Discrete Optimization of

Canonic Wave Digital Filters

---

by

Paul Soble

A thesis
presented to the University of Manitoba
in partial fulfillment of the
requirements for the degree of
Master of Science
in
Electrical Engineering

Winnipeg, Manitoba

DISCRETE OPTIMIZATION OF CANONIC WAVE DIGITAL FILTERS

BY

PAUL JOHN SOBLE

A thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

ABSTRACT

This thesis investigates the design of canonic wave digital filters (WDFs) based on elliptic reference filters, and the minimization of their realization requirements.

Two synthesis approaches are considered: cascade synthesis, in which the WD structure is composed of a chain of adaptors, and WD lattice synthesis in which the structure is based on an analog symmetrical lattice.

The use of the Brune adaptor in cascade synthesis is desireable as it allows the canonic (i.e. minimal) realization of WDFs based on elliptic ladder reference filters. A derivation of this adaptor and several examples of its application are presented.

Lattice WDFs, which are based on analog symmetrical lattice prototypes, are also desireable as they, too, allow canonic realization of odd-order elliptic reference filters.

WDFs have the property of low parameter sensitivity, which can be exploited to often achieve very short multiplier wordlengths. The possibility of replacing actual multipliers by binary shifts and additions allows further reduction of a filter's complexity.

An algorithm is proposed here which attempts to minimize the total number of shifts and additions required to realize a given design. A number of design examples are presented which illustrate the success of this approach for fifth- and seventh-order filters realized using cascades of Brune and other adaptors. The fifth-order examples, since they are sufficiently small, have been verified using a direct (exhaustive) search approach. Also, some examples illustrating the applicability of the scheme to WD lattice filters are presented, and which compare favourably to previously published results.

ACKNOWLEDGEMENT

The author wishes to express his gratitude to Professor G.O. Martens for his guidance, patience and support during the course of this work. He also wishes to thank Mr. Mark Jarmasz for his relentless encouragement and assistance, not to mention many helpful discussions. Thanks are also extended to Mlles. Marie Patenaude and Clare Soble, and Mssrs. Ted Soble and Ken Ehn. Financial assistance from the Natural Sciences and Engineering Research Council under Grant A3813 is appreciated.

TABLE OF CONTENTS

1. INTRODUCTION

Filtering is a process by which an input signal is reshaped to yield an output signal having different characteristics, specified in the time, or more commonly, the frequency domain. Most filters are frequency selective in that some frequencies are attenuated while others are passed or amplified.

Filtering may be performed on continuous signals or on signals which exist only at discrete instants of time. A discrete-time filter may then be viewed as a computational algorithm operating on an input sequence of numbers to produce an output sequence.

The theory of discrete-time systems and filters is well-developed, but is based on the assumption that signal representations and arithmetic operations are carried out to infinite precision. In practice, only finite precision is available since discrete-time systems usually are implemented using digital processors. The signals in digital systems, then, are discrete in time and also discrete in amplitude.

Digital filters are often classified into one of two broad categories, those which are recursive in nature and those which are not. Recursive filters are capable of high stopband attenuation and require lower computational complexity than for non-recursive structures.

A digital filter is essentially an approximation to a discrete-time prototype upon which finite precision constraints have been imposed. Deviations in behavior of the digital filter from its unrestricted prototype arise, and are due to what are generally termed finite wordlength effects (FWLEs) [1-4]. Finite wordlength effects are comprised of the following categories:

1) coefficient quantization error, which occurs due to the quantization of the filter coefficients to a finite precision, and

2) signal quantization error, which is the error introduced by quantization of input, output, and intermediate signal quantities to finite precision.

Coefficient quantization error, or roundoff noise, is a linear deterministic error which has the effect of deviating the frequency response from the desired response. Signal quantization error is a random additive error produced due to the discard of portions of the signal too small (underflow) or too large (overflow) to be represented by the given precision [5]. In the case of recursive filters, underflow or overflow errors may be correlated such that oscillations, known as limit cycles or parasitic oscillations, are sustained, even under zero-input conditions. Digital filters must be designed to control and minimize these undesirable effects.

Finite wordlength effects can be reduced, for a given discrete time realization, by simply increasing the precision used, at greater costs of implementation. An alternative is to choose structures inherently less susceptable to FWLEs [6-8]. Structures which exploit the well-known [9,10] relationship between roundoff noise and coefficient sensitivities have been derived [11-16]. Also, low-order sections able to suppress all types of limit cycles have been developed [17-22]. Further, filter realizations which are designed to suppress the highly destructive overflow oscillations have been investigated [24-27]. Of course, those which, in addition, are low in realization requirements are preferable.

An alternative structure which behaves favorably under FWL conditions is the wave digital filter (WDF) proposed by Fettweis [28] and developed by him and others [29-37]. They are high order recursive structures capable of high stopband attenuations. Wave digital filters are based on the premise that analog reference filters possessing the properties of good sensitivity and passivity can be transformed to an equivalent digital structure such that the desirable qualities are preserved. This transformation is achieved using a voltage wave network description and the bilinear z-transform. It has the effect of replacing analog reactive elements by simple delays, and simulates analog interconnections by means of wave adaptors.

Advantages of WDFs are very low coefficient sensitivity and corresponding low roundoff noise. In addition, Fettweis and Meerkötter have shown via the concept of stored pseudopower that all zero-input limit-cycles may be suppressed in canonic WDFs [39]. A disadvantage of WDFs is the requirement of a larger number of additions than for conventional realizations such as parallel or cascade connections of direct form low-order sections. Also, WD filters derived from reference filters non-minimal in reactance elements will be non-minimal in delays. Subsequent removal of these redundancies invalidates the simple stability criterion, requiring more complex means to achieve limit-cycle suppression [29,40,68,69].

More recently, contributions of the lattice adaptor [41], and the Brune adaptor [42-44] allow canonic realization of symmetric lattice and ladder topologies, respectively. Also, the low sensitivity of WDFs can be exploited to often drastically simplify multiplier requirements and hence reduce overall computational complexity [43,45-49].

The problem of minimizing digital filter hardware requirements has been addressed largely by means of optimization techniques [50-63], concentrating on cascades of low-order sections. Wegener and Owenier [45-49] have given optimized WDF designs of symmetric lattice and ladder prototypes, although the ladder realizations have been non-canonic.

The thesis presented herein is concerned with the reduction of the realization requirements of canonic, limit-cycle-free WDFs in which explicit multipliers have been replaced by binary shifts and additions. The reduction of realization requirements is formulated as an optimization problem in which the total number of shifts and additions is to be minimized. Canonic, limit-cycle free implementations of ladder networks of arbitrary order are obtained through the use of the Brune adaptor.

Chapter 2 presents background to the WD approach and covers the introduction of voltage wave variables and the bilinear z-transformation. Next, fundamental analog network elements and interconnections are related to their WD counterparts. A state-variable description of a WDF is developed by partitioning an analog network via reactance extraction, and then transforming the subnetworks into their WD equivalents. The reflection-free property is introduced to allow the interconnection of adaptors. Finally, sufficient conditions are given for ensuring stability of a WD network despite the nonlinear nature of FWL conditions.

In chapter 3, techniques for the synthesis of canonic, stable WDFs are given. In particular, the design of WDFs using the adaptors of Fettweis and the Brune adaptor of Jarmasz [43] are presented. A discussion of the WD lattice or Jaumann structure [41] is included since it is exceptionally low in realization requirements and so has gained popularity. The chapter concludes with the representation of multipliers in the canonical signed digital code (CSDC) and its consequences to some methods of physical implementation.

Chapter 4 formulates the problem of reducing realization requirements of cascade WDFs as an optimization problem. Two types of adaptors are covered: those for which simple fixed flowgraphs exist, such as the Fettweis adaptors, and adaptors, such as the general Brune, for which no simple flowgraph exists. (A simple flowgraph is one in which each multiplier appears only once.) An optimization algorithm suited to reduction of the realization requirements of WDFs based on both kinds of adaptors is presented. Several examples are given to demonstrate the capabilities of this approach.

2. INTRODUCTION TO WAVE DIGITAL FILTERS

Wave digital filters (WDFs) comprise a class of digital structures which imitate classical reactance filters so as to exploit their desirable properties. In particular, classical reactance networks are lossless and, when terminated by resistances and resistive sources, are relatively insensitive to element variations. These characteristics have the consequence of low passband sensitivity to coefficient variations, good dynamic range, and the possibility of the suppression of parasitic oscillations in the corresponding WDF.

2.1. The Wave Digital Transformation

A WDF is derived from a classical reactance network, called its reference network, by replacing the conventional signal quantities of voltage v and current i by voltage wave variables defined by

$$a(t) = v(t)+R\ i(t),\ \ b(t) = v(t)-R\ i(t) \tag{2.1}$$

$$A(\psi) = V(\psi)+R\ I(\psi),\ \ B(\psi) = V(\psi)-R\ I(\psi) \tag{2.2}$$

where, as depicted in Fig. 2.1, a and b are the incident and reflected waves, respectively, and R is an arbitrary port reference resistance. The digital equivalent of the reference network is derived by applying the transformation

$$\psi = \frac{z-1}{z+1} = \tanh\frac{sT}{2},\ \ z \triangleq e^{sT} \tag{2.3}$$

where s is the Laplace transform variable and T is the digital sampling period. We see that the filter voltage transfer characteristic described by

$$V(\psi) = H(\psi)\,E(\psi), \tag{2.4}$$

is transformed by letting $R_1{=}R_S$ and $R_2{=}R_L$, yielding

$$A_1(\psi) = E_1(\psi),\ \ B_1(\psi) = 2V_1 - E_1 \tag{2.5a}$$

$$A_2(\psi) = 0,\ \ B_2(\psi) = 2V_2, \tag{2.5b}$$

from which the voltage wave transfer function is given by

$$\frac{B_2(\psi)}{A_1(\psi)} = \frac{2V_2}{E_1}. \tag{2.6}$$

The voltage wave transfer function is therefore identical to the voltage transfer function except for the constant 2.

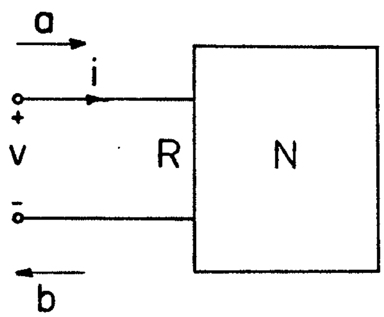


Figure 2.1 Definition of wave variables at one port.

The application of the transformations (2.1), (2.2) and (2.3) to some elementary analog one-ports and the WD equivalents which result is illustrated in Fig.2.2.

In order for two ports to be interconnected they must be compatible, that is

$$v_1=v_2, \;\; i_1=-i_2 \tag{2.7}$$

and

$$a_1=b_2, \;\; a_2=b_1, \;\; R_1=R_2, \tag{2.8}$$

which ensure that Kirchhoff's current and voltage laws are obeyed at the interconnection. To fulfill the last requirement the port voltage waves must be adapted to properly simulate the connection.

2.2. Voltage Wave Scattering Description

Consider a doubly terminated lossless reactance network N (Fig.2.3) consisting of two subnetworks M and $\tilde{M}$. Network $\tilde{M}$ contains the reactive elements of N, and M contains only interconnections and possibly ideal transformers. Define port voltage and current vectors describing the ports of M and partitioned with respect to ports containing resistive sources (possibly of zero value), inductances, and capacitances as follows:

$$\mathbf{v} = \begin{bmatrix}\mathbf{v}_1\\ \mathbf{v}_2\end{bmatrix}, \quad \mathbf{i} = \begin{bmatrix}\mathbf{i}_1\\ \mathbf{i}_2\end{bmatrix} \tag{2.9}$$

where

$$\mathbf{v}_1 = \begin{bmatrix}\mathbf{v}_S\\ \mathbf{v}_L\end{bmatrix}, \quad \mathbf{i}_1 = \begin{bmatrix}\mathbf{i}_S\\ \mathbf{i}_L\end{bmatrix} \tag{2.10}$$

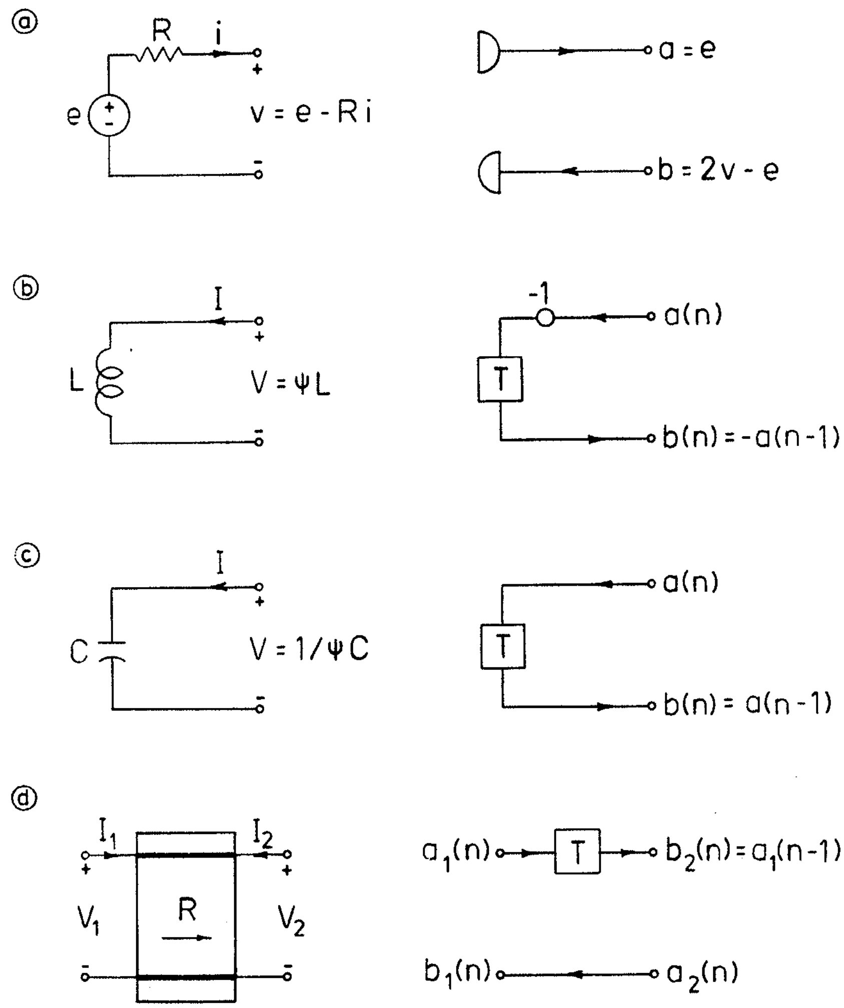


Figure 2.2 Some analog circuit elements and their wave-digital equivalents.

$$\mathbf{v}_2 = \begin{bmatrix} \mathbf{v}_L \\ \mathbf{v}_C \end{bmatrix}, \qquad \mathbf{i}_2 = \begin{bmatrix} \mathbf{i}_L \\ \mathbf{i}_C \end{bmatrix}. \tag{2.11}$$

We may now define the port wave vectors of M to be

$$\mathbf{a} = \mathbf{v} + \mathbf{R}\mathbf{i}, \qquad \mathbf{b} = \mathbf{v} - \mathbf{R}\mathbf{i} \tag{2.12}$$

where $\mathbf{R}$ is a real diagonal matrix of arbitrary port reference resistances, and $\mathbf{a}$ and $\mathbf{b}$ are partitioned conformable to $\mathbf{v}$ and $\mathbf{i}$. The voltage wave (scattering) variable description of M is then

$$\mathbf{b} = \mathbf{S}\,\mathbf{a} \tag{2.13}$$

where **S** is a real constant matrix describing the interconnections within M. Similarly, we may define voltage, current and voltage wave vectors describing $\tilde{M}$,

$$\tilde{\mathbf{v}} = \begin{bmatrix} \tilde{\mathbf{v}}_L \\ \tilde{\mathbf{v}}_C \end{bmatrix}, \qquad \tilde{\mathbf{i}} = \begin{bmatrix} \tilde{\mathbf{i}}_L \\ \tilde{\mathbf{i}}_C \end{bmatrix} \tag{2.14}$$

$$\tilde{\mathbf{a}} = \tilde{\mathbf{v}} + \tilde{\mathbf{R}}\tilde{\mathbf{i}}, \qquad \tilde{\mathbf{b}} = \tilde{\mathbf{v}} - \tilde{\mathbf{R}}\tilde{\mathbf{i}} \tag{2.15}$$

and

$$\tilde{\mathbf{b}} = \tilde{\mathbf{S}}\,\tilde{\mathbf{a}} \tag{2.16}$$

where $\tilde{S}$ is given by

$$\tilde{\mathbf{S}} = \frac{1-\psi}{1+\psi}\,(-\mathbf{U}_{n_L} \dotplus \mathbf{U}_{n_C}) \tag{2.17}$$

$$= \frac{1-\psi}{1+\psi}\Sigma,$$

We define Σ as

$$\Sigma = -\mathbf{U}_{n_L} \dotplus \mathbf{U}_{n_C}, \tag{2.18}$$

and $\dotplus$ denotes direct sum. At the interconnection Kirchhoff's voltage and current laws must be satisfied, implying

$$\mathbf{v}_2 = \tilde{\mathbf{v}}, \qquad \mathbf{i}_2 = -\tilde{\mathbf{i}} \tag{2.19}$$

or, in terms of scattering variables,

$$\mathbf{a}_2 = \tilde{\mathbf{b}}, \qquad \mathbf{b}_2 = \tilde{\mathbf{a}}, \qquad \mathbf{R}_2 = \tilde{\mathbf{R}}. \tag{2.20}$$

A convenient choice for **R** is

$$\mathbf{R} = diag(\,R_S\,, R_L\,, L_1, L_2, \,.\,.\,.\,,\; L_{n_L}, 1/C_1, 1/C_2, \,.\,.\,.\,,\; 1/C_{n_C}\,) \tag{2.21}$$

where the partitioning is conformable to **v** and **i**.

