\chi}_n$, is bounded by the corresponding signed-digit numbers u_n and l_n so that $l_n \leq \chi_n \leq u_n$ and $l_n \leq \bar{\chi}_n \leq u_n$. The search for the set of coefficients with minimum cost is "depth-first", starting at the root of the search tree and fixing successive coefficients. *Ito et al.* [26] recommend choosing, at each branch, the χ_n with the greatest difference $u_n - l_n$. The two sub-problems at that branch fix $\bar{\chi}_n$ to l_n and u_n. One of the two sub-problems is pushed onto a stack and the other is solved and the cost calculated. I assume that this cost is the least possible for the remaining coefficients on the current branch. If the cost of the current sub-problem is greater than the current minimum cost then the current branch is abandoned and a new sub-problem is popped off the problem stack. Otherwise, if the search has reached the maximum depth of the tree then the current solution is a new signed-digit minimum cost solution. If the current cost is less than the minimum cost and the search has not reached the maximum depth then the search continues with a new branch.

4.3 Successive relaxation search for the filter coefficients

A successive relaxation search for the filter coefficients fixes one coefficient and optimises the filter frequency response with respect to the remaining free coefficients. This is called a *relaxation* of the optimisation. The relaxation is repeated until all the coefficients are fixed. At each coefficient relaxation step the script finds the upper and lower signed-digit approximations to the current set of active coefficients and selects the coefficient with the largest difference in those approximations.

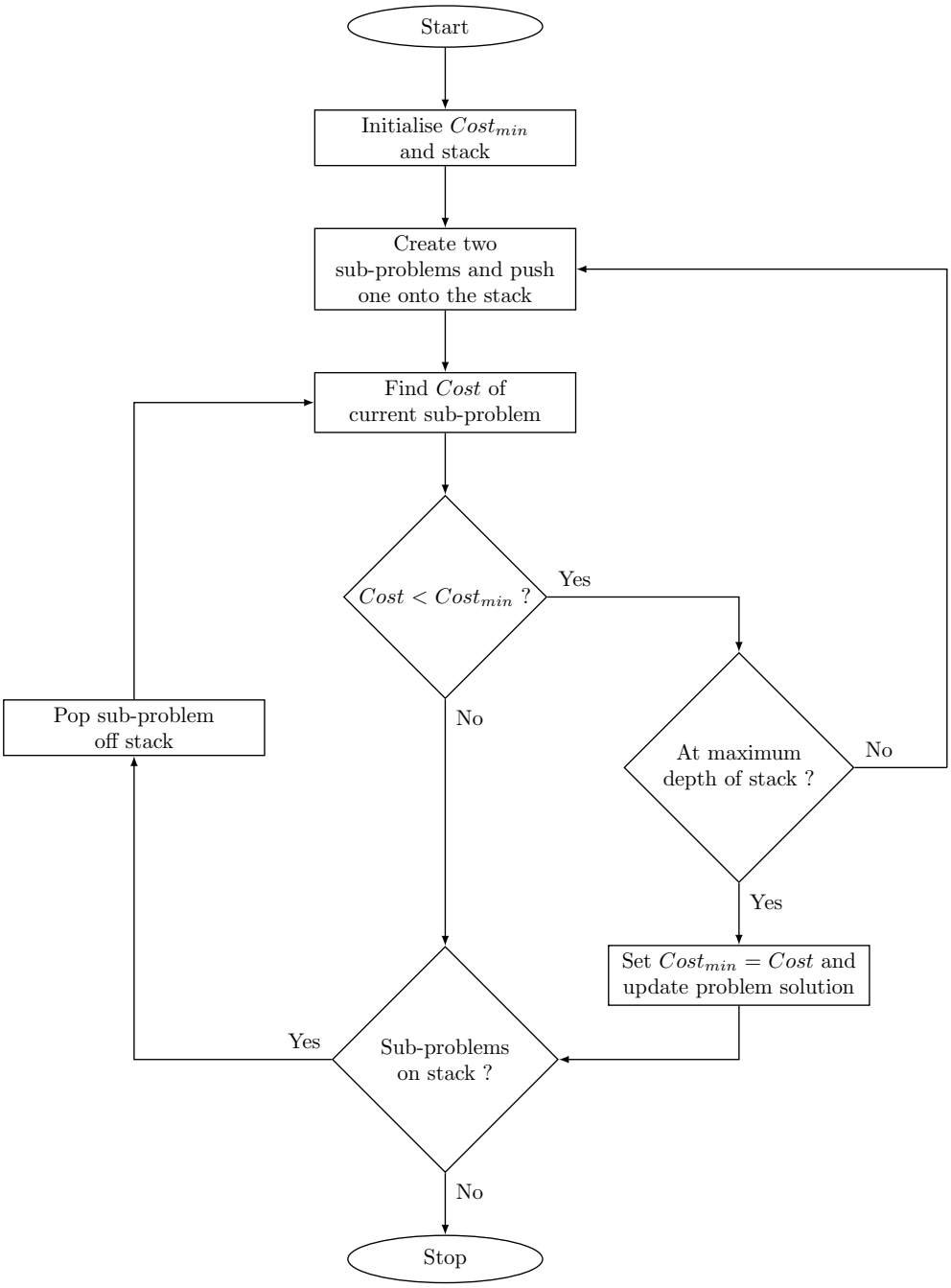


Figure 5: Branch-and-bound depth-first search algorithm

5 An example : design of a low pass differentiator filter

This section describes the design of a low pass differentiator filter implemented as $F_0(z) = 1 - z^{-1}$ in series with a tapped Schur one-multiplier lattice correction filter, $C(z)$, having denominator polynomial coefficients only for terms in z^{-2n}. In the pass band, the desired complex frequency response of a differentiator filter is:

$$F_d(\omega) = -\imath\frac{\omega}{2}e^{-\imath\omega t_p}$$

where t_p is the nominal filter pass band group delay in samples. The squared-magnitude response of the correction filter is:

$$|C(\omega)|^2 = \frac{|F_d(\omega)|^2}{|F_0(\omega)|^2}$$

where:

$$\begin{aligned}
|F_d(\omega)| &= \frac{\omega}{2} \\
|F_d(\omega)|^2 &= \frac{\omega^2}{4} \\
\frac{d|F_d(\omega)|^2}{d\omega} &= \frac{\omega}{2} = |F_d(\omega)|
\end{aligned}$$

and the response of the zero at $z = 1$ is:

$$\begin{aligned}
F_0(\omega) &= 1 - e^{\imath\omega} \\
&= 2\imath e^{\imath\frac{\omega}{2}}\sin\frac{\omega}{2} \\
|F_0(\omega)| &= 2\sin\frac{\omega}{2} \\
\frac{d|F_0(\omega)|^2}{d\omega} &= 2|F_0(\omega)|\frac{d|F_0(\omega)|}{d\omega} \\
&= 2\sin\omega
\end{aligned}$$