Application of the bilinear transformation (2.3) to M and $\tilde{M}$ yields the equations

$$\mathbf{B}(z) = \mathbf{S}\,\mathbf{A}(z), \qquad \tilde{\mathbf{B}}(z) = \frac{1}{z}\,\Sigma\,\tilde{\mathbf{A}}(z) \tag{2.22}$$

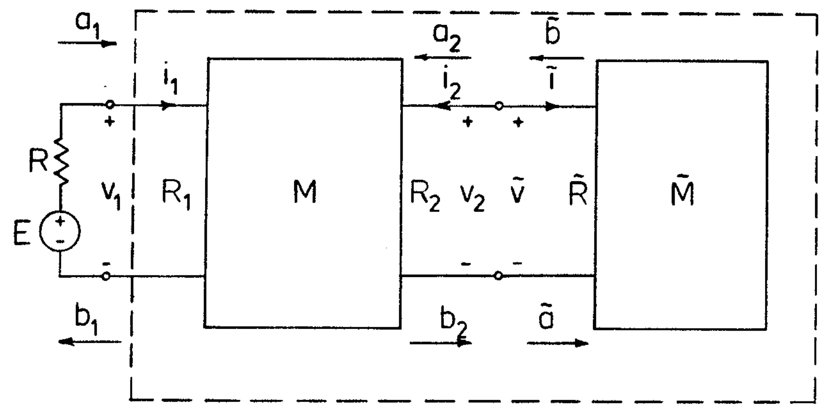


Figure 2.3 Doubly-terminated network showing reactance extraction partitioning

or equivalently

$$\mathbf{b}(n) = \mathbf{S}\,\mathbf{a}(n), \qquad \tilde{\mathbf{b}}(n) = \Sigma\,\tilde{\mathbf{a}}(n-1). \tag{2.23}$$

The above equations describe the computation of filter output and delay signal quantities. A natural and convenient extension of this is the state variable description of the filter.

2.3. State-Variable Description

A digital filter may be described in terms of the state variable matrices { **A**, **B**, **C**, **D** } :

$$\mathbf{x}(n+1) = \mathbf{A}\,\mathbf{x}(n) + \mathbf{B}\,\mathbf{u}(n) \tag{2.24}$$

$$\mathbf{y}(n) = \mathbf{C}\,\mathbf{x}(n) + \mathbf{D}\,\mathbf{u}(n)$$

where $\mathbf{x}(n), \mathbf{u}(n), \mathbf{y}(n)$ are the state, the input and the output vectors at the n^{th} sample instant. The WDF described above can be written as

$$\Sigma\,\mathbf{b}_2(n) = \mathbf{a}_2(n+1) = \Sigma\,\mathbf{S}_{22}\,\mathbf{a}_2(n) + \Sigma\,\mathbf{S}_{21}\,\mathbf{a}_1(n) \tag{2.25}$$

$$\mathbf{b}_1(n) = \mathbf{S}_{12}\,\mathbf{a}_2(n) + \mathbf{S}_{11}\,\mathbf{a}_1(n)$$

so that the state variable description of a WDF can be given by

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \Sigma\,\mathbf{S}_{22} & \Sigma\,\mathbf{S}_{21} \\ \mathbf{S}_{12} & \mathbf{S}_{11} \end{bmatrix}. \tag{2.26}$$

This system specifies the computation required to realize **S**. It remains that **S** be calculated from the reference filter. For this, the work of Martens and Meerkötter [65,66] provides a means to find $\mathbf{S}$ directly from an arbitrary analog network.

2.4. The N-Port Description

Consider an n-port reference network consisting of interconnections and ideal transformers only. The port voltage and current vectors may be partitioned into link ports l and twig ports t,

$$\mathbf{v} = \begin{bmatrix} \mathbf{v}_l \\ \mathbf{v}_t \end{bmatrix}, \qquad \mathbf{i} = \begin{bmatrix} \mathbf{i}_l \\ \mathbf{i}_t \end{bmatrix} \tag{2.27}$$

and similarly for the port wave vectors,

$$\mathbf{a} = \begin{bmatrix} \mathbf{a}_l \\ \mathbf{a}_t \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} \mathbf{b}_l \\ \mathbf{b}_t \end{bmatrix} \tag{2.28}$$

The link and twig quantities are related by

$$\mathbf{v}_l = \mathbf{N}^T \mathbf{v}_t, \qquad \mathbf{i}_t = -\mathbf{N}\,\mathbf{i}_l \tag{2.29}$$

where the turns ratio matrix $\mathbf{N}$ is real. Define a constant matrix K given by

$$\mathbf{K} = (\mathbf{G}_t + \mathbf{N}\mathbf{G}_l\mathbf{N}^T)^{-1}\,\mathbf{N}\mathbf{G}_l \tag{2.30}$$

where $\mathbf{G}_t$ and $\mathbf{G}_l$ are the diagonal branch conductance matrices for twig and link ports, respectively. Martens and Meerkötter have shown that the scattering matrix of a constant lossless network can be expressed as

$$\mathbf{S} = \begin{bmatrix} 2\mathbf{N}^T\mathbf{K} - \mathbf{U} & 2\mathbf{N}^T(\mathbf{U} - \mathbf{K}\mathbf{N}^T) \\ 2\mathbf{K} & \mathbf{U} - 2\mathbf{K}\mathbf{N}^T \end{bmatrix} \tag{2.31}$$

or

$$\mathbf{S} = \mathbf{F}\mathbf{T}\mathbf{F} \tag{2.32}$$

where

$$\mathbf{F} = \begin{bmatrix} -\mathbf{U} & \mathbf{N}^T \\ 0 & \mathbf{U} \end{bmatrix}, \qquad \mathbf{T} = \begin{bmatrix} -\mathbf{U} & 0 \\ -2\mathbf{K} & \mathbf{U} \end{bmatrix}. \tag{2.33}$$

Matrices $\mathbf{F}$ and $\mathbf{T}$ are self-inverse, and so $\mathbf{S}$ is also self-inverse, i.e.

$$\mathbf{S}^2 = \mathbf{U}. \tag{2.34}$$

The matrix **N** can usually be found directly from the oriented graph derived from the network; the matrix **K** can be found through a convenient network interpretation :

i) terminate all tree ports in their reference resistances,

ii) terminate all link ports with a voltage source e in series with the port reference resistance.

The relationship between twig voltages $\mathbf{v}_t$ and the excitations $\mathbf{e}_l$ is given by

$$\mathbf{v}_t = \mathbf{K}\,\mathbf{e}_l. \tag{2.35}$$

This voltage transfer matrix can be obtained analytically from the network, usually by application of Thevenin's theorem and superposition only.

Note the number of degrees of freedom of **K** is $t \cdot l$, which may not necessarily be the canonic number for the transfer function. This is the case for elliptic ladder filters. A representation for **K** in terms of a canonic number of multipliers has been shown to exist for several topologies, found by a suitable redefinition of the parameters used in obtaining **K** from the network.

The network of interconnections M may be (non-uniquely) decomposed into a number of smaller subnetworks. Fettweis and his colleagues have chosen to use adaptors with at most three ports, modelling series and parallel electrical interconnections as series and parallel adaptors, respectively, as depicted in Fig. 2.4. This is often convenient because each of the adaptors shown has a fixed flowgraph containing design parameters as multipliers. Adaptors for Brune, symmetrical lattice, Darlington C and D sections, and twin-T networks have also been derived [41-44,67].

2.5. The Reflection-Free Property

A potential realizability problem arises with the interconnection of adaptors in that a delay-free loop, an unrealizable network [38], may be created, as shown in Fig.2.5. This problem can be avoided by constraining the reflected wave at a port to be instantaneously independent of the incident wave at the same port, that is by making the port reflection-free [37]. Thus the scattering matrix **S** of a sub-network having port i reflection-free will have $s_{ii}=0$. Two examples of adaptors having a reflection-free port are given in Fig.2.6.

A reflection-free port can be interpreted to have its reference resistance equal to the port driving-point resistance when all other ports are terminated by their reference

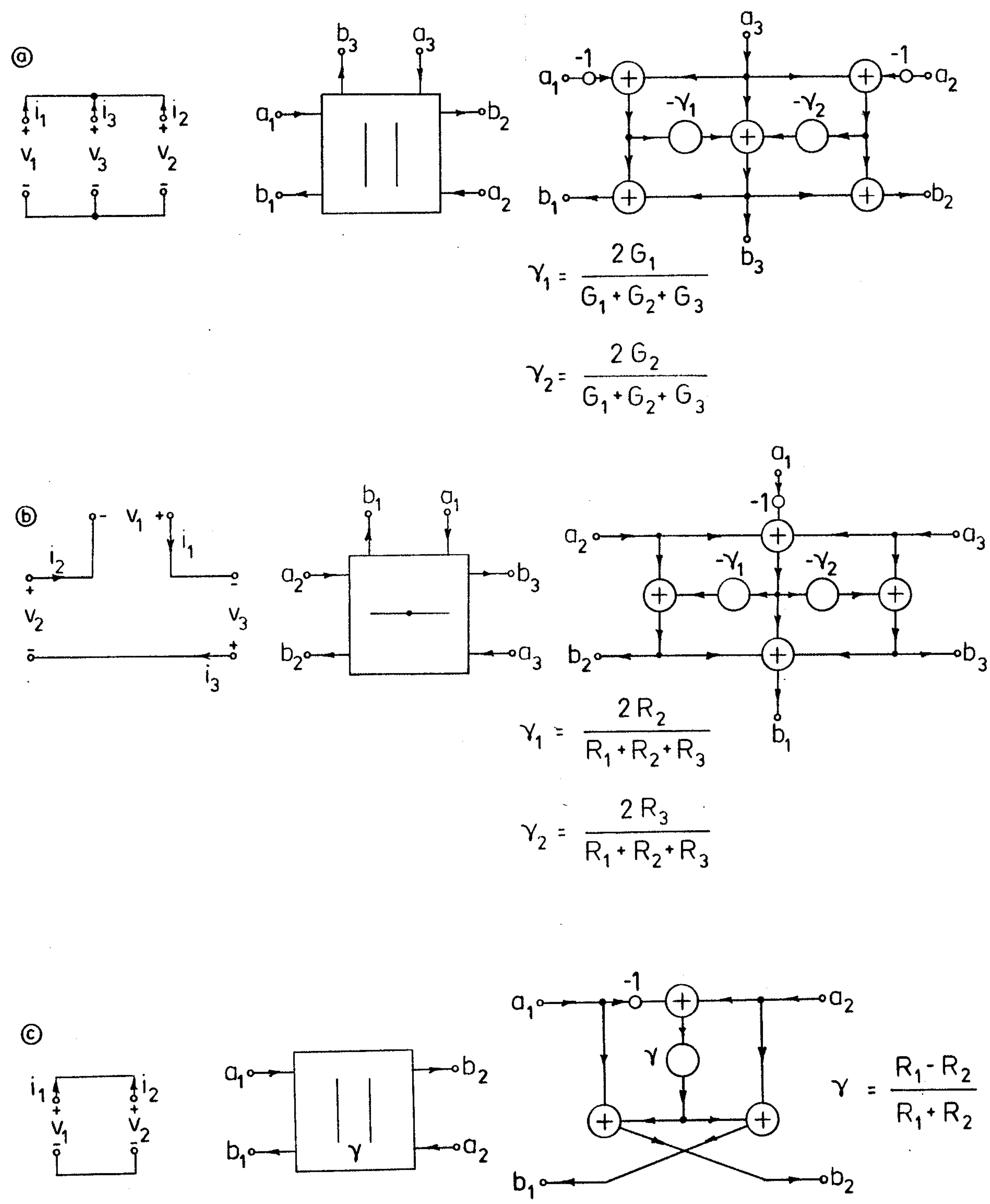


Figure 2.4 Some basic wave-digital adaptors: a) 3-port parallel adaptor, b) 3-port series adaptor, c) 2-port parallel adaptor.

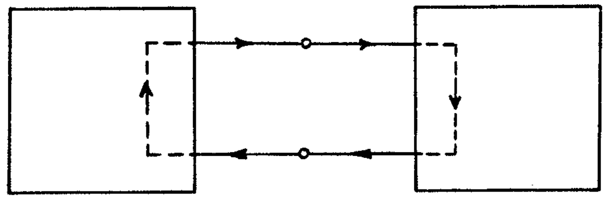


Figure 2.5 A delay-free loop condition.

resistances. The introduction of this constraint reduces the number of degrees of freedom of the adaptor by one, preserving the canonic number of degrees.

An example of a network, a third order elliptic ladder filter, realized by the interconnection of series and parallel adaptors is given in Fig.2.7.

2.6. Non-Linear Stability

Wave digital filters can be designed to have the important property of complete stability under normal operating conditions in which arithmetic operations are performed with finite precision [39,43,68]. Stability under ideal (infinite precision) conditions is directly achieved since the bilinear transform maps a stable analog reference filter onto a stable discrete-time one. However, with finite wordlength arithmetic the possibility of overflow and granularity oscillations also arises. We now specify, following Fettweis and Meerkötter [39], conditions which lead to complete stability and which may easily be taken into account in the arithmetic operations of a practical filter implementation.

The incident and reflected waves for a lossless, frequency-independent reciprocal network N are related by

$$\mathbf{b}(n) = \mathbf{S}\,\mathbf{a}(n). \tag{2.36}$$

Let $\mathbf{G}$ be the positive-definite diagonal port reference conductance matrix. Then the instantaneous pseudopower absorbed by N at the n^{th} time instant is given by

$$\begin{aligned} p_N(n) &= \mathbf{a}^T(n)\,\mathbf{G}\,\mathbf{a}(n) - \mathbf{b}^T(n)\,\mathbf{G}\,\mathbf{b}(n) \\ &= \mathbf{a}^T(n)\,\mathbf{G}\,\mathbf{a}(n) - \mathbf{a}^T(n)\,\mathbf{S}^T\mathbf{G}\,\mathbf{S}\,\mathbf{a}(n). \end{aligned} \tag{2.37}$$

Network N is pseudopassive if $p_N(n) \geq 0$, and pseudolossless if $p_N(n)=0$ for all $\mathbf{a}(n)$. For N pseudolossless we have

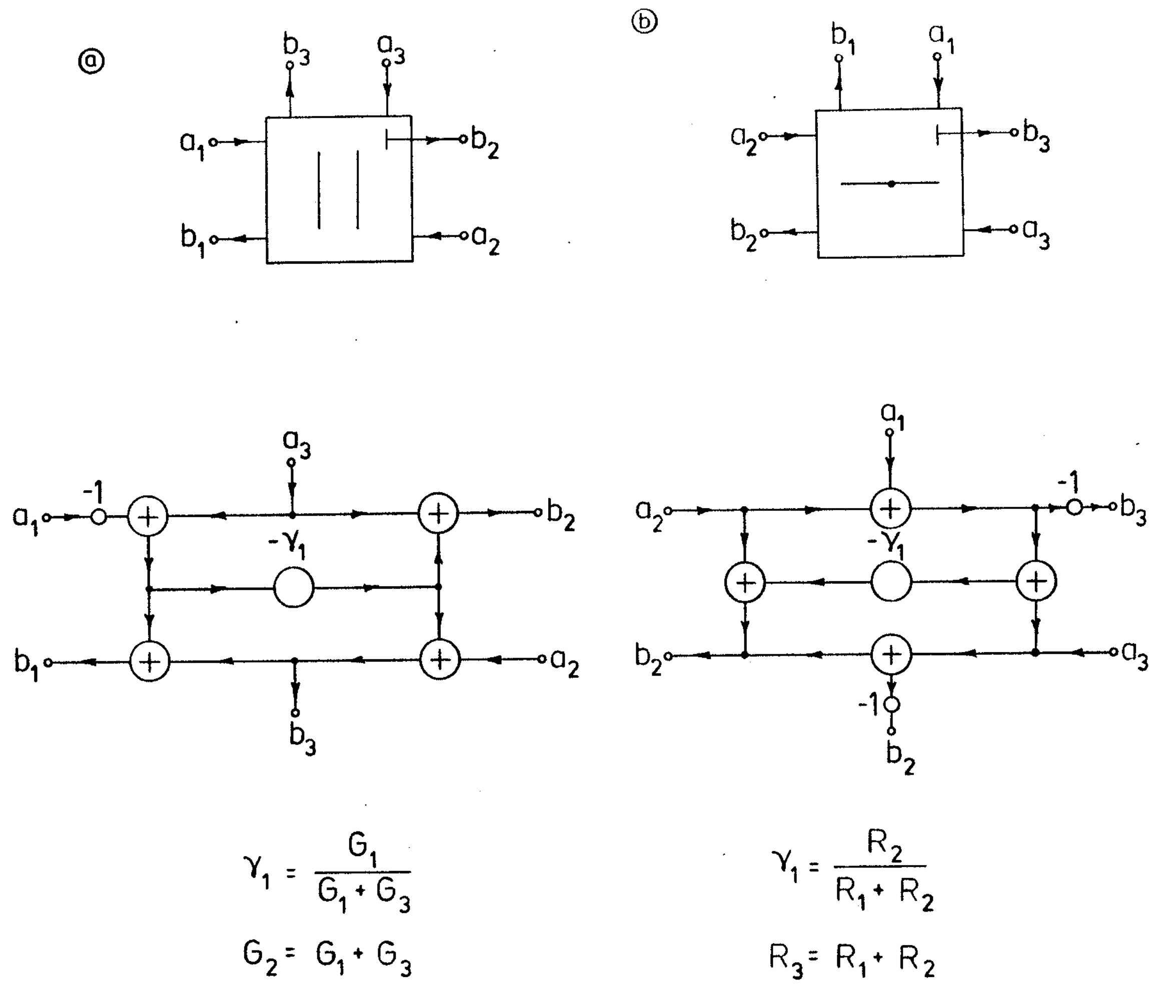


Figure 2.6 A parallel and a series 3-port adaptor, each having a reflection-free port.

$$\mathbf{a}^T(n)\,(\mathbf{G} - \mathbf{S}^T\mathbf{G}\,\mathbf{S})\,\mathbf{a}(n)=0 \tag{2.38}$$

for all $a(n)$, which implies

$$\mathbf{G} = \mathbf{S}^T\mathbf{G}\,\mathbf{S}. \tag{2.39}$$

Since $\mathbf{S}^2 = \mathbf{U}$, we can obtain from (2.39)

$$\mathbf{GS} = \mathbf{S}^T\mathbf{G} \tag{2.40}$$

which states that M is reciprocal with respect to the reference conductance matrix $\mathbf{G}$.