By the chain rule for differentiation, the gradient of $|F(\omega)|^2$ is:

$$\frac{d|F_d(\omega)|^2}{d\omega} = |F_0(\omega)|^2\frac{d|C(\omega)|^2}{d\omega} + \frac{|F_d(\omega)|^2}{|F_0(\omega)|^2}\frac{d|F_0(\omega)|^2}{d\omega}$$

Substituting and rearranging to obtain the desired gradient of $|C(\omega)|^2$ in terms of the desired overall squared amplitude response, $|F_d(\omega)|^2$, and the squared magnitude response of the zero at $z = 1$, $|F_0(\omega)|^2$:

$$\frac{d|C(\omega)|^2}{d\omega} = |F_d(\omega)|\left[\frac{1 - |F_d(\omega)|\cot\frac{\omega}{2}}{|F_0(\omega)|^2}\right]$$

5.1 Initial R=2 low pass differentiator filter

Figure 6 shows the frequency response of the initial R=2 low pass differentiator filter found by the method of *Tarczynski et al.*, described in Section 3.1. The correction filter order is $N = 10$, filter pass band and stop band edge frequencies are 0.2 and 0.4 (normalised to the digital sample rate), the nominal pass band phase is 1.5π radians (adjusted for delay) and the nominal pass band group delay is 9 samples.

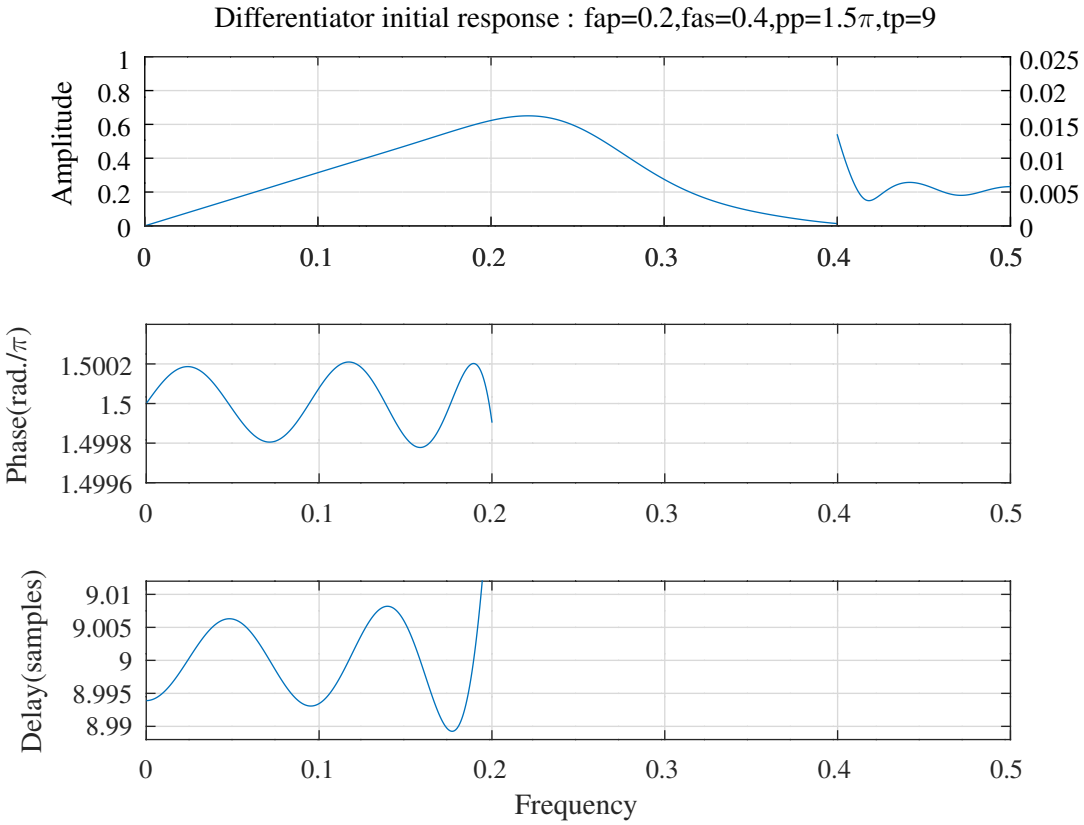


Figure 6: Amplitude, phase and group delay responses of an initial low pass differentiator filter composed of a Schur one-multiplier lattice correction filter with a denominator polynomial having terms only in z^{-2} in series with $1 - z^{-1}$.

The coefficients of the numerator and denominator polynomials of the initial correction filter transfer function are:

```
N0 = [  -0.0023668408,   0.0007550252,   0.0057495588,  -0.0017253427, ...
        -0.0145883151,   0.0048775476,   0.0447167151,  -0.0377320271, ...
        -0.2353842769,  -0.2685554589,  -0.1040317225 ]';


D0R = [   1.0000000000,   0.0000000000,   0.2381495222,  -0.0000000000, ...
         -0.0287006727,   0.0000000000,   0.0076755310,  -0.0000000000, ...
         -0.0017192596,  -0.0000000000,  -0.0001444401 ]';
```

The corresponding Schur one-multiplier lattice coefficients of the initial correction filter are:

```
k0 = [   0.0000000000,   0.2459545571,   0.0000000000,  -0.0306726072, ...
         0.0000000000,   0.0080726571,   0.0000000000,  -0.0016848613, ...
         0.0000000000,  -0.0001444401 ]';


epsilon0 = [  -0.0000000000,   1.0000000000,  -0.0000000000,   1.0000000000, ...
              -0.0000000000,  -1.0000000000,  -0.0000000000,   1.0000000000, ...
              -0.0000000000,   1.0000000000 ];


c0 = [  -0.0347172234,  -0.2163958123,  -0.2579098535,  -0.0407075228, ...
         0.0492516660,   0.0054064057,  -0.0161893453,  -0.0019086425, ...
         0.0063141328,   0.0007551343,  -0.0023668408 ]';
```

5.2 PCLS design of the R=2 low pass differentiator filter

The low pass differentiator correction filter was PCLS optimised with amplitude pass band error peak-to-peak ripple of 0.0009, amplitude stop band peak ripple of 0.007, phase pass band peak-to-peak ripple of 0.0002π radians, group delay pass band peak-to-peak ripple of 0.006 samples and correction filter gradient of amplitude-squared pass band error peak-to-peak ripple of 0.013.