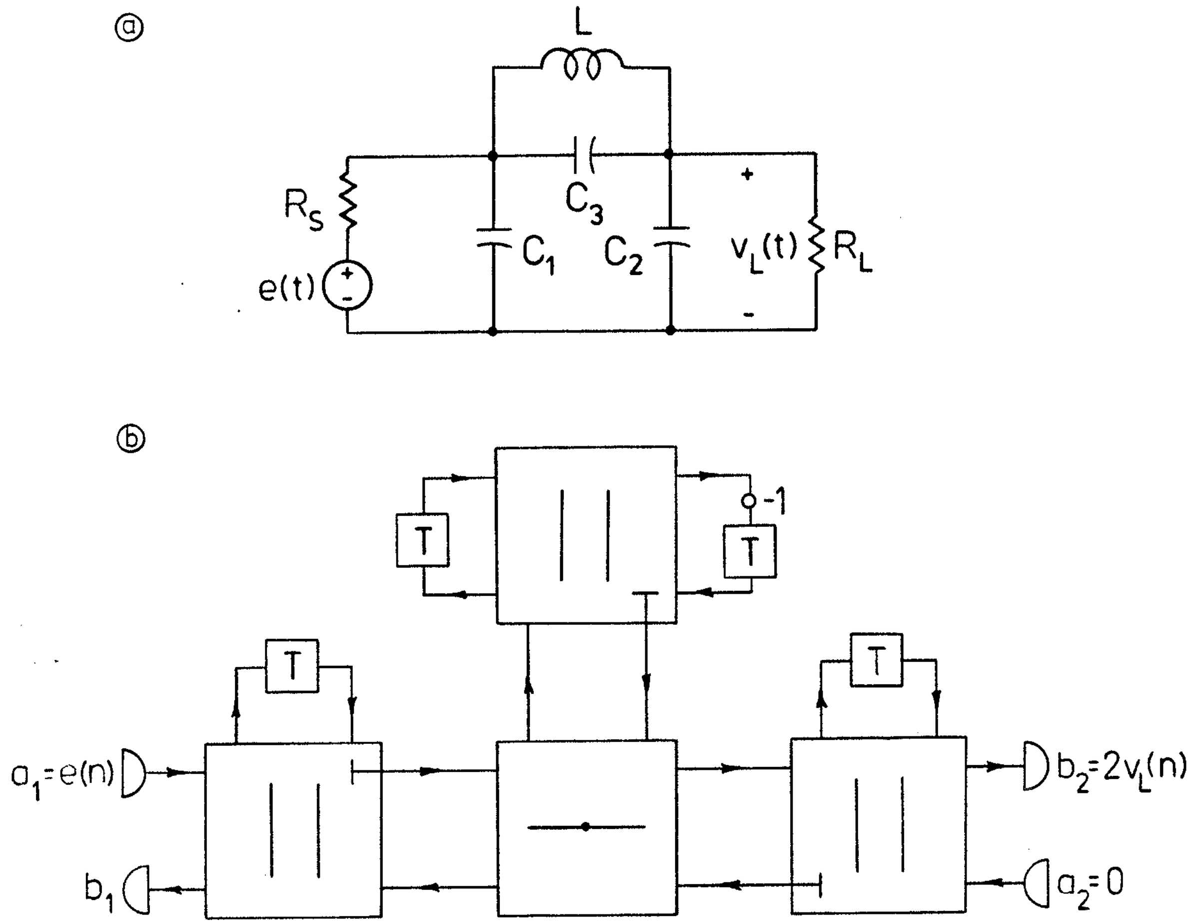


Figure 2.7 a) A third-order elliptic ladder filter, and b) its realization using series and parallel adaptors.

Consider now the pseudopower absorbed by N at the n^{th} time instant, given by

$$p(n) = \mathbf{a}_2^T(n)\, \mathbf{G}_{22}\, \mathbf{a}_2(n). \tag{2.41}$$

The decrease in absorbed pseudopower is then

$$\begin{aligned}\Delta p(n) &= p(n) - p(n+1) \qquad (2.42)\\ &= \mathbf{a}_2^T(n)\, \mathbf{G}_{22}\, \mathbf{a}_2(n) - \mathbf{a}_2^T(n+1)\, \mathbf{G}_{22}\, \mathbf{a}_2(n+1)\end{aligned}$$

Using (2.20) and (2.23),

$$\Delta p(n) = \mathbf{a}_2^T(n)\, \mathbf{G}_{22}\, \mathbf{a}_2(n) - (\Sigma \mathbf{b}_2(n))^T\, \mathbf{G}_{22}\, \Sigma \mathbf{b}_2(n) \tag{2.43}$$

$$= \mathbf{a}_2^T \mathbf{G}_{22} \mathbf{a}_2(n) - \mathbf{a}_2^T(n) \mathbf{S}_{22} \mathbf{G}_{22} \mathbf{S}_{22} \mathbf{a}_2(n)$$

$$= \mathbf{a}_2^T(n) (\mathbf{G}_{22} - \mathbf{S}_{22} \mathbf{G}_{22} \mathbf{S}_{22}) \mathbf{a}_2(n).$$

From (2.39) it follows that

$$\mathbf{G}_{22} = \mathbf{S}_{12}^T \mathbf{G}_{11} \mathbf{S}_{12} + \mathbf{S}_{22}^T \mathbf{G}_{22} \mathbf{S}_{22} \tag{2.44}$$

we obtain

$$\Delta p(n) = \mathbf{a}_2^T(n) \mathbf{S}_{12}^T \mathbf{G}_{11} \mathbf{S}_{12} \mathbf{a}_2(n) \tag{2.45}$$

$$= (\mathbf{S}_{12} \mathbf{a}_2(n))^T \mathbf{G}_{11} \mathbf{S}_{12} \mathbf{a}_2(n)$$

$$= \mathbf{b}_1^T(n) \mathbf{G}_{11} \mathbf{b}_1(n).$$

Since $\mathbf{G}_{11}$ is positive definite,

$$\Delta p(n) \geq 0 \tag{2.46}$$

which is sufficient for output stability in the sense of Lyapunov, if the $p(n)$ is the Lyapunov function.

The above condition holds when arithmetic computations are performed exactly. We also wish it to hold under finite precision operation, which has the effect of introducing quantizers into the linear system. Define a system $\tilde{N}$ which is equivalent to N except that each of the output signals $\tilde{b}_i(n)$ is given by

$$\tilde{b}_i(n) = Q(b_i(n)) \tag{2.47}$$

where Q is a non-linear quantization function. Define the decrease in stored pseudopower $\Delta\tilde{p}(n)$ in a manner similar to $\Delta p(n)$. A sufficient condition for output stability of N is then

$$\Delta\tilde{p}(n) \geq \Delta p(n) \tag{2.48}$$

which is implied by

$$\tilde{\mathbf{b}}_2^T \mathbf{G}_{22} \tilde{\mathbf{b}}_2 \leq \mathbf{b}_2^T \mathbf{G}_{22} \mathbf{b}_2. \tag{2.49}$$

Since $\mathbf{G}$ is positive definite and diagonal, this is implied by

$$|\tilde{b}_{2i}| \leq |b_{2i}| \quad for\ every\ i. \tag{2.50}$$

A scheme which satisfies this condition for the underflow case is sign- magnitude truncation;

saturation arithmetic may be used for the overflow case. Fettweis and Meerkötter have extended the stability argument to the case when N is composed of an interconnection of adaptors, provided that (2.50) is true at all ports of each adaptor.

Minimality, both in delays and in degrees of freedom, plays an important role in the preceding development. Note that the condition

$$\Delta p(n) = \mathbf{a}_2^T(n)\, \mathbf{G}_{22}\, \mathbf{a}_2(n) - \mathbf{a}_2^T(n+1)\, \mathbf{G}_{22}\, \mathbf{a}_2(n+1) \tag{2.51}$$

$$= \mathbf{b}_1^T(n)\, \mathbf{G}_{11}\, \mathbf{b}_1(n)$$

$$= 0$$

can be true when $\mathbf{b}_1(n) = 0$ for a non-zero state vector $\mathbf{a}_2(n)$ only if the system is unobservable. Therefore only output stability is guaranteed by (2.51). To guarantee complete stability, the system must, in addition to (2.51), be observable. Ashley has shown [40] that reciprocal, observable WD networks are also controllable and therefore minimal in delays. Thus minimal WD networks are completely stable in the sense of Lyapunov.

WDFs based on non-canonic analog networks are themselves non-minimal in delays. Methods for removing the redundant delays have been developed [29], but these change the WD network by introducing off-diagonal entries in the conductance matrix $\mathbf{G}$, and as a consequence (2.50) may no longer be sufficient. Recently, methods for diagonalizing the reference conductance matrix have been developed to overcome this problem by means of an exact diagonalization transformation [40,72].

Of course, this problem could be circumvented by choosing a canonic reference network such as the Jaumann symmetric lattice structure. Alternatively, the non-minimal network may be transformed into an equivalent network canonic in reactances, but yet containing a surplus parameter. It has been shown that the extra parameter may be expressed in terms of a suitable redefinition of the canonic design parameters such that the extra parameter will be finite wordlength binary (FWLB) if the canonic parameters are FWLB. The development of this idea will be covered in the next chapter.

3. SYNTHESIS OF WAVE DIGITAL FILTERS

The design of canonic stable WDFs is essentially the mapping of a suitable analog network onto an adaptor or interconnection of adaptors whose ports are terminated in delays. Ultimately the adaptors, which define a computational algorithm, are to be realized physically. In this chapter the derivation of discrete-time wave filters from analog reference filters, and some possible digital implementation will be discussed.

The synthesis of a stable WD filter generally involves the following steps:

i) specification of the frequency domain magnitude response, most often specified as a maximum allowable error in the passband and minimum attenuation in the stopband.

ii) choice of a suitable $H(\psi)$, which in most cases can be satisfied by an equiripple transfer function and found with the aid of design tables or a computer program. For specifications which are not equiripple, either perturbation, continuous optimization, or other techniques could be used.

iii) realization of $H(\psi)$ as a doubly-terminated lossless reactive network, which may contain inductances, capacitances, unit elements, and ideal transformers. The ladder realization of $H(\psi)$ is widely available from tables. This step is not actually necessary, but an analog realization of $H(\psi)$ must exist.

iv) possible removal of redundant reactances via a suitable network transformation.

v) transformation of the analog network into an equivalent WD network by replacing the reactive elements with delays, possibly in series with an inverter, and by replacing the interconnections by a WD multiport or an interconnection of adaptors.

vi) approximate the canonic design parameters by binary fractions such that the design specifications are still met.

vii) scale the digital filter to minimize the probability of overflow and to maximize dynamic range.

viii) implement the filter as an algorithm on general- or special purpose hardware, ensuring that (2.50) holds, such as by sign-magnitude truncation at the states.

Two methods for the design of WDFs will be considered:

i) transformation of a ladder network to a WD network canonic in delays and consisting of a cascade of first- and second-order sections. The removal of the reactive redundancies will be achieved by simple analog network transformations (which, incidentally, are equivalent to the diagonalization of Ashley [40]).

ii) realizing a transfer function as the equivalent of a symmetric lattice, and using first, second, or higher-order cascades of unit elements to realize the two lattice reactances. This realization is inherently canonic in delays and multipliers, but has the disadvantages of high stopband sensitivity and that it is restricted to symmetric (odd-order) filter networks.

Consider the ladder realization, shown in Fig.2.7, of a third-order elliptic transfer function as the reference filter for a WDF. A redundant reactance, a capacitor, exists within the loop of capacitors C_1, C_2, and C_3 (the dual network would contain a redundant inductor in one of its cutsets). A network canonic in reactances can be obtained via a network transformation [73], given in Fig.3.1, the result of which would be equivalent to deriving the WDF from the non-minimal network, removing the redundancy, and rediagonalizing the port reference conductance matrix **G**. Application of this transformation to the capacitive loop in the filter of Fig.2.7 yields the network shown in Fig.3.2. A corresponding WDF (a cascade realization) can then be derived through application of the Brune adaptor of Martens and Jarmasz and a parallel adaptor, as depicted in Fig. 3.3, where the Brune adaptor was arbitrarily chosen to have the reflection-free port. The design of the Brune adaptor follows.

3.1. Design of the Brune Adaptor

Consider the Brune section shown in Fig.3.4a). To proceed with an N-port adaptor representation of the Brune section, $\mathbf{v}_t = \mathbf{K}\mathbf{e}_l$ and $\mathbf{v}_l = \mathbf{N}^T \mathbf{v}_t$ must be found, which requires that the network first be partitioned into link and twig ports. Although there are six possible partitionings, the one which yields the simplest entries for **N** is used. By inspection of Fig.3.4b), the loop equations are

$$v_3 = v_1 + v_2 \tag{3.1}$$

$$v_4 = v_1 + n\, v_2$$

or

$$\mathbf{v}_l = \mathbf{N}^T \mathbf{v}_t , \tag{3.2}$$

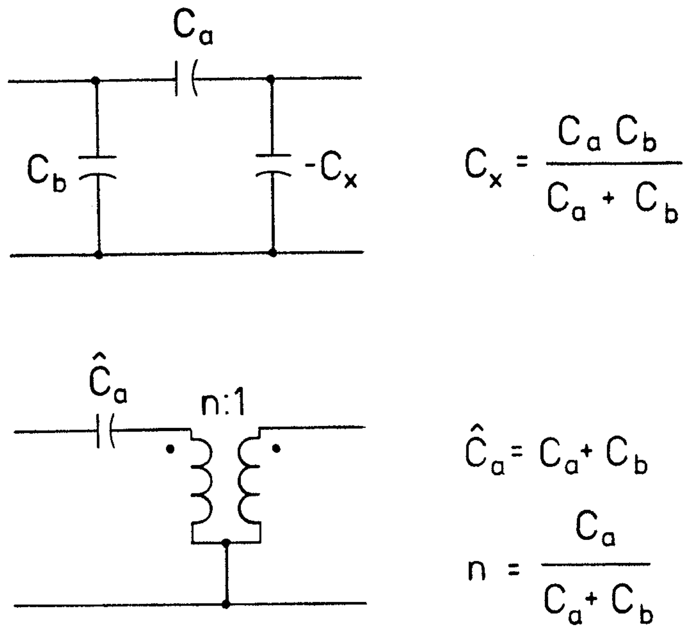


Figure 3.1 Two equivalent networks.

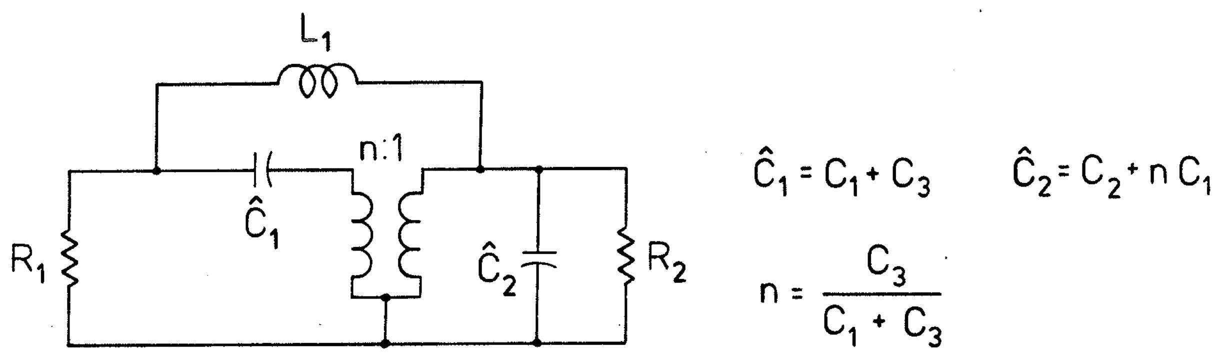


Figure 3.2 Equivalent third order elliptic filter.

where

$$\mathbf{N}^T = \begin{bmatrix} 1 & 1 \\ 1 & n \end{bmatrix}, \qquad \mathbf{v}_l = \begin{bmatrix} v_3 \\ v_4 \end{bmatrix}, \qquad \mathbf{v}_t = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}. \tag{3.3}$$

Terminate twig ports in their reference conductances and link ports in their reference conductances in series with a voltage source, as shown in Fig.3.5. We may now analyze the network to obtain the voltage transfer relationship between link-port sources and twig voltages, and hence obtain **K**. To simplify the analysis process, the ideal transformer can be eliminated via

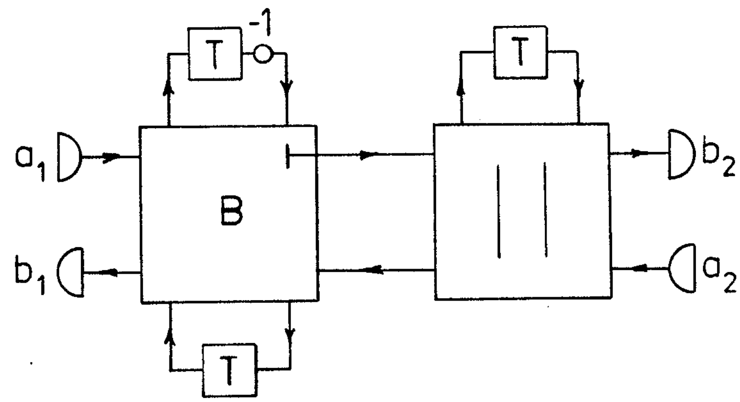


Figure 3.3 Third-order elliptic WDF realized using a Brune and a parallel adaptor.