The Schur one-multiplier lattice coefficients of the PCLS SOCP optimised correction filter are:

```
k2 = [   0.0000000000,   0.2065205470,   0.0000000000,  -0.0281020445, ...
         0.0000000000,   0.0093023973,   0.0000000000,  -0.0038787444, ...
         0.0000000000,   0.0012820857 ]';


epsilon2 = [  -0.0000000000,   1.0000000000,  -0.0000000000,   1.0000000000, ...
              -0.0000000000,  -1.0000000000,  -0.0000000000,   1.0000000000, ...
              -0.0000000000,  -1.0000000000 ]';


c2 = [  -0.0212083759,  -0.2188914462,  -0.2870345994,  -0.0306418043, ...
         0.0731356951,  -0.0170264555,  -0.0231060228,   0.0170587049, ...
         0.0012358212,  -0.0062679753,   0.0022181847 ]';
```

The corresponding correction filter numerator and denominator polynomial coefficients are:

```
N2 = [   0.0022181847,  -0.0062599392,   0.0016787930,   0.0157161394, ...
        -0.0227974703,  -0.0132156271,   0.0674605209,  -0.0332183997, ...
        -0.2598747453,  -0.2583845612,  -0.0835347840 ];


D2 = [   1.0000000000,   0.0000000000,   0.2004144265,   0.0000000000, ...
        -0.0261222080,   0.0000000000,   0.0084913768,   0.0000000000, ...
        -0.0036217895,   0.0000000000,   0.0012820857 ];
```

Figure 7 shows the amplitude error, phase and group delay response errors of the $R = 2$ low pass differentiator filter after PCLS SOCP optimisation. Figure 8 shows the error in the gradient of the squared-magnitude response of the $R = 2$ correction filter and of the low pass differentiator filter after PCLS SOCP optimisation. Figure 9 shows the pole-zero plot of the $R = 2$ low pass differentiator filter. Figure 10 shows the gradients of the correction filter squared-magnitude, phase and delay with respect to the coefficients of the direct form implementation of the filter. Figure 11 shows the gradients of the correction filter squared-magnitude, phase and delay with respect to the coefficients of the Schur lattice implementation of the filter.

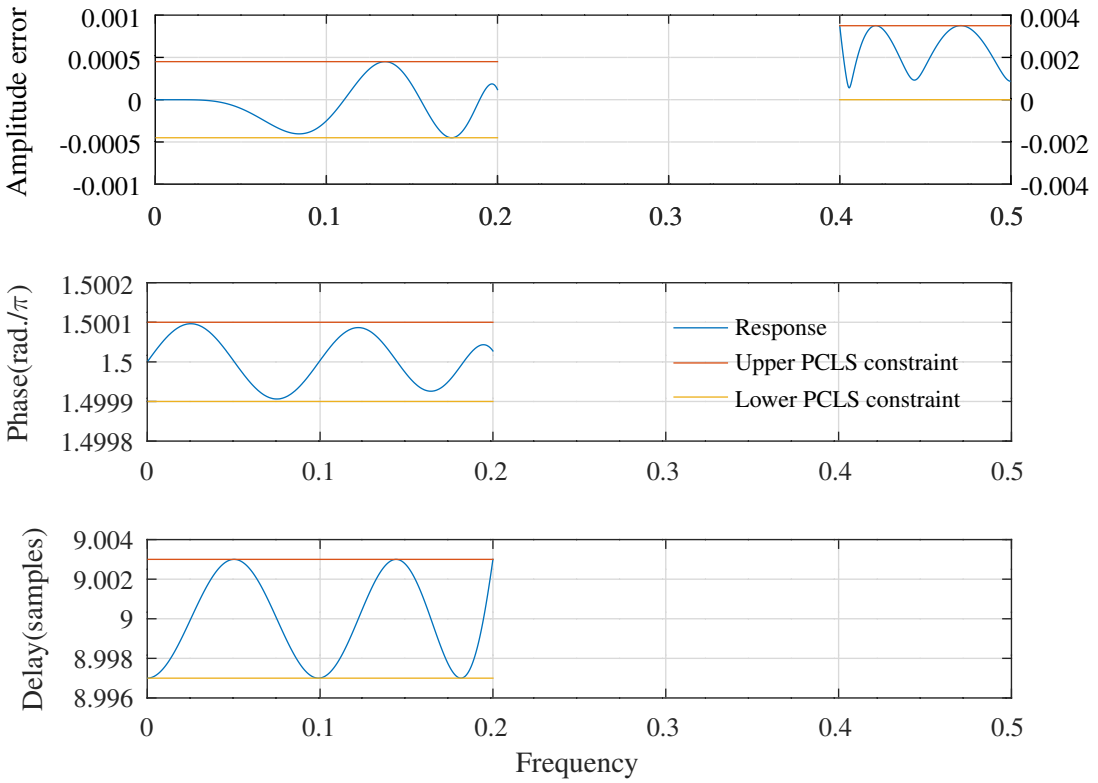


Figure 7: PCLS optimised amplitude error, phase and group delay responses of a low pass differentiator filter composed of a Schur one-multiplier lattice correction filter with a denominator polynomial having terms only in z^{-2} in series with $1 - z^{-1}$.

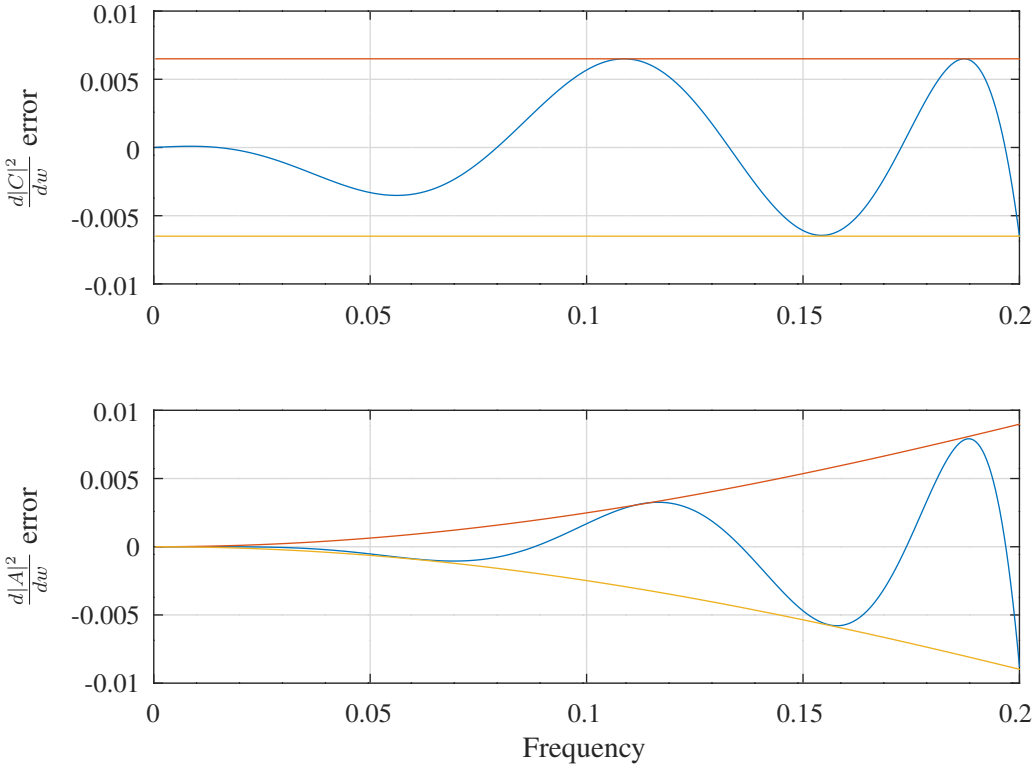


Figure 8: PCLS optimised pass band error of the gradient of the squared magnitude response of the correction filter and that of the differentiator filter.

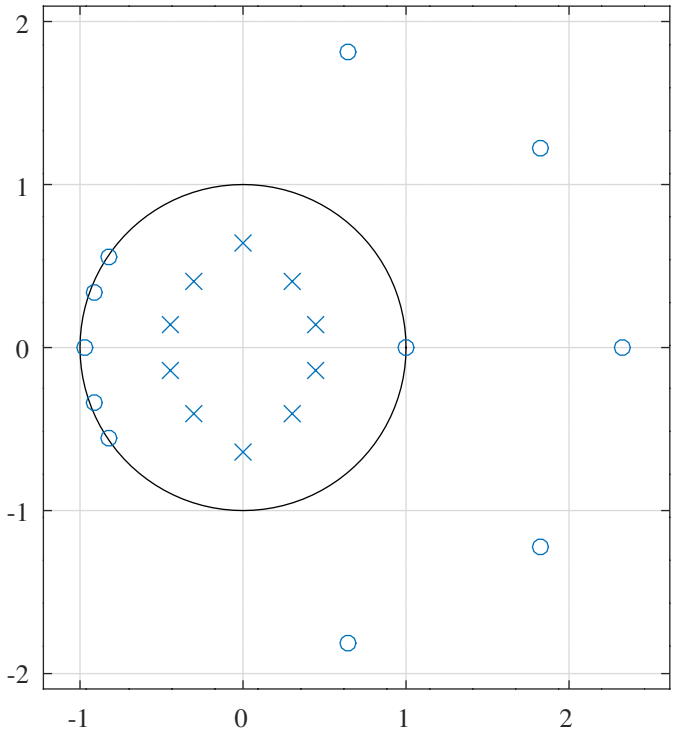


Figure 9: Pole-zero plot of the PCLS optimised low pass differentiator filter composed of a Schur one-multiplier lattice correction filter with a denominator polynomial having terms only in z^{-2} in series with $1 - z^{-1}$.

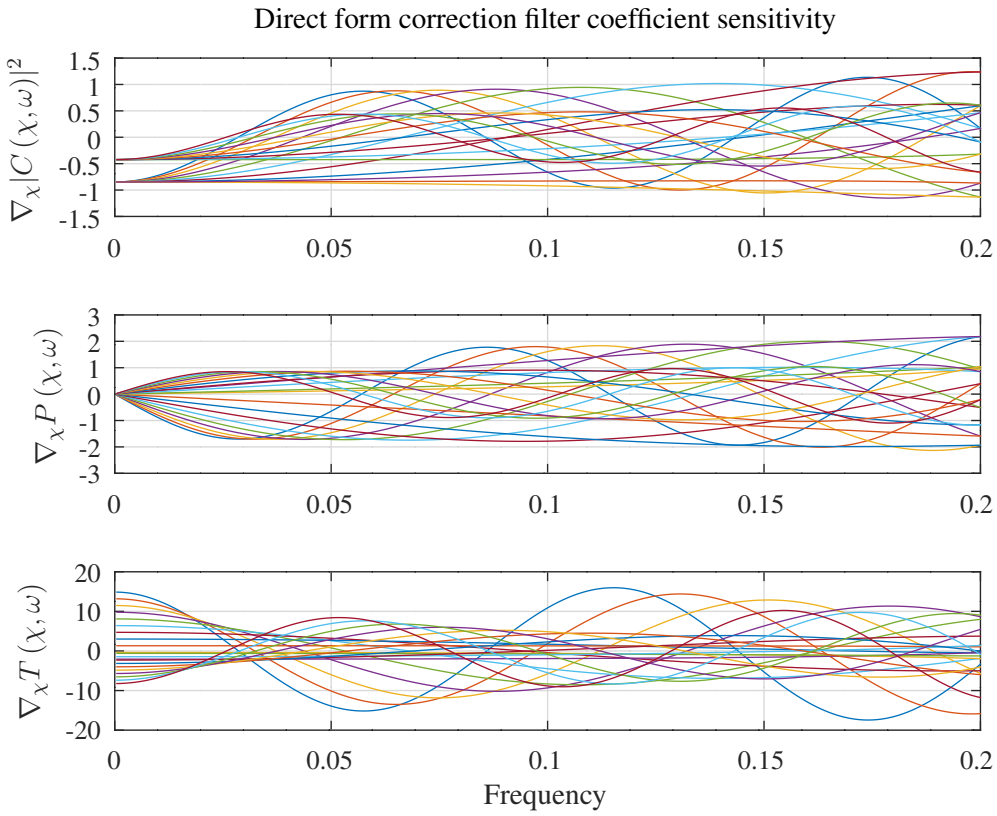


Figure 10: Gradients of the pass band response with respect to the coefficients of the direct form implementation of the PCLS optimised low pass differentiator correction filter.