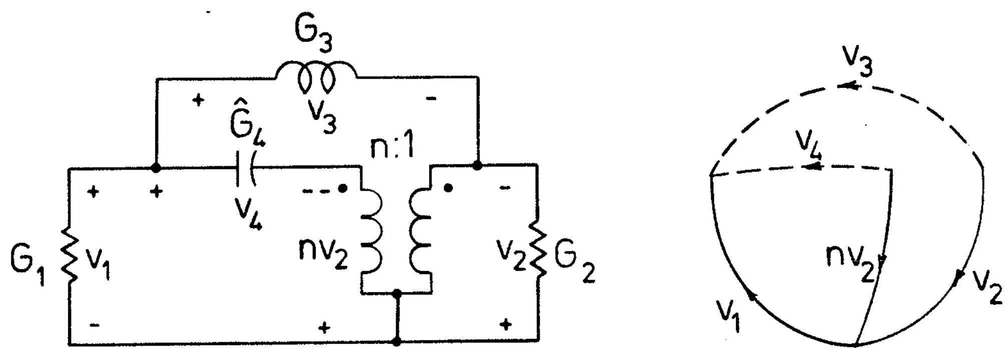


Figure 3.4 a) A Brune section showing branch voltages, and b) an oriented graph for the network.

the network transformation of Fig.3.1 to yield the network of Fig.3.6a). Application of Thévenin's theorem together with the definition of some naturally-occurring constants yields the network of Fig.3.6b), where

$$\alpha_1 = \frac{G_5}{G_1 + G_5}, \quad \alpha_2 = \frac{G_3}{G_3 + G_4}. \tag{3.4}$$

Continuing this process, we find the twig voltages v_1 and v_2,

$$v_1 = \frac{G_b}{G_1 + G_5 + G_b}\left[\alpha_2 e_3 + (1-\alpha_1-\alpha_2)\, e_4\right] + \alpha_1 e_4 \tag{3.5}$$

$$= \alpha_2 \alpha_3 e_3 + \left[\alpha_3(1-\alpha_2) + \alpha_1(1-\alpha_3)\right] e_4$$

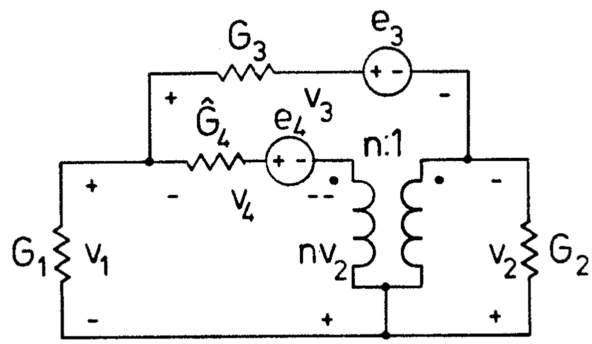


Figure 3.5 Brune network for calculation of the K matrix.

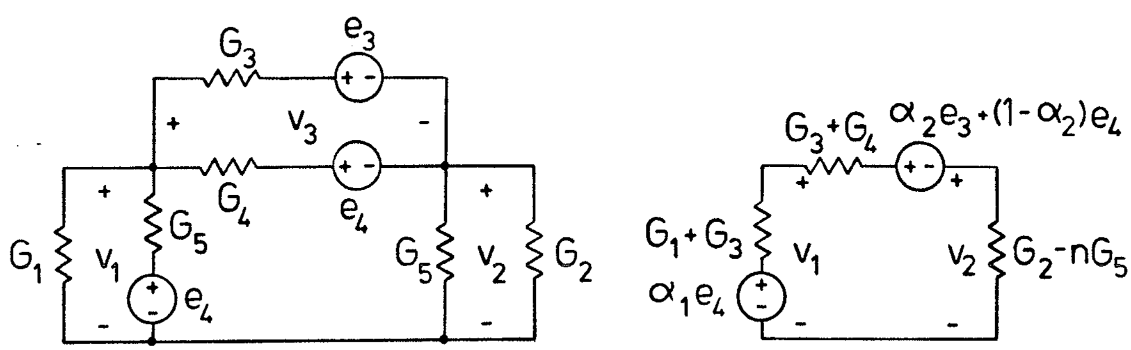


Figure 3.6 a) Network after application of equivalence of Fig.3.1., and b) after further simplification.

$$v_2 = \frac{G_a}{G_a + G_2 - nG_5}\left[\alpha_2\, e_3 + (1-\alpha_1-\alpha_2)\, e_4\right] \tag{3.6}$$

$$= \alpha_2\alpha_4 e_3 + \alpha_4(1-\alpha_1-\alpha_2)\, e_4$$

in which

$$\alpha_3 = \frac{G_b}{G_b + G_1 + G_5} = \frac{1}{1 + (G_1 + G_5)(\frac{1}{G_3 + G_4} + \frac{1}{G_2 - nG_5})} \tag{3.7}$$

$$\alpha_4 = \frac{G_a}{G_a + G_2 - n\, G_5} = \frac{1}{1 + (G_2 - n\, G_5)(\frac{1}{G_3 + G_4} + \frac{1}{G_1 + G_5})} \tag{3.8}$$

and where

$$G_a = \frac{1}{\dfrac{1}{G_3 + G_4} + \dfrac{1}{G_1 + G_5}}, \quad G_b = \frac{1}{\dfrac{1}{G_3 + G_4} + \dfrac{1}{G_2 - n\, G_5}}. \tag{3.9}$$

Now the relation $\mathbf{v}_t = \mathbf{K}\mathbf{e}_l$ can be written as

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \alpha_2\,\alpha_3 & \alpha_3(1-\alpha_2) + \alpha_1(1-\alpha_3) \\ \alpha_2\,\alpha_4 & \alpha_4(1-\alpha_1 - \alpha_2) \end{bmatrix} \begin{bmatrix} e_3 \\ e_4 \end{bmatrix}. \tag{3.10}$$

The overall scattering matrix for the adaptor is given by

$$S = \left[\begin{array}{llll} 1 - 2[\alpha_3(1-\alpha_1) + \alpha_1] & -2[\alpha_2\alpha_3(1-n) + n\,\alpha_3(1-\alpha_1) + n\,\alpha_1] & 2\alpha_2\alpha_3 & 2[\alpha_3(1-\alpha_2) + \alpha_1(1-\alpha_3)] \\ -2\alpha_4(1-\alpha_1) & 1 - 2[\alpha_2\alpha_4(1-n) + n\,\alpha_4(1-\alpha_1)] & 2\alpha_2\alpha_4 & 2\alpha_4(1-\alpha_1-\alpha_2) \\ 2 - 2[\alpha_1 + (\alpha_3+\alpha_4)(1-\alpha_1)] & 2 - 2[(\alpha_3 + \alpha_4)(n\,(1-\alpha_1-\alpha_2)+\alpha_2) + n\,\alpha_1] & 2\alpha_2(\alpha_3+n\,\alpha_4) & 2[(\alpha_3+\alpha_4)(1-\alpha_1-\alpha_2) + \alpha_1] \\ 2 - 2[(\alpha_3+n\,\alpha_4)(1-\alpha_1) + \alpha_1] & 2n - 2[(\alpha_2(1-n) + n\,(1-\alpha_1))(\alpha_3+n\,\alpha_4) + n\,\alpha_1] & 2\alpha_2(\alpha_3+n\,\alpha_4) & 2[(\alpha_3+n\,\alpha_4)(1-\alpha_1-\alpha_2) + \alpha_1]-1 \end{array}\right] \tag{3.11}$$

The matrix $\mathbf{K}$ has now been expressed in terms of the design parameters $\{\alpha\}$ and n. Only four degrees of freedom exist in the original network (three independent conductance ratios and the parameter n), and so the new parameter set is non-minimal. In solving equations (3.4, 3.7, 3.8) for the conductance ratios we find

$$\frac{G_1}{G_2} = \frac{(1-\alpha_1)\alpha_4}{\alpha_3 + n\ \alpha_1\,\alpha_4}, \quad \frac{G_3}{G_2} = \frac{\alpha_2}{1-\alpha_2}\ \frac{n}{1-n}\ \frac{\alpha_1\,\alpha_4}{\alpha_3 + n\ \alpha_1\,\alpha_4} \tag{3.12}$$

$$\frac{G_4}{G_2} = \frac{n}{1-n}\ \frac{\alpha_1\,\alpha_4}{\alpha_3 + n\ \alpha_1\,\alpha_4}, \quad \frac{G_5}{G_2} = \frac{\alpha_1\,\alpha_4}{\alpha_3 + n\ \alpha_1\,\alpha_4} \tag{3.13}$$

and that the following dependence relation holds:

$$\alpha_1 = \frac{\alpha_3\,(1-n)(1-\alpha_2)}{n\ (1- \alpha_3 - \alpha_4\,)}. \tag{3.14}$$

Through a suitable redefinition of parameters, using $\{\alpha\}$ together with (3.14), a minimal parameter set can be obtained. A definition will be suitable if expressing the new set as binary fractions ensures that the old set will also be expressible as binary fractions. If the relationship between the new and old sets can be expressed in sum-of-products (SOP) form, this is sufficient. Following the approach of Jarmasz [43], three suitable redefinitions are found to exist, one of which is given here :

$$
\begin{aligned}
\beta_1 &= n \\
\beta_2 &= \alpha_3 \qquad\qquad & \alpha_1 &= (1-\beta_1)\,\beta_2\,\beta_4 \\
\beta_3 &= \alpha_4 \qquad\qquad & \alpha_2 &= 1-\beta_1\,(1-\beta_2-\beta_3)\beta_4 \\
\beta_4 &= \frac{1-\alpha_2}{n\,(1-\alpha_3-\alpha_4)}
\end{aligned}
\tag{3.15}
$$

The choice of which definition to use is not generally clear, but for a given parameter set, each will have different implications for realization requirements. A summary of all Brune adaptor design and analysis equations is given in Appendix A.

3.1.1. Port 1 Reflection-Free Brune Section

Either port one or two could be reflection-free to allow the interconnection of adaptors. An additional dependence relationship is obtained for each case by setting the appropriate diagonal term in the adaptors scattering matrix to zero. For the case of port one reflection-free we have

$$2(1-\alpha_1)(1-\alpha_3) - 1 = 0. \tag{3.16}$$

Solving for α_1 and substituting into the dependence relation (3.14), we can solve for α_4,

$$\alpha_4 = (1-\alpha_3)\left[1 - \frac{2\alpha_3(1-n)(1-\alpha_2)}{n(1-2\alpha_3)}\right]. \tag{3.17}$$

Rational entries in **K** are created by the substitution of α_1 and α_4. Those terms in **K** which pose a problem are:

$$\alpha_2\,\alpha_4 = \alpha_2(1-\alpha_3)\left[1 - \frac{2\alpha_3(1-n)(1-\alpha_2)}{n(1-2\alpha_3)}\right] \tag{3.18}$$

and

$$\alpha_1\,\alpha_4 = \frac{1-2\alpha_3}{2(1-\alpha_3)}(1-\alpha_3)\left[1 - \frac{2\alpha_3(1-n)(1-\alpha_2)}{n(1-2\alpha_3)}\right]. \tag{3.19}$$

One suitable redefinition of parameters sufficient to eliminate these rational expressions from

K is

$$\beta_1 = n \qquad \alpha_1 = \frac{1-2\beta_2}{2(1-\beta_2)}$$
$$\beta_2 = \alpha_3 \qquad \alpha_2 = 1 - \beta_1\beta_3\,(1-2\beta_2) \tag{3.20}$$
$$\beta_3 = \frac{1-\alpha_2}{n\,(1-2\alpha_3)} \qquad \alpha_4 = (1-2\beta_2\,(1-\beta_1)\,\beta_3)(1-\beta_2)$$

3.1.2. Port 2 Reflection-Free Brune Adaptor

If port 2 is to be reflection-free, the following condition must hold:

$$\alpha_2\alpha_4(1-n) + n\,\alpha_4(1-\alpha_1) = \frac{1}{2}. \tag{3.21}$$

To eliminate α_1 and α_2 from the **K**-matrix, solve the above equation for α_1, and the dependence relation (3.14) for α_2, yielding

$$\alpha_1 = \frac{\alpha_3(2\alpha_4-1)}{2n\,\alpha_4(1-\alpha_4)}, \tag{3.22}$$

$$1 - \alpha_2 = \frac{(2\alpha_4-1)(1-\alpha_3-\alpha_4)}{2(1-\alpha_4)(1-n)\alpha_4} \tag{3.23}$$

and substituting these into the **K**-matrix, we find the following rational terms in **K**:

$$\alpha_2\alpha_3 = \alpha_3\left[1 - \frac{(2\alpha_4-1)(1-\alpha_3-\alpha_4)}{2(1-\alpha_4)(1-n)\alpha_4}\right] \tag{3.24}$$

$$\alpha_2\alpha_4 = \alpha_4 - \frac{(2\alpha_4-1)(1-\alpha_3-\alpha_4)}{2(1-\alpha_4)(1-n)} \tag{3.25}$$

$$\alpha_1\alpha_4 = \frac{\alpha_3(2\alpha_4-1)}{2n\,\alpha_4(1-\alpha_4)} \tag{3.26}$$

$$\alpha_1(1-\alpha_3) = \frac{\alpha_3(1-\alpha_3)(2\alpha_4-1)}{2n\,\alpha_4(1-\alpha_4)} \tag{3.27}$$

In this case it is necessary to define two new parameters to convert to SOP form the entries of **K**. One of the two choices for a parameter set is

$$\beta_1 = n \qquad \alpha_1 = \frac{(1 - 2\alpha_4\beta_2(1-\beta_1))\beta_3}{\alpha_4}$$
$$\beta_2 = \frac{1-\alpha_3-\alpha_4}{2(1-n)\alpha_4(1-\alpha_4)} \qquad \alpha_2 = 1 - \beta_1\beta_2\beta_3 \tag{3.28}$$
$$\alpha_3 = (1-\alpha_4)(1-2\alpha_4\beta_2(1-\beta_1))$$
$$\beta_3 = \frac{2\alpha_4-1}{2n} \qquad \alpha_4 = \beta_1\beta_3 + \frac{1}{2}$$

The above equations ensure that the entries of **K**, and hence **S**, will contain only SOP functions of the canonic parameters $\{\beta\}$, and as a consequence will allow a Brune adaptor to be implemented as a binary-arithmetic, digital algorithm.

3.1.3. Simplified Brune Section

A simplified design for a Brune section without reflection-free ports can be obtained by imposing the constraint

$$\alpha_1 = 1 - \alpha_2 \tag{3.29}$$

or equivalently

$$\frac{G_1}{G_5} = \frac{G_3}{G_4}, \tag{3.30}$$

which, using (3.14) also implies

$$\alpha_3 = n(1-\alpha_4) \tag{3.31}$$

The resulting **K**-matrix is

$$\mathbf{K} = \begin{bmatrix} (1-\alpha_1)n(1-\alpha_4) & \alpha_1 \\ (1-\alpha_1)\alpha_4 & 0 \end{bmatrix} \tag{3.32}$$

which represents a reduction in design complexity compared to the unconstrained design of § 3.1. An appropriate parameter set is simply

$$\beta_1 = n, \qquad \beta_2 = 2(1-\alpha_1), \qquad \beta_3 = 2\alpha_1\alpha_5. \tag{3.33}$$

The resulting adaptor has a corresponding flowgraph, which is shown in Fig.3.7. For the port 2 reflection-free case, imposition of (3.21) yields the **K**-matrix

$$\mathbf{K} = \begin{bmatrix} n(\frac{1}{2}-\alpha_1) & \alpha_1 \\ \frac{1}{2} & 0 \end{bmatrix}. \tag{3.34}$$

Although the constraint (3.29) reduces the number of degrees of freedom by one, experience has shown that many designs can be satisfactorily realized regardless. Similar adaptors are possible through different choices for the simplifying constraint.

Various designs for the Brune section have been presented above, and are summarized in

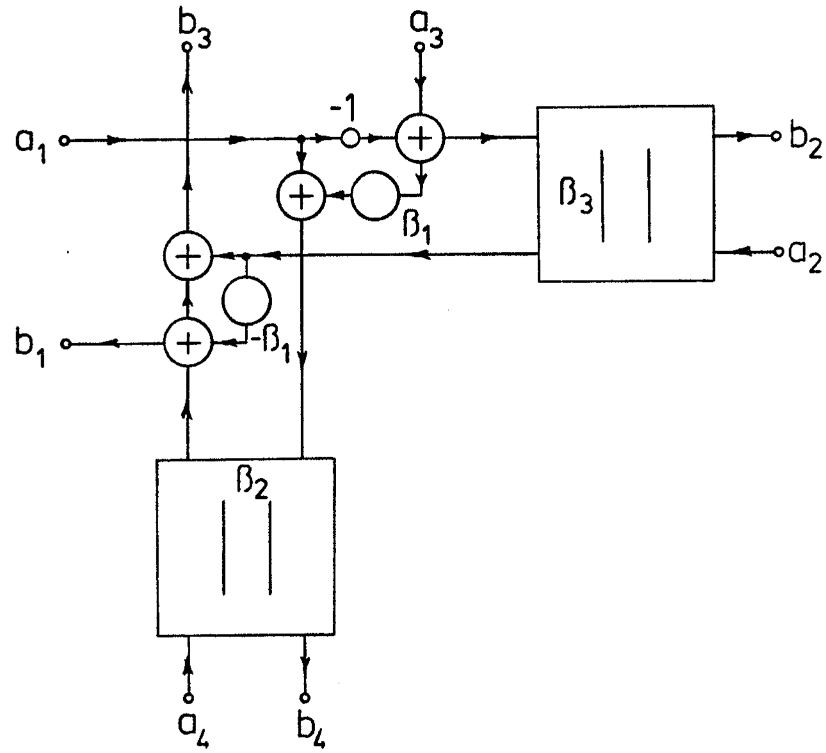


Figure 3.7 Flowgraph of non-reflection-free simplified Brune adaptor. If port 2 is to be reflection-free its two-port adaptor is simply replaced by a straight connection.