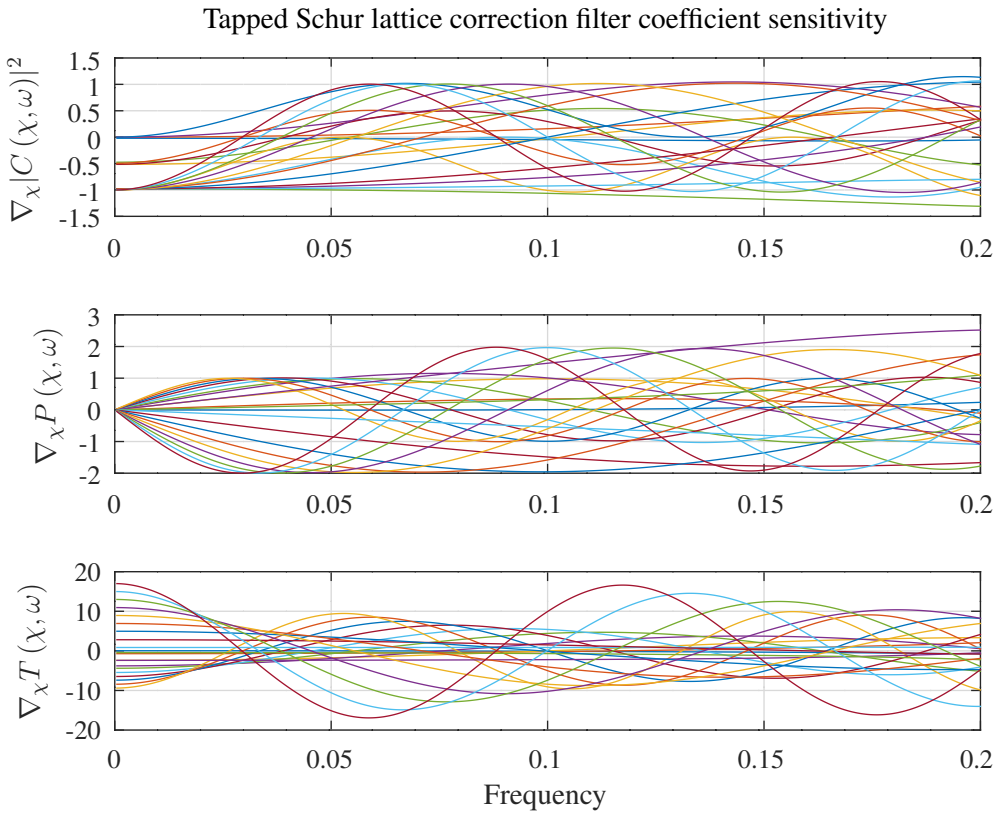


Figure 11: Gradients of the pass band response with respect to the coefficients of the tapped Schur lattice implementation of the PCLS optimised low pass differentiator correction filter.

5.3 Search for the signed-digit coefficients of the R=2 low pass differentiator filter

When truncated to 12 bits and 3 signed-digits the Schur one-multiplier lattice coefficients of the PCLS optimised correction filter are:

```
k0_sd_no_alloc = [        0,      416,        0,      -58, ...
                          0,       19,        0,       -8, ...
                          0,        3 ]'/2048;


c0_sd_no_alloc = [      -44,     -448,     -592,      -63, ...
                        152,      -35,      -47,       35, ...
                          3,      -13,        5 ]'/2048;
```

The numbers of signed-digits that the heuristic of *Lim et al.* allocates to each cofficient of the PCLS optimised correction filter are:

```
k_allocsd_digits = [ 0, 3, 0, 3, ...
                     0, 2, 0, 3, ...
                     0, 3 ]';


c_allocsd_digits = [ 3, 4, 4, 3, ...
                     4, 2, 3, 3, ...
                     2, 3, 3 ]';
```

The signed-digit coefficients of the PCLS optimised filter with the numbers of signed-digits allocated by the heuristic of *Lim et al.* are:

```
k0_sd = [        0,      416,        0,      -58, ...
                 0,       20,        0,       -8, ...
                 0,        3 ]'/2048;


c0_sd = [      -44,     -448,     -588,      -63, ...
               150,      -36,      -47,       35, ...
                 3,      -13,        5 ]'/2048;
```

The signed-digit coefficients of the PCLS optimised filter with the numbers of signed-digits allocated by the heuristic of *Lim et al.* and found with branch-and-bound search are:

```
k_min = [        0,      420,        0,      -57, ...
                 0,       18,        0,       -8, ...
                 0,        2 ]'/2048;


c_min = [      -44,     -448,     -588,      -62, ...
               149,      -35,      -47,       35, ...
                 2,      -13,        5 ]'/2048;
```

The corresponding correction filter transfer function polynomials found with branch-and-bound search are:

```
N_min = [   0.0024414062,  -0.0063414574,   0.0014616626,   0.0157439326, ...
           -0.0227064177,  -0.0133058291,   0.0672092889,  -0.0328849775, ...
           -0.2604138131,  -0.2580922194,  -0.0834874571 ];


D_min = [   1.0000000000,   0.0000000000,   0.1990876198,   0.0000000000, ...
           -0.0259700550,   0.0000000000,   0.0079858585,   0.0000000000, ...
           -0.0037118248,   0.0000000000,   0.0009765625 ];
```

The signed-digit coefficients of the PCLS optimised filter with the numbers of signed-digits allocated by the heuristic of *Lim et al.* and found with SOCP relaxation search are:

```
k_min = [         0,       416,         0,       -57, ...
                  0,        18,         0,        -7, ...
                  0,         2 ]'/2048;


c_min = [       -44,      -449,      -588,       -63, ...
                148,       -34,       -48,        35, ...
                  2,       -12,         4 ]'/2048;
```

The corresponding correction filter transfer function polynomials found with SOCP relaxation search are:

```
N_min = [   0.0019531250,  -0.0058536530,   0.0013607526,   0.0158604764, ...
           -0.0231929742,  -0.0128753752,   0.0667041521,  -0.0332393313, ...
           -0.2607570166,  -0.2584290613,  -0.0829409967 ];


D_min = [   1.0000000000,   0.0000000000,   0.1971936226,   0.0000000000, ...
           -0.0259993193,   0.0000000000,   0.0080895491,   0.0000000000, ...
           -0.0032253936,   0.0000000000,   0.0009765625 ];
```

Table 1 compares, for each filter, a cost function, the maximum pass band and stop band response errors, the total number of signed-digits required by the 12-bit coefficients and the number of shift-and-add operations required to implement the correction filter coefficient multiplications of the correction filter.

	Correction filter cost	Max. pass amp. error	Max. stop amp. error	Max. pass phase error (rad./π)	Max. pass delay error (samples)	12-bit signed digits	Shift-and-adds
Floating-point	2.69e-05	4.50e-04	3.51e-03	9.64e-05	3.00e-03		
Signed-Digit	1.05e-04	1.21e-03	8.39e-03	9.61e-04	1.61e-02	41	25
Signed-Digit(Lim)	1.83e-04	7.08e-03	8.23e-03	1.50e-03	2.35e-02	38	22
Branch-and-bound	3.65e-05	6.17e-04	4.84e-03	2.78e-04	8.83e-03	41	25
SOCP-relaxation	3.22e-05	6.22e-04	6.24e-03	1.96e-04	4.64e-03	37	21

Table 1: Comparison of the response errors and the total number of signed digits in the 12-bit coefficients, each having an average of 3-signed digits, allocated by the method of *Lim et al.*, required for a tapped Schur one-multiplier lattice, R=2, low pass differentiator correction filter found by branch-and-bound and SOCP relaxation search.