Appendix A. Next we will consider the WD lattice configuration.

3.2. Lattice Wave Digital Filters

A second, recently popularized approach to WDF design is through the use of a classical doubly-terminated symmetrical lattice as the reference structure. The symmetrical lattice and an equivalent (Jaumann) structure is shown in Fig.3.8. Its voltage wave transfer function may be expressed as

$$S_{21} = \frac{2V_2}{E} = \frac{B_2}{A_1} = \frac{1}{2}(S_2 - S_1) \tag{3.35}$$

where the reflectances S_1 and S_2 are given by

$$S_1 = \frac{Z_1 - R}{Z_1 + R}, \quad S_2 = \frac{Z_2 - R}{Z_2 + R}. \tag{3.36}$$

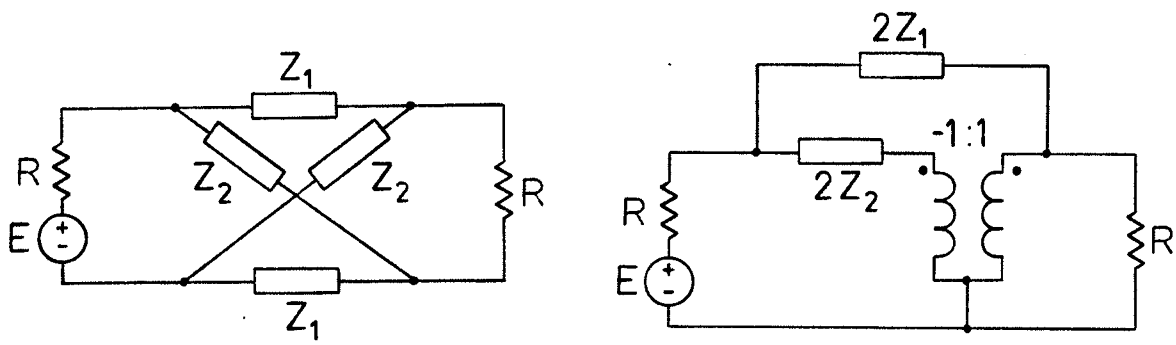


Figure 3.8 a) A symmetrical lattice and b) the equivalent Jaumann structure.

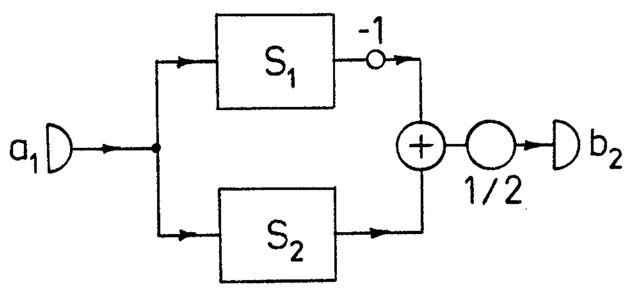


Figure 3.9 Wave flow diagram of the unidirectional WD lattice.

Fettweis has given a WD structure which can realize (3.35), which is shown in Fig.3.9.

We choose to describe the analog network in terms of the canonic form of the scattering matrix for a lossless reciprocal two-port [73-75], given by

$$\mathbf{S} = \frac{1}{g}\begin{bmatrix} h & \sigma f_* \\ f & -\sigma h_* \end{bmatrix} \tag{3.36a}$$

where f , g , and h are the canonic polynomials. Some properties of the canonic scattering matrix are:

i) f , g , and h are real polynomials in the complex frequency variable ψ,

ii) g is Hurwitz,

iii) either g or f is monic,

iv) $\sigma = \pm 1$,

v) $hh_* + ff_* = gg_*$ holds.

The * indicates the Hurwitz conjugate which in the case of real polynomials corresponds to replacing ψ by $-\psi$.

For the symmetric case, $\sigma = f_*/f = -h_*/h$, which implies

$$gg_* = \sigma\,(f - h)(f + h). \tag{3.37}$$

The polynomial g can be expressed as a unique product such that the following hold [45] :

$$g = g_1 g_2, \qquad g_1 g_{2*} = f + h, \qquad g_{1*} g_2 = \sigma(f - h), \tag{3.38}$$

from which we may derive the reflectances

$$S_1 = -\frac{\sigma g_{1*}}{g_1}, \qquad S_2 = \frac{g_{2*}}{g_2} \tag{3.39}$$

and the canonic impedances

$$Z_1 = R\left(\frac{g_{1o}}{g_{1e}}\right)^{\sigma}, \quad Z_2 = R\left(\frac{g_{2e}}{g_{2o}}\right). \tag{3.40}$$

(In the case of elliptic and similar transfer functions it has been shown that the zeros of g are distributed alternately on the $j\omega$-axis between g_1 and g_2 [76].) The impedances Z_1 and Z_2, or alternately S_1 and S_2, may then be realized by any number of classical synthesis techniques. We shall choose cascades of first- and second-order all pass sections or n-th order cascades of unit elements [77,78].

3.2.1. Cascade of First- and Second-Order All-Pass Sections.

To realize a reflectance S as a cascade of first and second order all pass sections it is necessary only to express S in the following factored form:

$$S = \frac{kg_*}{g} = \frac{-\psi + a_1}{\psi + a_1} \prod_{i=2}^{\frac{1}{2}(n-1)} \frac{\psi^2 - a_i\psi + b_i}{\psi^2 + a_i\psi + b_i}, \tag{3.41}$$

where $k = -\sigma$ in the case of S_1 and $k = 1$ for the synthesis of S_2. Each second-order section can be realized simply by application of Richard's reactance extraction [79]. The resulting analog network and corresponding WD flow diagram for n odd is shown in Fig.3.10, where

$$
\begin{aligned}
R_1 &= Ra_1 & \gamma_1 &= \frac{1-a_1}{1+a_1} \\
R_{2i} &= R\frac{b_i}{a_i} & \gamma_{2i} &= \frac{a_i-b_i-1}{a_i+b_i+1} \\
R_{2i+1} &= \frac{R}{a_i}, \quad i=1(1)\frac{n-1}{2} & \gamma_{2i+1} &= \frac{1-b_i}{1+b_i}, \quad i=1(1)\frac{n-1}{2}.
\end{aligned}
\tag{3.42}
$$

If n is even, the first order section is simply omitted.

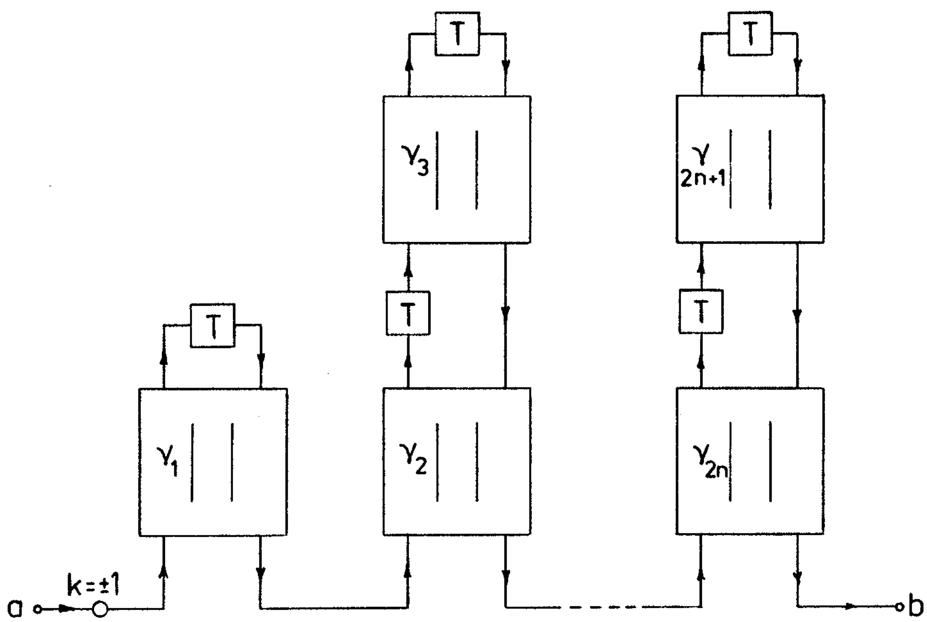


Figure 3.10 Realization of an impedance via a cascade of first- and second-order all-pass sections.

3.2.2. Chain of Unit Elements

To realize a reflectance $S = (Z-R)/(Z+R)$ as a chain of unit elements Richard's reactance extraction may be applied to Z according to the following recursive relation

$$Z_{i+1} = R_i \frac{Z_i - \psi R_i}{R_i - \psi Z_i}, \quad Z_1 = Z, \quad i=1(1)n, \tag{3.43}$$

which will terminate in either a short or open circuit, and where

$$R_i = Z_i(1) = -Z_i(-1). \tag{3.44}$$

A WD cascade of unit elements is given in Fig.3.11, where the multipliers $\{\gamma\}$ are given by

$$\gamma_i = \frac{R_{i-1} - R_i}{R_{i-1} + R_i}, \quad i=1(1)n, \quad R_0=R. \tag{3.45}$$

Depending on whether the recursion terminates in an open or short circuit, the constant k will be 1 or -1, respectively.

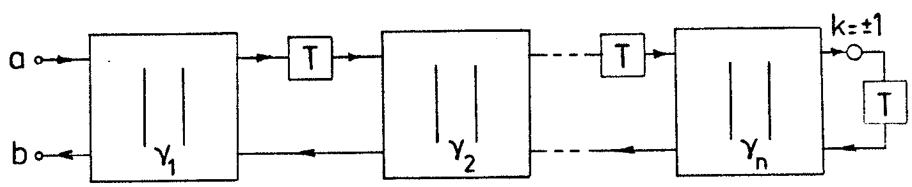


Figure 3.11 Realization of an impedance by a chain of unit elements.

Different choices for the method of realizing S_1 and S_2 will yield a different set of multipliers, some of which may lead to simpler hardware realizations than others. Wegener has established some rules for this choice [45,49], which attempt to minimize a multiplier's sensitivity in the neighborhood of its nominal value. The synthesis of WD lattice filters is of secondary importance here, and so their realization will not be dealt with in further detail.

3.3. Digital Filter Implementation

For a digital filter to be implemented as a digital algorithm using binary arithmetic (generally two's complement), the filter multipliers must be expressed as binary fractions of the form

$$\gamma = \sum_{i=m}^{n} \delta_i \, 2^i, \quad \delta_i \in \{0,1\}, \; m \le i \le n, \tag{3.46}$$

where the multiplier wordlength is defined as $w = n - m + 1$.

If the multipliers can be implemented as a sequence of multiplications or divisions by a multiple of two and additions (shift and add), the use of actual hardware multipliers can be avoided. The sensitivity properties of WDFs generally allow the multipliers to be of low wordlength compared to other structures, assuming that some design margin exists, and hence fewer shifts and adds will be required.

Usually the operation of negation is simpler to implement than addition. In such a case it is advantageous to express the multiplier in canonical signed digital code (CSDC), which is of the form

$$\gamma = \sum_{i=m}^{l} d_i 2^i, \quad d_i \in \{0, \pm 1\}, \quad m \leq i \leq l \tag{3.47}$$

such that

$$d_i \, d_{i-1} = 0 \quad for \ every \ i \tag{3.48}$$

and

$$l \leq n+1. \tag{3.49}$$

Elements of CSDC have a canonic number of non-zero digits [80], and so require the minimum number of additions when implemented using the shift-add method.

The excellent sensitivity properties of WDFs allow significant reductions in multiplier wordlength requirements, and correspondingly low roundoff noise. Due to the interaction of roundoff noise and dynamic range [10] one can expect good dynamic range behavior. However, some scaling of internal variables is necessary to produce the optimal overall dynamic range, that is a balance between the level of roundoff noise and the probability of arithmetic overflow for all nodes having the potential for overflow. The L_2-norm scaling of Jackson *et al* [81] can always be used to achieve this. To avoid the introduction of additional multipliers for scaling, the scale values are approximated by simple shifts and are absorbed into the filter structure wherever possible.

The actual implementation of digital filters will take the form of an interconnection of adaptors, delay elements, inverters, and possibly including pairs of inverse multipliers for scaling. Each adaptor is described in terms of its scattering matrix **S**, from which the adaptor output signals are calculated from the adaptor inputs. For each of n outputs a calculation will be required of the form

$$b_i = \sum_{j=1}^{n} s_{ij} a_j . \tag{3.50}$$

It is advantageous to calculate this inner product as efficiently as possible. We now examine two distributed arithmetic methods for this purpose.

3.3.1. Shift-and-Add Algorithm

An algorithm, developed by Moon and Martens [82], allows the inner-product expression to be computed as a series of shifts (division by two in binary arithmetic) and additions, weighted by a unimodular factor $c \in \{0, \pm 1\}$.

Since the entries of **S** are binary fractions we may write

$$s_{ij} = \sum_{k=0}^{q_i} d_{ijk} 2^{-k}, \qquad d_{ijk} \in \{0,1\} \tag{3.51}$$

or equivalently in CSD code

$$s_{ij} = \sum_{k=0}^{q_i} c_{ijk} 2^{-k}, \quad c_{ijk} \in \{0, \pm 1\} \tag{3.52}$$

We may express the inner product (3.50) such that the only multiplication is by an integer power of two, thus,

$$\begin{aligned} b_i &= \sum_{j=1}^{n} \sum_{k=0}^{q_i} c_{ijk} 2^{-k} a_i \\ &= \sum_{k=0}^{q_i} \left(\sum_{j=1}^{n} c_{ijk} a_j \right) 2^{-k} \\ &= \sum_{k=0}^{q_i} x_k 2^{-k} \\ &= (\cdots (((x_{q_i}/2) + x_{q_i-1})/2 + x_{q_i-2})/2 + \cdots x_1)/2 + x_0. \end{aligned} \tag{3.53}$$

The use of CSDC ensures that the minimum number of additions will be required. Also, common partial sums among $\{x_k\}$ for each output may be removed so as to further reduce the number of additions. The actual implementation takes the form of a specialized structure, different for a different set of coefficients, and consisting of a near- minimum number of adders and shifters.

3.3.2. Stored-Product Algorithm

An alternative way of expressing the inner product (3.50) developed by Peled and Liu, and Crosier *et al* [83,84], is based on representing an input signal a_j in two's complement form

$$a_j = -a_{j0} + \sum_{k=1}^{r-1} a_{jk} 2^{-k}, \qquad a_{jk} \in \{0, \pm 1\}, \tag{3.54}$$

where a_{j0} is the sign bit and r is the wordlength. Substituting this equation in the inner product (3.50) yields

$$\begin{aligned} b_i &= \sum_{j=1}^{n} s_{ij} \sum_{k=1}^{r-1} a_{jk} 2^{-k} - a_{j0} \sum_{j=1}^{n} s_{ij} \\ &= \sum_{k=1}^{r-1} \left[\sum_{j=1}^{n} s_{ij} a_{jk} 2^{-k} \right] - a_{j0} \sum_{j=1}^{n} s_{ij} \\ &= \sum_{k=1}^{r-1} e_{ik} 2^{-k} - e_{i0}. \end{aligned} \tag{3.55}$$

Since $e_{ik} \in \{0,1\}$, the e_{ik} can have only 2^r discrete values, in practice often few enough for them to be stored in a lookup table. This is known as the stored-product method. An actual implementation consists of the memory lookup table containing the partial products e_{ik}, an adder, shifter, and some registers. To change the coefficients the memory contents need only be changed.

Both of the above two methods circumvent the use of hardware multipliers via distributed arithmetic and specialized hardware. Current technology has made available general-purpose signal processors, most of which are capable of fast hardware multiplication, so the problem of minimizing realization requirements (in this case code size or execution speed) is redefined in terms of the new processing resource. At least one implementation of WD filters has been presented [85] which utilizes the features of a single-chip signal processor such as the Intel 2920. The problem of implementing digital filters on general processors deviates from the current topic and will not be discussed here.

4. OPTIMIZATION AND EXAMPLES

A digital filter is derived from a nominal discrete-time design by approximating the signal quantities and filter coefficients to a number of bits sufficient to meet the design specifications. The cost of a digital filter realization depends on the complexity of the digital filter algorithm, whether it is realized as specialized hardware components or as software, and on the signal and multiplier wordlengths. In both of the synthesis techniques discussed in the previous chapter, only a minimal number of delays and multipliers are required, so any reduction in complexity depends on the way in which the canonic number of multipliers is implemented, and on the signal wordlengths used.

Multiplier wordlength has a large effect on filter complexity, since the increase in signal wordlength due to multiplication implies that more hardware will be required to carry or store the resultant signals, and may place greater wordlength demands on subsequent operations. Therefore multiplier wordlength is an important component of a filter's complexity figure of merit.

Often it is feasible to implement multiplications by the shift-add method, in which a filter's complexity depends on the number of shifts (equivalent to multiplier wordlength) and additions needed. In the case of a fixed-flowgraph structure, a parallel adaptor for example, the number of additions required to implement it is the sum of a fixed number required to implement the flowgraph, plus a variable number required to implement the multipliers as a sequence of shifts and additions (or subtractions). Hence, it is desirable to include this variable number of additions in a filter's complexity figure of merit, if the filter is to be implemented using the shift-add method.

In the case of the unsimplified Brune adaptor, it cannot be implemented as a simple fixed flowgraph, but instead could be implemented as a matrix-by-vector multiplication. Reduction of the number of operations required to implement the overall multiplication would then be the goal, and not simply the requirements of the design parameters. As it is generally cumbersome to calculate the number of additions required to implement a matrix multiplication, we will assume that a reduction wordlength of the entries of the matrix will result in a reduction in the number of additions as well. Also, overall wordlength reduction of the matrix entries is desirable when the stored product method is used.

Two measures will be used to evaluate a digital filter's merit. The first describes the degree to which the design specifications are met, and the second describes the relative realization requirements. We now formulate the first of these two functions.

4.1. Objective Function for the Frequency Response

Consider the frequency response of a filter $H(\psi,\gamma)$ having a parameter vector γ, and an attenuation function defined as

$$A(\omega,\gamma) = -20\log_{10} \mid H(j\tan\frac{\omega}{2},\gamma) \mid \tag{4.1}$$

We define an error function

$$F_1 = \max_{\omega \in [0,\pi]} (\epsilon_p, \epsilon_s) / F_{\max} \tag{4.2}$$

where

$$\epsilon_p = \max_{\omega \in [0,\omega_p]} (A(\omega,\gamma) - A_0) / \delta(\omega), \tag{4.3}$$

is the passband error,

$$\epsilon_s = \max_{\omega \in [\omega_s,\pi]} \delta(\omega) / (A(\omega,\gamma) - A_0) \tag{4.4}$$

is the stopband error, $\delta(\omega)$ is a tolerance function describing the design specifications, and A_0 is a gain constant chosen so that

$$A_0 = \min_{\omega \in [0,\pi]} A(\omega,\gamma). \tag{4.5}$$

The constants ω_p and ω_S are the passband and stopband cutoff frequencies, respectively. The constant $F_{\max}$ is chosen, typically equal to unity, to allow a tradeoff between realization requirements and slight deviations from the specifications. The specifications are satisfied when $F_1 \leq 1$. The smaller F_1 is, the larger the design margin and the larger the expected possible improvement in multiplier values. In practice, a discrete number of frequency points are used to evaluate F_1, the number and locations of which are best determined through some experimentation; placing a greater number of points near the critical frequencies ω_P and ω_S is an appropriate strategy.

4.2. Objective Function for Realization Requirements

The second figure of merit describes the implementation cost of a filter. The complexity of simple and of complex adaptor realizations depends chiefly on the wordlength of the design parameters, and on the overall scattering matrix wordlength, respectively. In the case of simple adaptors, the design parameters are present as multipliers in the adaptor's flowgraph. For more complex adaptors, expressions including sums of products of the design parameters determine the matrix wordlength.

A simple adaptor's complexity, if implemented via a shift-and-add method, depends depends solely on the wordlengths of each of the design parameters and the number of additions required to implement them. (If the adaptor is implemented via the stored-product method, the number of additions required by a design parameter is irrelevant.) Typically, it is preferable that all parameters have similar wordlengths. Consider a single multiplier γ_i, expressed in CSDC,

$$\gamma_i = \sum_{j=0}^{m_i} d_{ij} 2^{-j}, \qquad d_{ij} \in \{0, \pm 1\}. \tag{4.6}$$

Therefore the number of additions required to implement γ_i is given by

$$D_i = \sum_{j=0}^{m_i} |d_{ij}| - 1. \tag{4.7}$$

We may then define an objective function describing relative hardware requirements as

$$F_2 = \sum_{i=1}^{n} D_i \, 2^{m_i}. \tag{4.8}$$

A change in multiplier wordlength most greatly affects F_2.

A slightly different approach is required for more complex adaptors, such as the general Brune. In their implementation, the scattering matrix entries are the multipliers, and each entry is a function of the design parameters. There are three distinct parameter definitions possible for a Brune adaptor having no reflection-free ports, and three definitions for a Brune adaptor with one reflection-free port. Instead of calculating the S-matrix to evaluate its wordlength, it is simpler to evaluate the wordlength of two matrices of which it is composed, the voltage transfer matrix **K**, and the turns ratio matrix **N**. The entries of the turns-ratio matrix **N** contain only simple occurrences of one design parameter, so reduction of the wordlength of **K**, which contains SOP functions of the design parameters, is the prime consideration.

Examine the **K**-matrix in terms of the design parameters $\{\alpha\}$.

$$\mathbf{K} = \begin{bmatrix} \alpha_2\alpha_3 & \alpha_3(1-\alpha_2) + \alpha_1(1-\alpha_1) \\ \alpha_2\alpha_4 & \alpha_4(1-\alpha_1-\alpha_2) \end{bmatrix}. \tag{4.9}$$

It can be readily calculated in terms of each of the parameter sets defined in Chapter 2. In terms of parameter set 1 for a non-reflection-free Brune adaptor (3.15) we have

$$K_{11} = [1-\beta_1(1-\beta_2-\beta_3)\beta_4]\beta_2 \tag{4.10}$$

$$K_{21} = [1-\beta_1(1-\beta_2-\beta_3)\beta_4]\beta_3$$

$$K_{12} = \beta_2\beta_4[1-\beta_2-\beta_1\beta_3]$$

$$K_{22} = \beta_3\beta_4[\beta_1(1-\beta_3) - \beta_2].$$

The wordlength of **K** is then

$$Q_k = \max(q_{11}, q_{12}, q_{21}, q_{22}) \tag{4.11}$$

where q_{ij} is the wordlength of K_{ij}. We choose to define an objective function F_2 to be

$$F_2 = 2^{Q_K}. \tag{4.12}$$

which would be appropriate for either stored-product or shift-add realizations. Similar expressions can be derived for the two other parameter sets, and for the cases of port one or port two reflection-free adaptors.

The wordlength Q_S of the scattering matrix must satisfy

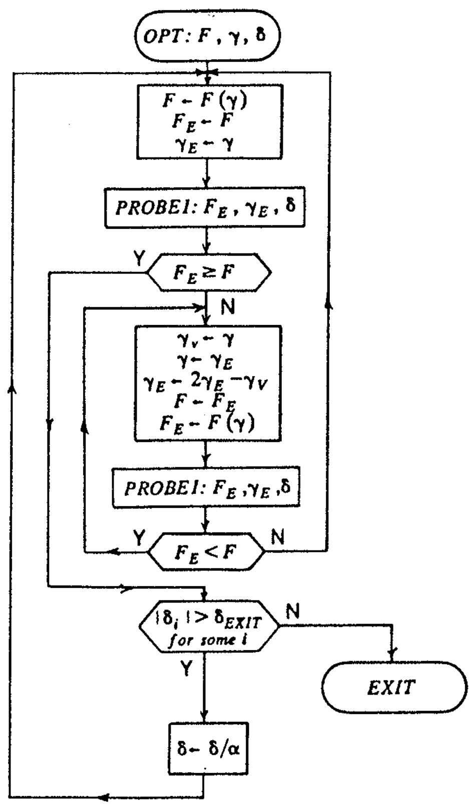
$$Q_S \le (Q_K - 1) + Q_N \tag{4.13}$$

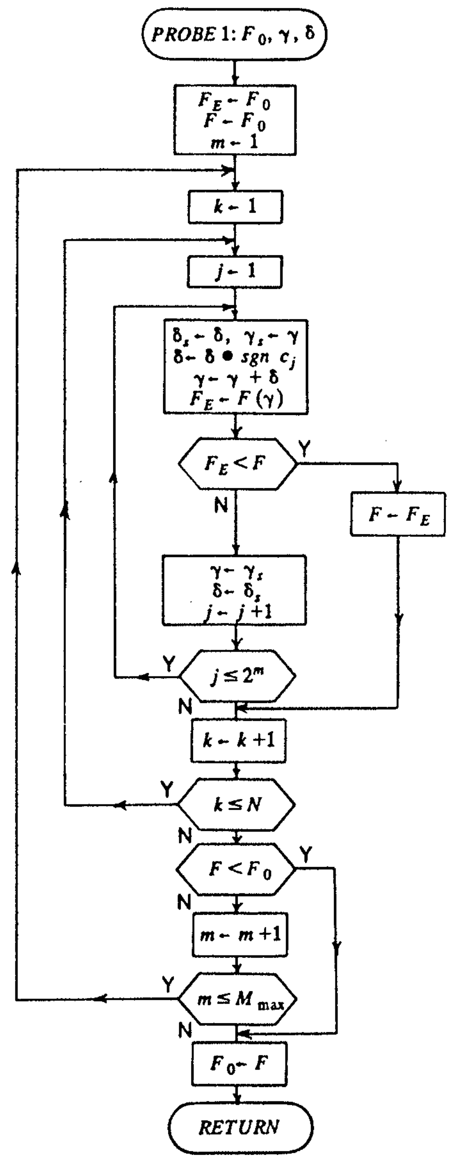
where Q_N is the wordlength of the turns-ratio n. Since reduction of Q_K and Q_N does not guarantee a minimum number of additions for S, this is partially solved by also considering the wordlengths of individual design parameters during optimization.

Given the objective functions F_1 and F_2 which characterize the relative merit of a filter's frequency response and coefficient realization requirements, a suitable optimization scheme will find a parameter set $\{ \gamma \}$ which minimizes F_2 subject to $F_1 \le 1$.

4.3. Search Algorithm

Heuristic schemes have been presented based on the well-known Hooke and Jeeves pattern search [86] and which have given good results [49,63]. It is a univariate search with an acceleration feature. A multivariate version of the search is illustrated in Fig.4.1. It has the advantages of simplicity and the ability to conform to the restrictions of a parameter space consisting of a uniform rectangular grid. However, it is a continuous optimization algorithm and as such is not appropriate for the minimization of a wordlength-based objective function such as F_2.





Flowchart Symbol Glossary

F, F_1	- frequency response objective function
γ	- vector of coefficients, a point in N-dimensional discrete parameter space
δ	- step size vector
Q	- wordlength vector
F_E	- objective function at point resulting from probe movement
γ_E	- point resulting from probe movement
N	- number of coefficients
●	- indicates direct product
F_2	- wordlength objective function
F_{1X}	- maximum allowable value for F_1
M_{max}	- maximum number of coefficients to be varied at once
c_j	- vector containing j^{th} pattern for probe movement

Figure 4.1 Multivariate pattern search. Also used as a subroutine for Fig. 4.2.

A number of modifications have been made to the pattern search to better suit the characteristics of F_2. The number of variables that can be varied at once has been generalized to account for the strong effect of the interaction of parameters in the Brune section K-matrix. Different parameter index orderings have been made possible, the most useful of which is based on the parameter sensitivity with respect to F_1, so that the wordlength of the most sensitive parameters will be reduced first. Each parameter is given a different probe step size according to its current wordlength, so that during the search only grid points which offer an improvement in wordlength will be tested. The contraction step and exit criterion are unnecessary and have been eliminated

The discrete search is designed to be part of an algorithm having the following features. Any subset of parameters can be optimized while allowing those parameters not yet optimized to be varied freely so as to attempt to satisfy $F_1 \leq 1$ while taking advantage of improvements in F_2 based on the parameter subset. This is to allow each section of a cascade realization, or parameters whose interaction greatly affects F_2 to be considered together. Also, the order in which these subsets are optimized can be chosen to allow sections which are expected to demand the greatest realization requirements to be considered earlier in the process, and so take advantage of the larger design margin available then. Other optimization schemes can and have been used, although only a global search guarantees optimality. The approach given lends versatility to allow a compromise between computational requirements and the quality of the final solution.

A more detailed description of the discrete search follows; a flowchart diagram is given in Fig.4.2 (Appendix B contains a program listing). The search is applied to the task of minimizing F_2 under the condition that $F_1 \leq 1$ is maintained. An initial point in discrete parameter space with $F_1 \leq 1$ must first be available. Such a point can be found by approximating the coefficients of the nominal design to sufficiently long wordlengths. Beginning at the initial basis point γ, a probe operation is performed in which the objective function F_2 is evaluated at neighboring points. If any improvement in F_2 is found, say at γ_E, $F_1(\gamma_E)$ is evaluated. If $F_1 > 1$, and some parameters have not been included in this search and are therefore free, an optimization is performed on them to minimize F_1. If $F_1 \leq 1$, a new point is calculated by extrapolating through the γ_E and the search is started anew with this as the new basis point. If no improvement is found, then the previous best point is used for the new basis point. After two consecutive failures the search stops.

The largest proportion of computational effort is expended in the calculation of F_1, thus to save time the condition $F_1 > 1$ is recognized early in order to avoid unnecessary exact

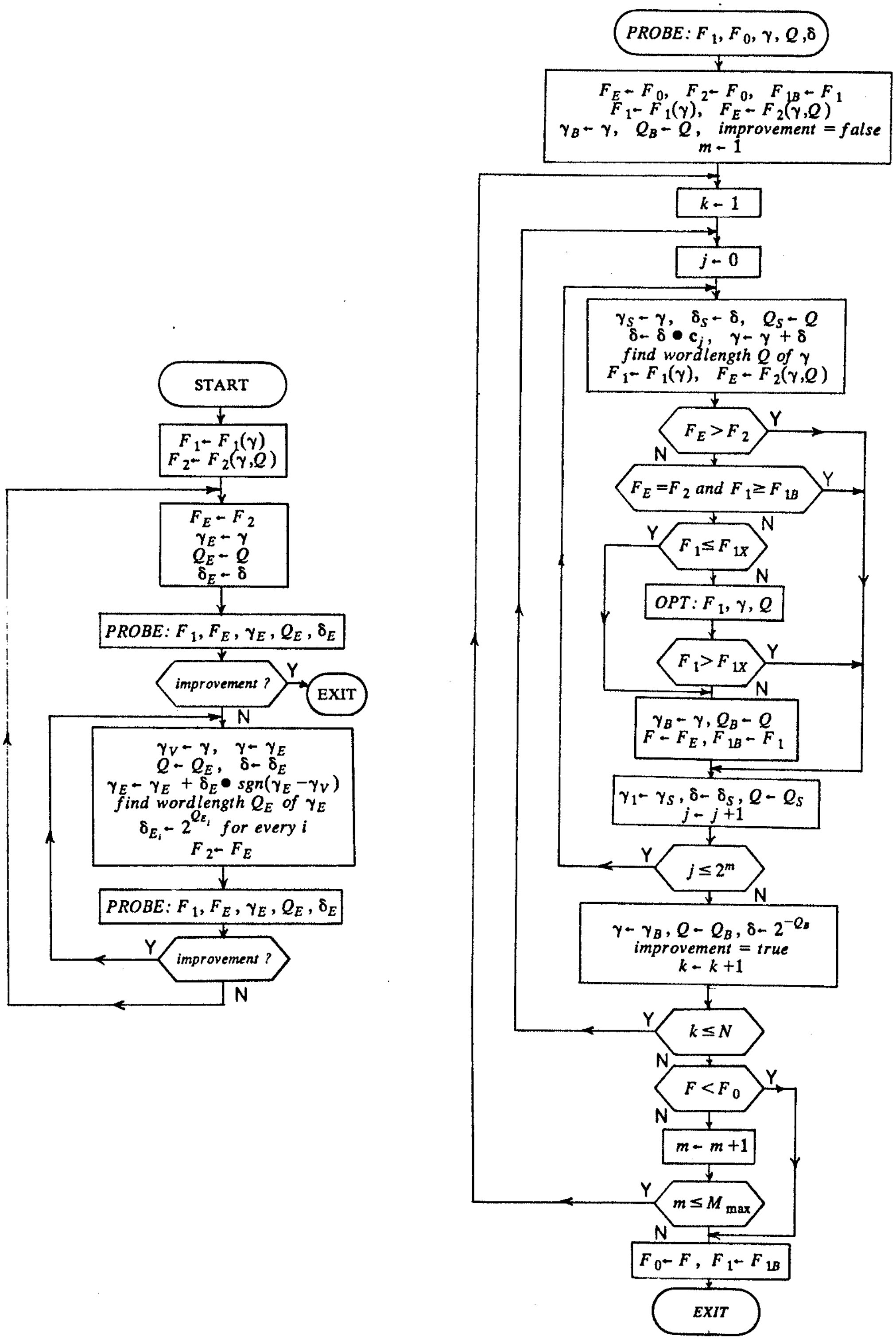


Figure 4.2 Pattern search for minimizing wordlength requirements.

calculation. The response at critical frequencies, those most likely to deteriorate, are evaluated first.

A useful property of WDFs is that each is a direct mapping via the bilinear transform to an analog network. This is convenient for optimization applications since the frequency response can conveniently be calculated from a filter's corresponding analog equivalent using the frequency variable mapping

$$\Omega = \tan(\pi f / F_S), \qquad F_S = \textit{sampling frequency}. \tag{4.14}$$

Several examples are presented in the next section to illustrate the design and optimization procedures. The search used is designed for use with cascade filter realizations which include Brune sections. However good results were obtained for the WD lattice filters, and so these are included to illustrate the search's merit as a more general tool. For smaller examples, it is often feasible to use an exhaustive search approach economically. However, for filters of order $\geq$ 7 this is generally not feasible. A comparison with results obtained using a global search is given for fifth-order filters to show the consistently good results of the proposed technique.