Figure 12 compares the pass band and stopband error amplitude response of each filter for 12-bit 3-signed-digit coefficients and 12-bit coefficients with an average of 3-signed-digits allocated by the method of *Lim et al.* found with branch-and-bound search and SOCP relaxation search. Figures 13, 14 and 15 and compare the corresponding pass band phase, group-delay and gradient of the squared magnitude responses of each filter.

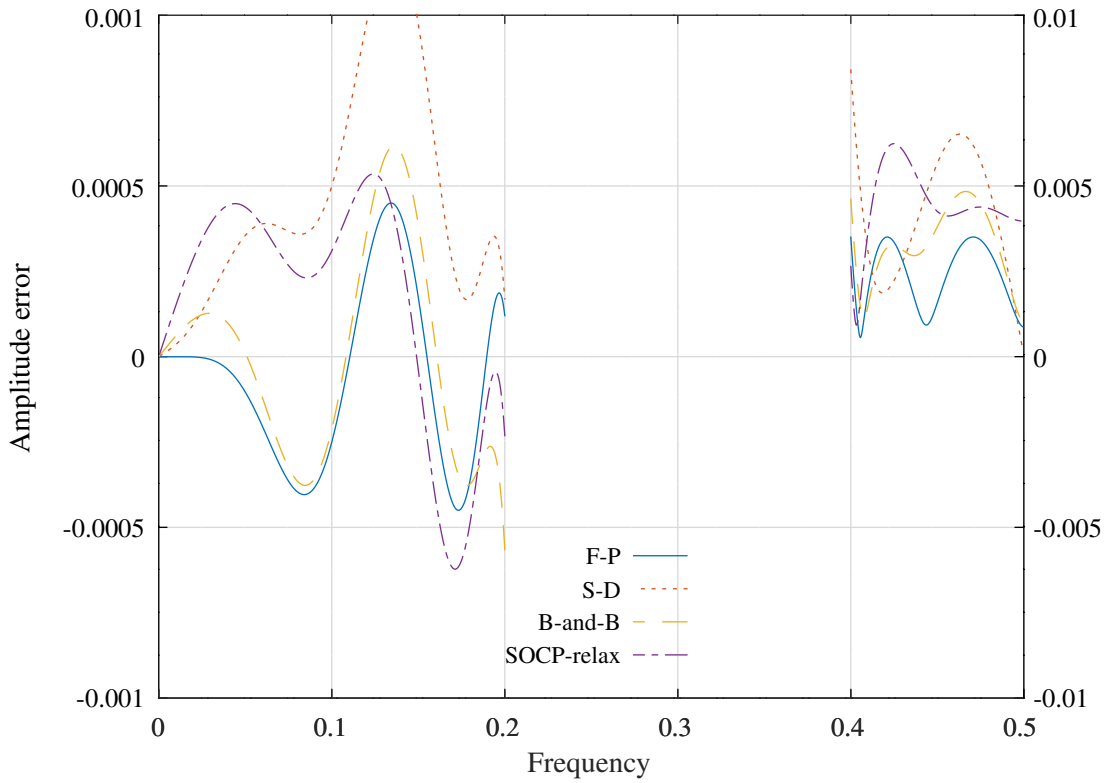


Figure 12: Comparison of the pass band and stop band amplitude response errors of a tapped Schur one-multiplier, R=2, lattice lowpass differentiator correction filter with floating point coefficients, 12-bit 3-signed-digit coefficients, and 12-bit signed-digit coefficients each having an average of 3-signed-digits allocated by the method of *Lim et al.*, found by branch-and-bound and SOCP relaxation search.

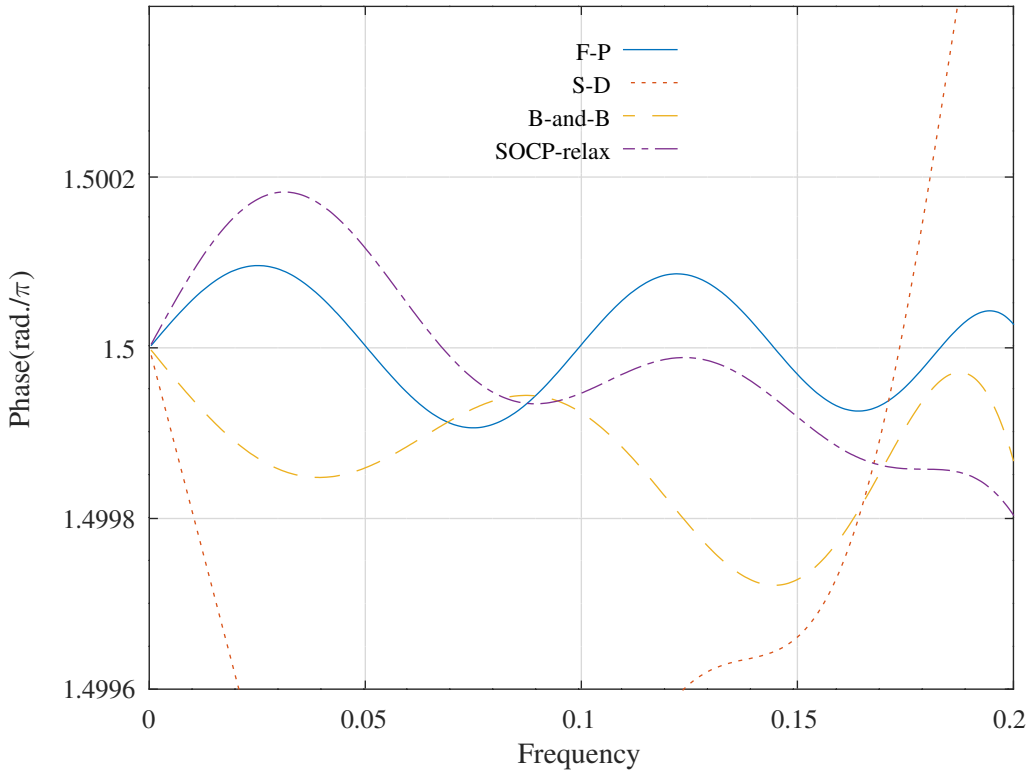


Figure 13: Comparison of the pass band phase response errors of a tapped Schur one-multiplier, R=2, lattice low pass differentiator correction filter with floating point coefficients, 12-bit 3-signed-digit coefficients, and 12-bit signed-digit coefficients each having an average of 3-signed-digits allocated by the method of *Lim et al.*, found by branch-and-bound and SOCP relaxation search.

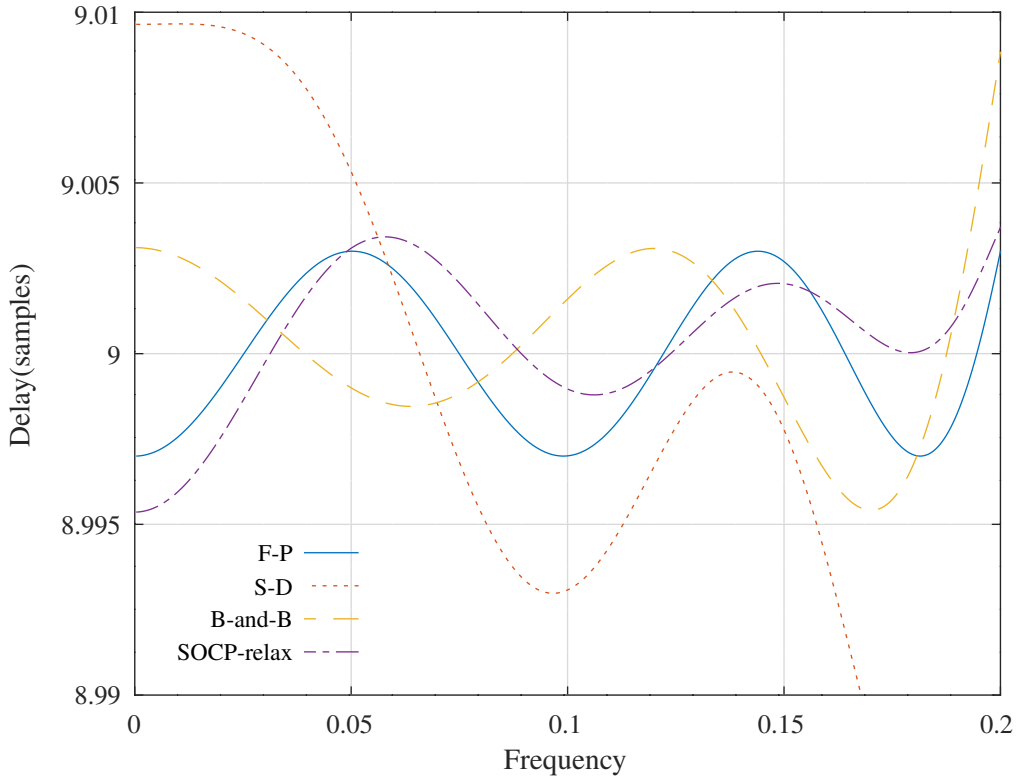


Figure 14: Comparison of the pass band group delay response errors of a tapped Schur one-multiplier, R=2, lattice low pass differentiator correction filter with floating point coefficients, 12-bit 3-signed-digit coefficients, and 12-bit signed-digit coefficients each having an average of 3-signed-digits allocated by the method of *Lim et al.*, found by branch-and-bound and SOCP relaxation search.

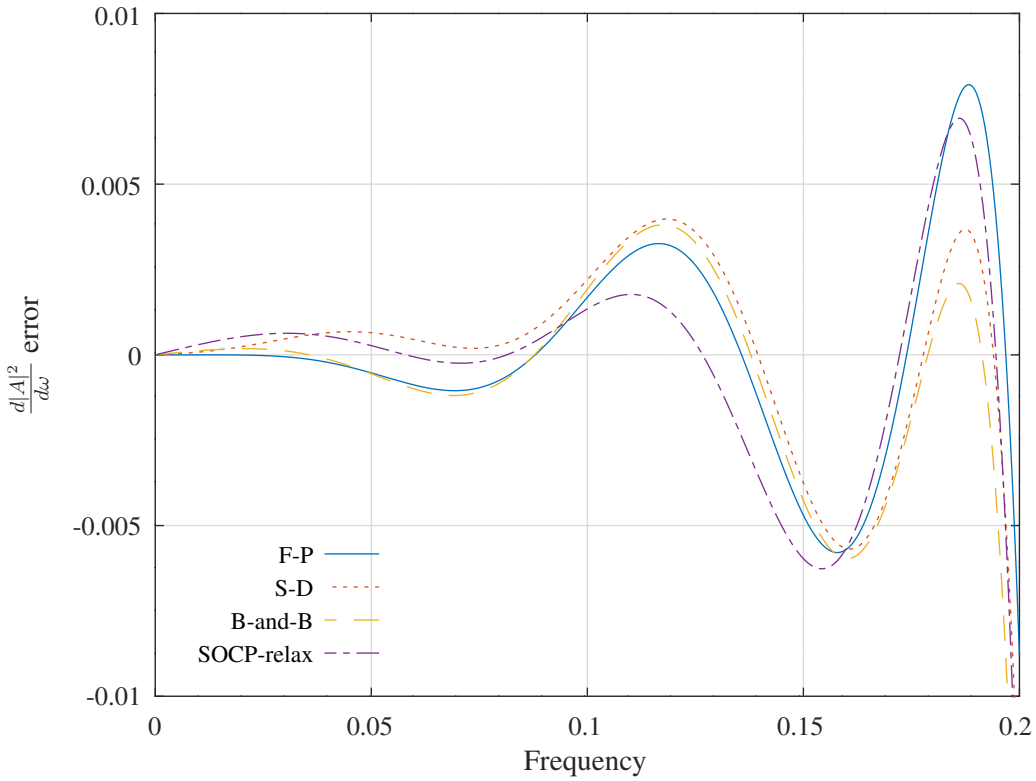


Figure 15: Comparison of the error in the pass band gradient of the squared magnitude response of a tapped Schur one-multiplier, R=2, lattice low pass differentiator correction filter with floating point coefficients, 12-bit 3-signed-digit coefficients, and 12-bit signed-digit coefficients each having an average of 3-signed-digits allocated by the method of *Lim et al.*, found by branch-and-bound and SOCP relaxation search.

6 Summary

The Schur lattice digital filter has a simple stability constraint. This article demonstrates the realisation of an IIR digital filter transfer function with stability and frequency response constraints by optimising the floating point and fixed point coefficients of a tapped one-multiplier Schur lattice digital filter.