4.4. Design Examples

4.4.1. Fifth-Order Ladder Filters

Example 1

The first example is described by the following specification :

$$A \leq 0.3\ dB, \quad f \in [0,3.4]\ kHz,$$
$$A \geq 32.0\ dB, \quad f \in [4.6,16.0]\ kHz, \quad F_S = 32\ kHz.$$

These specify a digital filter used in an interpolator which increases the sampling rate from 8 kHz to 32 kHz [36]. To find a suitable analog prototype, we calculate the corresponding analog critical frequencies,

$$\Omega_p = \tan\pi \frac{f_p}{F_S} = 0.34677, \qquad \Omega_s = \tan\pi \frac{f_s}{F_S} = 0.48503, \qquad \Omega_s / \Omega_p = 1.39872\ .$$

From the design tables [87], we find that a 5th order elliptic filter designated CC051548 will satisfy the specifications, and allow some margin in the passband and stopband. The topology of the analog prototype and a WD equivalent are given in Fig.4.3, in which

$$\hat{G}_5 = G_5 + G_7, \quad \hat{G}_6 = G_6 + G_9, \quad \hat{G}_7 = G_8 + n_1 G_7 + n_2 G_9, \tag{4.15}$$

and the analog element values and corresponding design parameters are given in Table 4.1. Design equations for this configuration of adaptors are listed in Appendix C. Although other adaptor configurations are also feasible, the symmetrical structure given here is generally preferable.

It is desirable that the constraints

$$G_3 G_7 = G_1 G_5, \qquad G_4 G_9 = G_2 G_6 \tag{4.16}$$

be applied so as to simplify the two Brune sections, if the specifications can still be met. A fixed-flowgraph realization will then be available. The constraint is most easily applied to G_1 and G_2, and so we have

$$G_1 = G_3 G_7 / G_5 = 1.40509, \quad G_2 = G_4 G_9 / G_6 = 0.47316.$$

Using the objective function F_1 of (4.2) and the frequency response algorithm of Appendix D, it was found that the imposition of the two constraints perturbed the frequency response beyond tolerance limits. A continuous optimization, based on the Simplex algorithm [88], was employed to find a suitable nominal design.

An expression describing hardware requirements for this configuration is given by

$$F_2 = \sum_{l=1}^{3} Q_l, \tag{4.17}$$

where

$$Q_l = \sum_{i=1}^{2} D_{li}\, 2^{m_{li}} \tag{4.18}$$

and D_{li} is the number of non-zero bits in the CSDC representation and m_{li} is the wordlength of the i^{th} multiplier of the l^{th} section. The optimization procedure was applied to the filter using the F_1 given by (4.2) and the F_2 given above. Three stages were used, one for both Brune adaptors, then two for each parameter of the parallel adaptor. The program found the design given in Table 4.2, which is identical to the one found using a direct (exhaustive) search approach. Its frequency response is presented in Fig.4.4. Only 19 adders are required for a shift-add implementation of the filter: 7 for the first Brune adaptor's flowgraph, 5 for the second Brune, 6 for the parallel adaptor flowgraph, and 1 addition due to the multipliers. A total of 14 050 evaluations of F_1 were used, requiring the filter frequency response to be

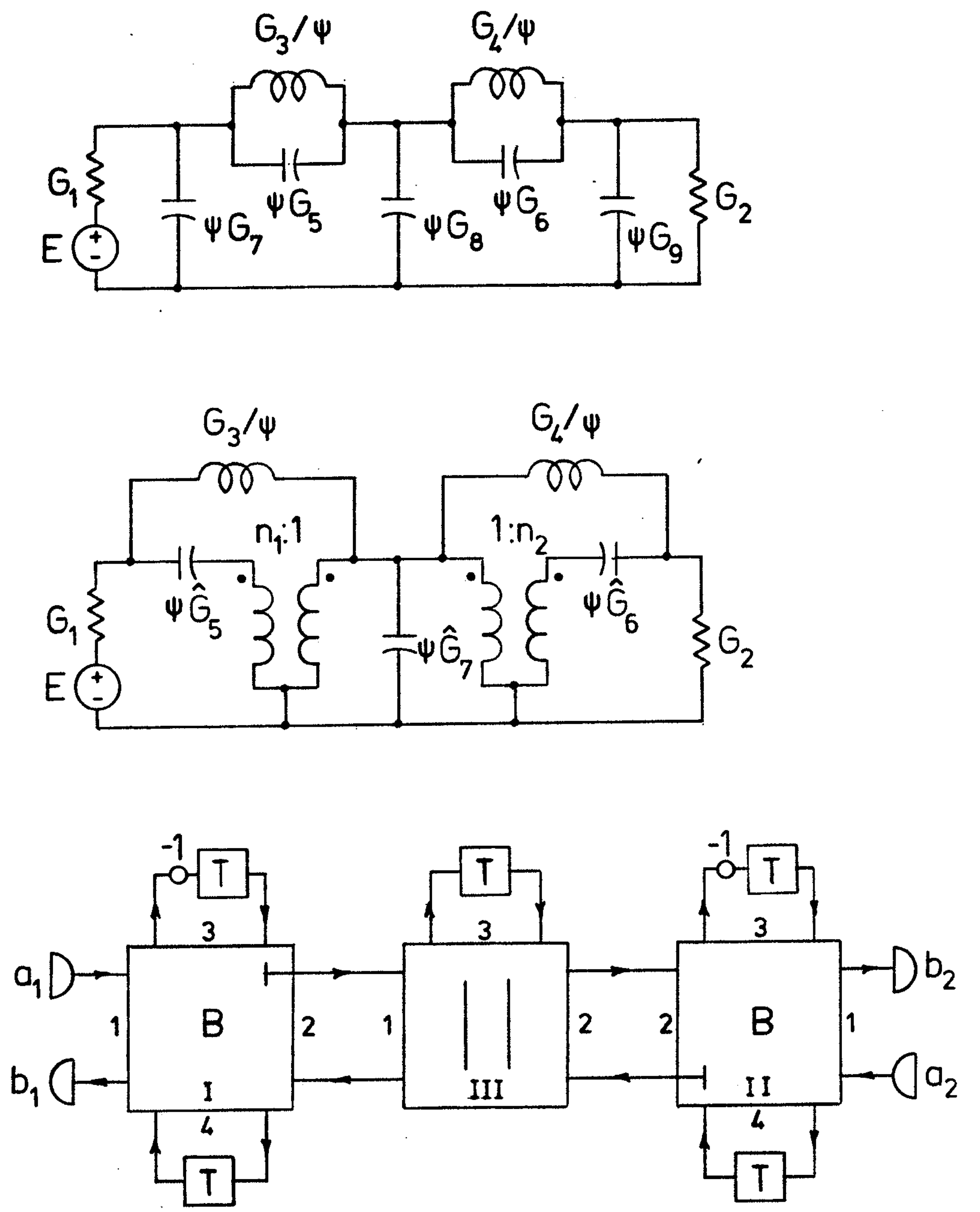


Figure 4.3 Fifth-order elliptic ladder filter, its equivalent network involving Brune sections, and a corresponding WDF.

evaluated at 112 940 points; 713 evaluations of F_2 were needed. Only 7.93 seconds of Amdahl 5850 CPU time was expended compared to the 2 minutes, 55 seconds used by the direct search, which required 2 270 268 evaluations of F_1 using 2 557 670 frequency points. (The final

Table 4.1 Initial Design for Example 1.

n	Conductances normalized	Conductances denormalized	Parameters
1	1.0	1.0	0.331313
2	1.0	1.0	0.188890
3	0.86162	0.29878	0.175356
4	1.24601	0.43208	0.477310
5	0.20911	0.60303	0.038815
6	0.64341	1.85542	0.035578
7	0.98340	2.835884	
8	1.52876	4.40858	
9	0.70459	2.031861	

solution for direct search was picked by hand from a number of feasible designs.)

Table 4.2 Final Conductances and Parameters for Example 1.

n	Conductances	Parameters: rational	Parameters: CSDC	Parameters: adaptor
1	48	1/4	.0 1	1
2	9	1/4	.0 1	1
3	16	1/4	.0 1	2
4	3	1/4	.0 1	2
5	48	1/16	.0 0 0 1	3
6	9	3/256	.0 0 0 0 0 1 0 -1	3
7	144			
8	135			
9	27			

Example 2

A second example is specified by the following requirements:

$$A \leq 0.7\ dB,\ f \in [0,\ 3.68]\ kHz,$$
$$A \geq 45.0\ dB,\ f \in [6.56,\ 16.0]\ kHz, \quad F_S = 32\ kHz.$$

These describe a filter first presented by Wanhammer [89]. From the design tables, we find that the fifth order elliptic filter designated CC052532 will satisfy the requirements when frequency-scaled and allow a fair margin, particularly in the passband. The initial element values are given in Table 4.3, referred to the topology of the previous example. We attempt to use the simplifying constraints (4.16), yielding

$$G_1 = G_3 G_7 / G_5 = 4.85404, \qquad G_2 = G_4 G_9 / G_6 = 1.83613,$$

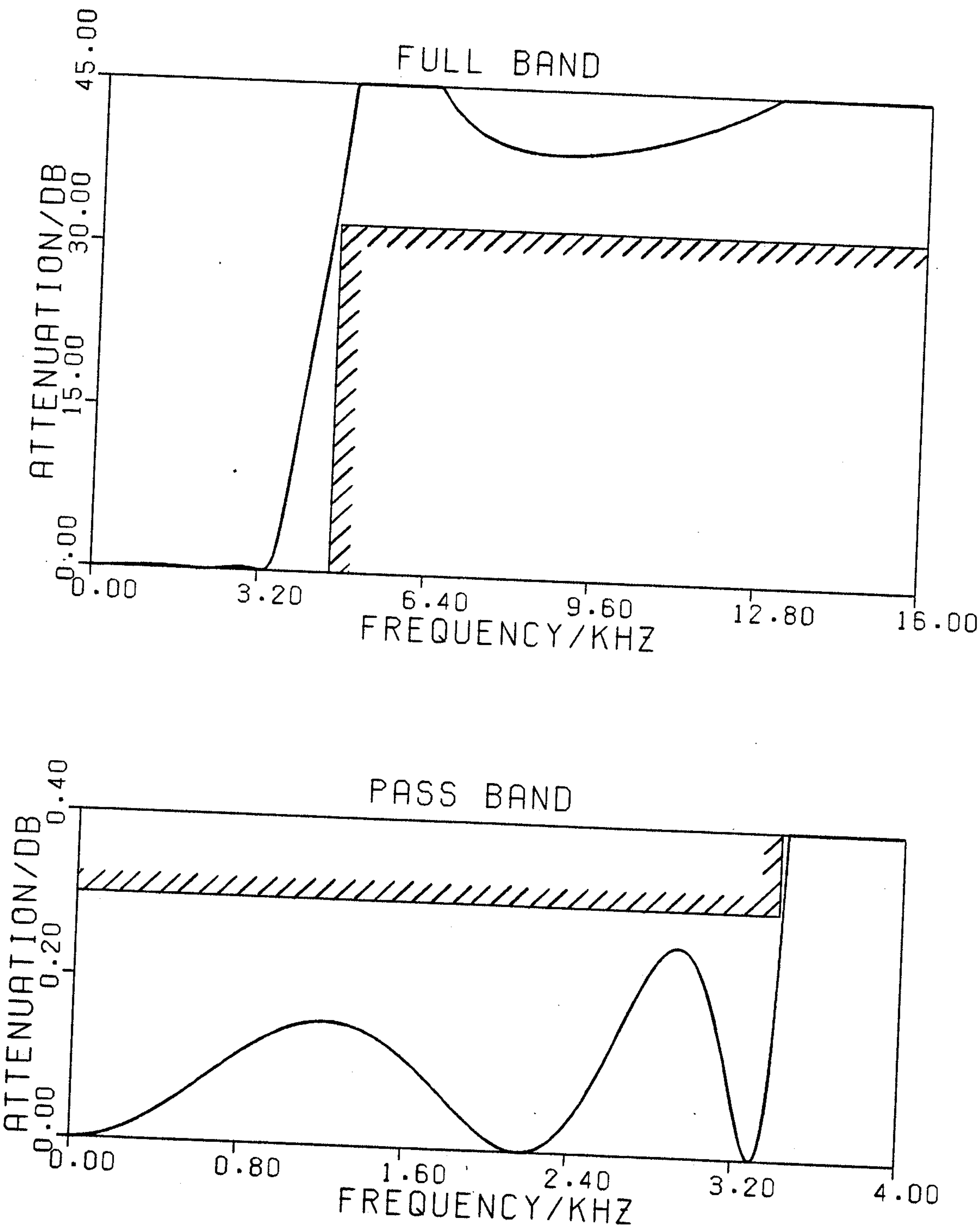


Figure 4.4 Frequency response for Example 1.

Table 4.3 Initial Conductances and Parameters for Example 2.

n	Conductances		Parameters
	normalized	denormalized	
1	1.0	1.0	0.57012
2	1.0	1.0	0.35644
3	0.81617	0.30841	0.059740
4	0.92697	0.35027	0.16021
5	0.08787	0.23254	0.04290
6	0.23897	0.63241	0.043518
7	1.3830	3.660	
8	2.05013	5.42551	
9	1.25268	3.31512	

from which a nominal parameter set is obtained. This design does not satisfy the specifications, so a continuous optimization is again used to obtain a satisfactory set. The resulting parameter set allows some design margin. The objective function F_2 used for the previous example will again be suitable.

As before, the Brune sections are first optimized together, followed by one stage for each of the two parameters of the parallel adaptor. The parameter set which resulted, given in Table 4.4, required 9287 evaluations of F_1, a total of 54 714 frequency points, and 414 evaluations of F_2. An equivalent parameter set was yielded by a direct search approach, which required 712 800 evaluations of F_1 and a total of 906 204 frequency points. The frequency response is given in Fig.4.5.

A shift-add implementation of the design will require 23 adders: 18 for the adaptors, as in Example 1, plus 5 adders due to the multipliers.

Table 4.4 Final Conductances and Parameters for Example 2.

n	Conductances	Parameters		
		rational	CSDC	adaptor
1	105	5/8	.1 0 1	1
2	45	3/8	.1 0 -1	1
3	15	1/8	.0 0 1	2
4	15	1/4	.0 1	2
5	9	7/128	.0 0 0 1 0 0 -1	3
6	25	3/64	.0 0 0 1 0 -1	3
7	63			
8	189			
9	75			

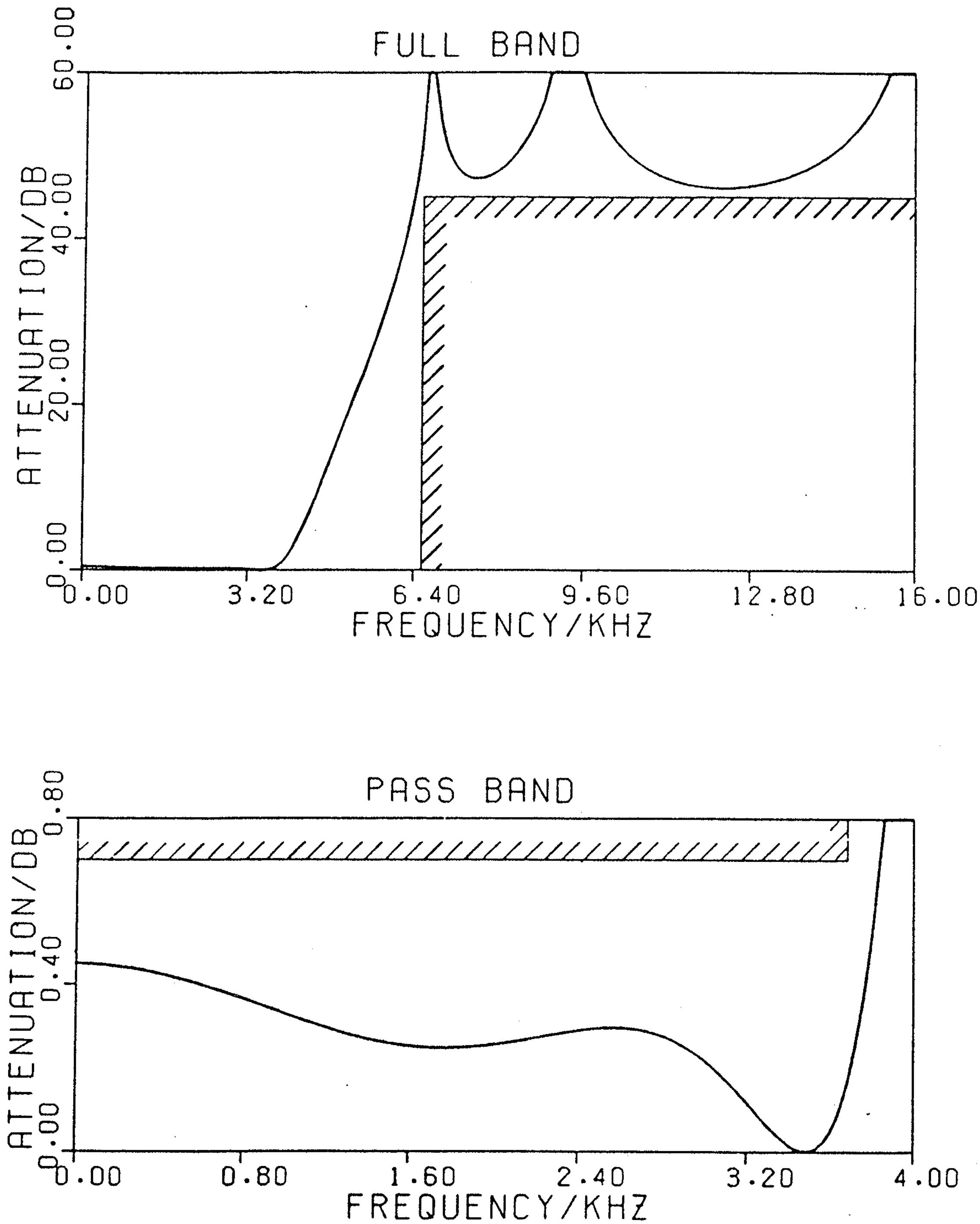


Figure 4.5 Frequency response for Example 2.