References

[1] A. G. Deczky. Synthesis of recursive digital filters using the minimum p-error criterion. *IEEE Transactions on Audio and Electroacoustics*, 20:257–263, October 1972. 1, 6, 9

[2] A. H. Gray and J. D. Markel. Digital Lattice And Ladder Filter Synthesis. *IEEE Transactions on Audio and Electroacoustics*, 21(6):491–500, December 1973. 1, 2, 4

[3] A. H. Land and A. G. Doig. An Automatic Method Of Solving Discrete Programming Problems. *Econometrica*, 28(3):497–520, July 1960. 13

[4] A. Tarczynski, G. Cain, E. Hermanowicz and M. Rojewski. A WISE Method for Designing IIR Filters. *IEEE Transactions on Signal Processing*, 49(7):1421–1432, July 2001. 9

[5] E. M. Hofstetter. A New Technique for the Design of Non-Recursive Digital Filters, December 1970. Massachusetts Institute of Technology, Lincoln Laboratory, Technical Note 1970-42. 10

[6] F. Alizadeh and D. Goldfarb. Second-Order Cone Programming. *Mathematical Programming*, 95:3–51, 2001. 10

[7] G.W. Medlin, and J.W. Adams, and C.T. Leondes. Lagrange multiplier approach to the design of FIR filters for multirate applications. *IEEE Transactions on Circuits and Systems*, 35(10):1210–1219, 1988. 1

[8] I. Schur. On power series which are bounded in the interior of the unit circle, Part I. In I. Gohberg, editor, *I. Schur Methods in Operator Theory and Signal Processing*, volume 18 of *Operator Theory: Advances and Applications*, pages 31–59. Springer Basel AG, 1986. DOI 10.1007/978-3-0348-5483-2. 1, 2

[9] I. W. Selesnick and C. S. Burrus. Exchange Algorithms for the Design of Linear Phase FIR Filters and Differentiators Having Flat Monotonic Passbands and Equiripple Stopbands. *IEEE Transactions on Circuits and Systems-II:Analog and Digital Signal Processing*, 43(9):671–675, September 1996. 1, 10

[10] I. W. Selesnick, M. Lang and C. S. Burrus. Constrained Least Square Design of FIR Filters without Specified Transition Bands. *IEEE Transactions on Signal Processing*, 44(8):1879–1892, August 1996. 1

[11] I. W. Selesnick, M. Lang and C. S. Burrus. A Modified Algorithm for Constrained Least Square Design of Multiband FIR Filters without Specified Transition Bands. *IEEE Transactions on Signal Processing*, 46(2):497–501, February 1998. 1, 10, 11

[12] J. H. McClellan, T. W. Parks and L. R. Rabiner. A Computer Program for Designing Optimum FIR Linear Phase Digital Filters. *IEEE Transactions on Audio and Electroacoustics*, 21(6):506–526, December 1973. 1

[13] John D. Markel and A. H. Gray Jr. On Autocorrelation Equations as Applied to Speech Analysis. *IEEE Transactions on Audio and Electroacoustics*, AU-21(2):69–79, April 1973. 1, 2

[14] John W. Eaton and the Octave project developers. GNU Octave manual : a high-level interactive language for numerical computations. https://docs.octave.org/latest, 2025. 1

[15] Keshab K. Parhi. *VLSI Digital Signal Processing Systems : Design and Implementation*. Wiley Inter-Science, 1999. ISBN 0-471-24186-5. 2, 3, 6, 12

[16] L. Theile. On the Sensitivity of Linear State-Space Systems. *IEEE Transactions on Circuits and Systems*, 33(5):502–510, May 1986. 7

[17] M. A. Richards. Application of Deczky's Program for Recursive Filter Design to the Design of Recursive Decimators. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 30(5):811–814, October 1982. 1, 6, 9

[18] M. C. Lang. Least-Squares Design of IIR Filters with Prescribed Magnitude and Phase Responses and a Pole Radius Constraint. *IEEE Transactions on Signal Processing*, 48(11):3109–3121, November 2000. 1

[19] Octave. Non-linear optimization package. https://gnu-octave.github.io/packages/optim. 1

[20] Octave. Signal processing package. https://gnu-octave.github.io/packages/signal. 1

[21] P. A. Regalia, S. K. Mitra and P. P. Vaidyanathan. The Digital All-Pass Filter: A Versatile Signal Processing Building Block. *Proceedings of the IEEE*, 76(1):19–37, January 1988. 1

[22] P. P. Vaidyanathan and Sanjit K. Mitra. Robust Digital Filter Structures: A Direct Approach. *IEEE Circuits and Systems Magazine*, pages 14–32, January-March 2019. DOI 10.1109/MCAS.2018.2889204. 1

[23] P. P. Vaidyanathan, S. K. Mitra and Y. Nuevo. A New Approach to the Realization of Low-Sensitivity IIR Digital Filters. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 34(2):350–361, April 1986. 1

[24] R. A. Roberts and C. T. Mullis. *Digital Signal Processing*. Addison Wesley, 1987. ISBN 0-201-16350-0. 9

[25] R. Sedgwick. *Algorithms in C++*. Addison-Wesley, 1990. ISBN 0-201-51059-6. 13

[26] Rika Ito, Tetsuya Fujie, Kenji Suyama and Ryuichi Hirabayashi. A powers-of-two term allocation algorithm for designing FIR filters with CSD coefficients in a min-max sense. http://www.eurasip.org/Proceedings/Eusipco/Eusipco2004/defevent/papers/cr1722.pdf. 13

[27] J. Sturm. SeDuMi_1_3 at GitHub. https://github.com/sqlp/sedumi. 10

[28] T. Iwasaki and S. Hara. Generalised KYP Lemma: Unified Frequency Domain Inequalities With Design Applications. *IEEE Transactions on Automatic Control*, 50(1):41–59, January 2005. 1

[29] T. Kailath. A theorem of I. Schur and its impact on modern signal processing. In I. Gohberg, editor, *I. Schur Methods in Operator Theory and Signal Processing*, volume 18 of *Operator Theory: Advances and Applications*, pages 9–30. Springer Basel AG, 1986. DOI 10.1007/978-3-0348-5483-2. 1, 2

[30] T. N. Davidson, Z.-Q. Luo and J. F. Sturm. Linear Matrix Inequality Formulation of Spectral Mask Constraints With Applications to FIR Filter Design. *IEEE Transactions on Signal Processing*, 50(11):2702–2715, November 2002. 1

[31] T. W. Parks and J. H. McClellan. Chebyshev Approximation for Nonrecursive Digital Filters with Linear Phase. *IEEE Transactions on Circuit Theory*, 19(2):189–194, March 1972. 1, 10

[32] W.-S. Lu and T. Hinamoto. Optimal Design of IIR Digital Filters With Robust Stability Using Conic Quadratic-Programming Updates. *IEEE Transactions on Signal Processing*, 51(6):1581–1592, June 2003. 1

[33] Wu-Sheng Lu. Use SeDuMi to Solve LP, SDP and SCOP Problems: Remarks and Examples. http://www.ece.uvic.ca/~wslu/Talk/SeDuMi-Remarks.pdf. 10

[34] Y. C. Lim, R. Yang, D. Li and J. Song. Signed Power-of-Two Term Allocation Scheme for the Design of Digital Filters. *IEEE Transactions on Circuits and Systems-II:Analog and Digital Signal Processing*, 46(5):577–584, May 1999. 12