Example 3

Consider the following specifications:

$$A \le 0.75\ dB, f \in [0, 2.4]\ kHz$$
$$A \le 1.45\ dB, f \in [2.4, 3.0]\ kHz$$
$$A \le 2.88\ dB, f \in [3.0, 3.4]\ kHz$$
$$A \ge -0.75\ dB, f \in [0, 3.4]\ kHz$$
$$A \ge 40.7\ dB, f \in [4.6, 32]\ kHz, \quad F_S = 64\ kHz.$$

These describe a digital filter used in an interpolator which increases the sampling rate from 8 kHz to 64 kHz [45]. From Saal [90] we find that the 5th-order elliptic filter designated CC055048 can satisfy the specifications. A set of element values are given in Table 4.5. As usual we attempt to impose the the constraints

$$G_1 = G_3G_7 / G_5 = 1.45343, \qquad G_2 = G_4G_9 / G_6 = 0.56872$$

and use the conductances to calculate a nominal design parameter set. This nominal design still satisfies the specifications despite the imposition of (4.16), but a continuous optimization was performed to improve the design margin. We may again use the objective function of the previous examples.

Table 4.5 Initial Conductances and Parameters for Example 3.

n	Conductances		Parameters
	normalized	denormalized	
1	1.0	1.0	0.10469
2	1.0	1.0	0.052097
3	1.12338	0.18925	0.11521
4	1.49094	0.25116	0.30634
5	0.27264	1.61839	0.0084788
6	0.76988	4.570	0.0088221
7	2.09392	12.4295	
8	2.49416	14.8053	
9	1.74325	10.348	

Now we may apply the optimization procedure to minimize F_2. We choose to minimize the Brune adaptors first, followed by the parallel adaptor in the same manner as for the previous examples. The parameter set given in Table 4.6 was obtained, and required a total of 5558 evaluations of F_1, 57 305 frequency points to be checked, and 421 evaluations of F_2. The direct search approach yielded the same design but required 425 984 evaluations of F_1 and 748 977 frequency points to be checked. Its frequency response is given in Fig.4.6. Only 20 adders are required to implement the design.

Table 4.6 Final Conductances for Example 3.

n	Conduc-tances	Parameters		
		rational	CSDC	adaptor
1	42	3/32	.0 0 1 0 -1	1
2	21	1/16	.0 0 0 0 1	1
3	6	1/8	.0 0 1	2
4	7	1/4	.0 1	2
5	58	1/128	.0 0 0 0 0 0 1	3
6	105	1/128	.0 0 0 0 0 0 1	3
7	406			
8	315			
9	532			

Preceding examples have shown the merit of the proposed optimization procedure relative to an exhaustive search approach. Verification by exhaustive search was possible since the parameter space for these examples is sufficiently small. Some examples will now be presented for which exhaustive search is not feasible.

4.4.2. Seventh-Order Examples

Example 4

Consider the following specifications [45] :

$$A \le 0.11\ dB, f \in [0, 2.8]\ kHz,$$
$$A \le 0.22\ dB, f \in [2.8, 3.2]\ kHz$$
$$A \le 0.44\ dB, f \in [3.2, 3.4]\ kHz$$
$$A \ge -0.11\ dB, f \in [0, 4.0]\ kHz$$
$$A \ge 30.0\ dB, f \in [4.0, 4.8]\ kHz$$
$$A \ge 40.0\ dB, f \in [4.8, 5.4]\ kHz$$
$$A \ge 50.0\ dB, f \in [5.4, 12.0]\ kHz, \quad F_S = 24\ kHz$$

They describe a filter intended to be part of a transmultiplexer system proposed by Fettweis [91]. Its passband attenuation is 1/20 of the CCITT requirements for channel filters, and the stopband specification depends on the solution for other parts of the transmultiplexer. These can be satisfied with the 7th order equiripple elliptic transfer function designated CC072056, although with little margin in the equiripple sense. The conductance values are given in Table 4.7 for normalized and denormalized cutoff frequencies. Using the ladder network shown in Fig.4.7 as the reference filter, we may consider several WDF realizations.

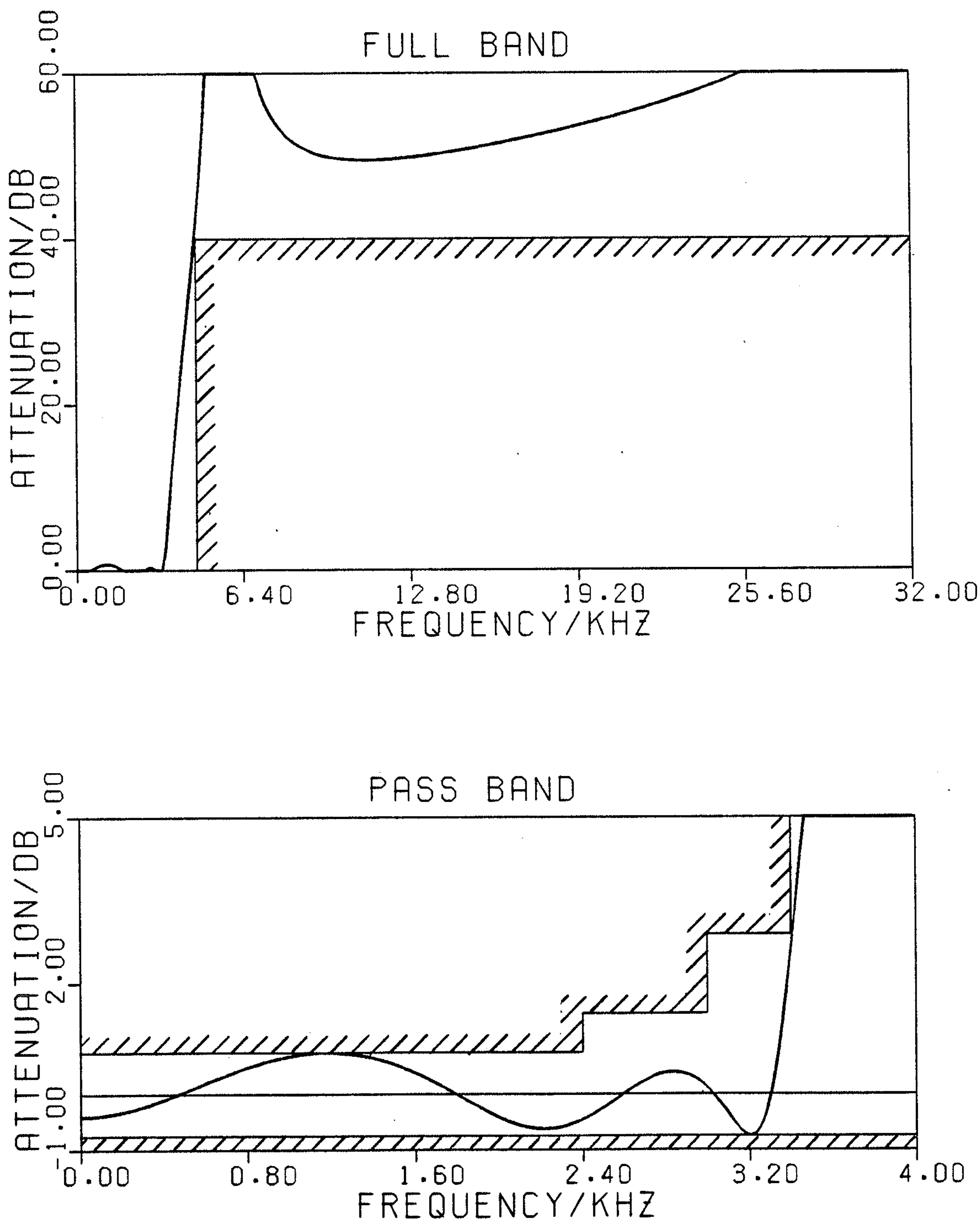


Figure 4.6 Frequency response for Example 3.

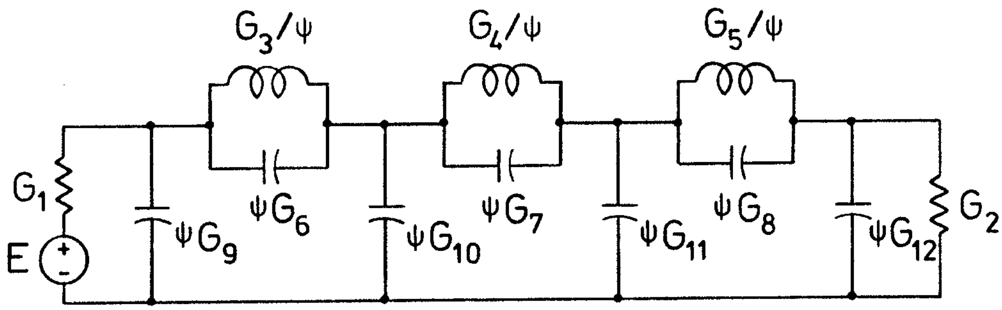


Figure 4.7 Seventh-order elliptic prototype filter containing reactive redundancies.

Table 4.7 Initial Conductances for 7th-Order Examples.

n	Conductances	
	normalized	denormalized
1	1.0	1.0
2	1.0	1.0
3	0.8161	0.3893
4	1.2210	0.58237
5	1.13685	0.54225
6	0.16141	0.33840
7	0.81628	1.71137
8	0.57779	1.21136
9	1.20207	2.5202
10	1.60392	3.36269
11	1.39365	2.9218
12	0.90654	1.9006

Given that three Brune sections and a parallel adaptor will be used, there are 16 possible configurations. We will consider three which have been chosen, somewhat arbitrarily, to include each of the three options for reflection-free ports of a Brune adaptor for the middle adaptor. As usual, we will attempt to impose the constraints

$$\frac{G_1}{G_9} = \frac{G_3}{G_6}, \qquad \frac{G_2}{G_{12}} = \frac{G_5}{G_8} \tag{4.19}$$

whenever possible so as to simplify two of the Brune adaptors so as to allow their subsequent realization as fixed-flowgraph structures.

Initially we shall attempt to implement the WDF as shown in Fig.4.8, in which

$$\hat{G}_6 = G_6 + G_9, \quad \hat{G}_7 = G_7 + G_{11} + n_2 G_{12}, \tag{4.20}$$

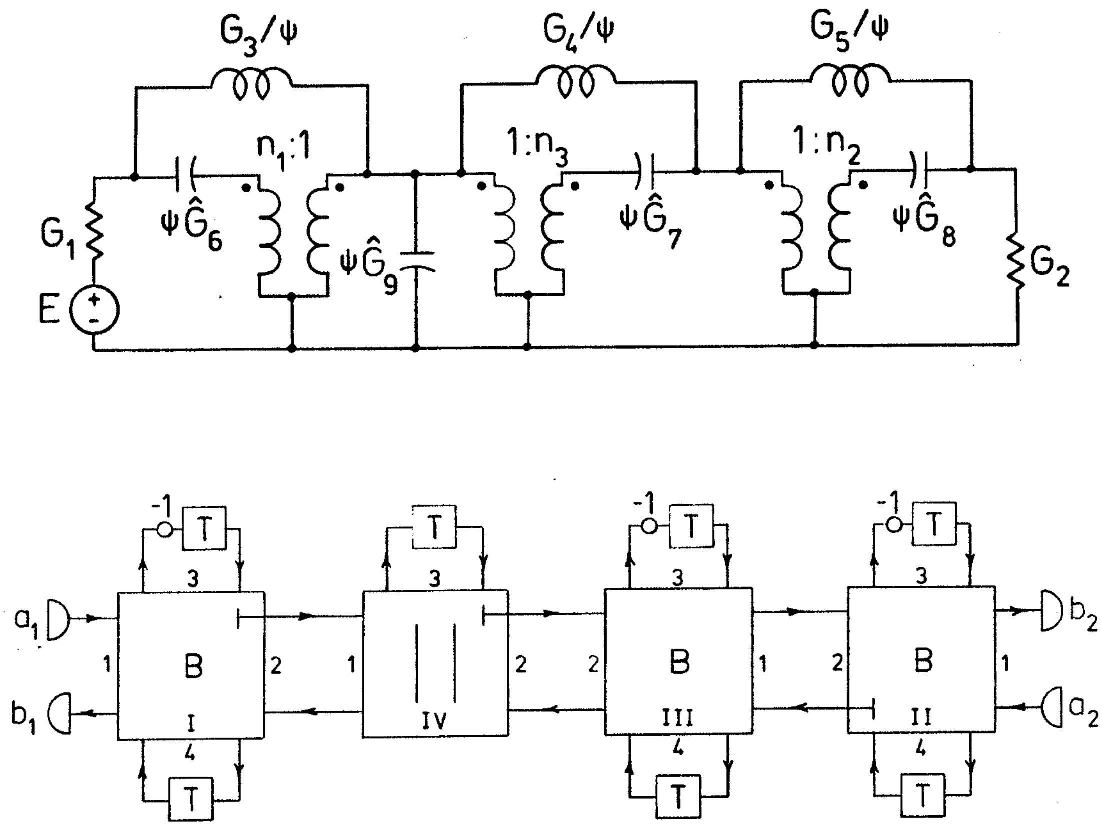


Figure 4.8 a) Seventh-order equivalent ladder filter involving Brune sections, and b) a corresponding WDF (for Example 4).

$$\hat{G}_8 = G_8 + G_{12}, \quad \hat{G}_9 = G_{10} + n_1 G_9 + n_3(G_{11} + n_2 G_{12})$$

This configuration has a non-reflection- free Brune adaptor and two adaptors having the simplifying constraints, which are

$$G_1 = G_3 G_9 \,/\, G_6 = 2.89894, \qquad G_2 = G_5 G_{12} \,/\, G_8 = 0.85078$$

We obtain the initial parameter set, given in Table 4.8, which does not satisfy the specifications. A continuous optimization was employed, and found a satisfactory set. A fair design margin exists and indicates that improvements in hardware requirements are probable.

The function describing hardware requirements may be expressed as the sum

$$F_2 = F_{2B} + F_{2S} \tag{4.21}$$

where F_{2B} is the objective function for a non-reflection-free Brune adaptor computed

Table 4.8 Initial Parameters for Example 4.

n	Parameters
1	0.11838
2	0.53495
3	0.065939
4	0.32779
5	0.26753
6	0.26677
7	5.02998
8	0.38926
9	0.30922

according to (4.12), and F_{2S} is given by (4.8), in which the parameters for adaptors one, two and four, which all have simple flowgraphs, are included.

A typical strategy for minimization is as follows. Since adaptor three is the most complex, it will be minimized first, so as to allow the largest portion of the design margin to be used in finding a favourable combination of parameters for the section. The turns-ratio, because it is present in both **N** and **K**, will be the first parameter considered, followed by the fourth parameter, since it is greater than than unity and often allows cancellations between the numerator and denominator of the entries of **K**. The remaining parameters are minimized last. Next, adaptor four is minimized, followed by adaptors one and two. Application of this strategy resulted in the design of Table 4.9, which satisfies the specifications, as can be seen from its frequency response in Fig.4.9. A total of 553 evaluations of F_2 were required, and a total of 515 614 evaluations of F_1, using 6 429 794 frequency points. Execution time for the procedure was about 9 minutes.

Excluding adaptor three, 21 adders are required, of which 5 are required to implement the multipliers. The scattering matrix for adaptor three was found to have an overall wordlength of 19 bits. An attempt to decompose the matrix into a shift-add structure yielded an impractical requirement of 70 adders. A stored-product implementation of this adaptor would be more suitable.

Example 5

As a further example consider realizing the same filter using the ladder network and WDF structure shown in Fig.4.10, in which

$$\hat{G}_6 = G_6 + G_9, \quad \hat{G}_7 = G_{10} + G_7 + n_1 G_9, \tag{4.22}$$

Table 4.9 Final Conductances and Parameters for Example 4.

n	Conductances	Parameters		
		rational	CSDC	adaptor
1	0.24195	1/4	0.0 1	1
2	0.23268	9/16	0.1 0 0 1	1
3	0.080649	1/16	0.0 0 0 1	4
4	0.11417	5/16	0.0 1 0 1	3
5	0.091048	17/64	0.0 1 0 0 0 1	3
6	0.062727	17/64	0.0 1 0 0 0 1	3
7	0.31250	5	1 0 1.0	3
8	0.10989	9/32	0.0 1 0 0 1	2
9	0.18818	29/64	0.1 0 0 -1 0 1	2
10	0.64541			
11	0.60852			
12	0.28082			

$$\hat{G}_8 = G_8 + G_{12}, \quad \hat{G}_9 = G_{11} + n_2 G_{12} + n_3(G_{10} + n_1 G_9)$$

This realization differs from the previous ones in that it includes a Brune section having port one reflection-free. An initial attempt to apply both constraints (4.19) was unsuccessful, as no satisfactory design could be found through continuous optimization. Application of the single constraint (4.19b) was attempted instead. From the conductances given in Table 4.7 we may calculate the set of design parameters given in Table 4.10. A continuous optimization algorithm found a nominal design which satisfies the specifications. We choose the wordlength objective function to be

$$F_2 = F_{2B_1} + F_{2B_2} + F_{2S}, \tag{4.23}$$

where F_{2B_1} and F_{2B_2} are evaluated according to (4.12), except that for F_{2B_2} the **K**-matrix is calculated in terms of the parameter set $\{ \beta \}$ of (3.20) for a port 1 reflection-free Brune adaptor. The term F_{2S} is computed according to (4.8) in which the parameters for the simplified Brune and parallel adaptors are included.

The order in which the adaptors in the cascade will be minimized is : adaptor three, followed by adaptor one, then adaptor four, and adaptor two last. Within each of the unsimplified Brune adaptors the turns-ratios will be reduced first, then their last parameter, followed by the remaining parameters, as described in Example 4. Application of this strategy resulted in the parameter set of Table 4.11, which was found after 182 330 evaluations of F_1, a total of 2 451 587 frequency points, and requiring 352 evaluations of F_2. The search took about 3 minutes, 32 seconds to complete. A plot of the filter frequency response is given in

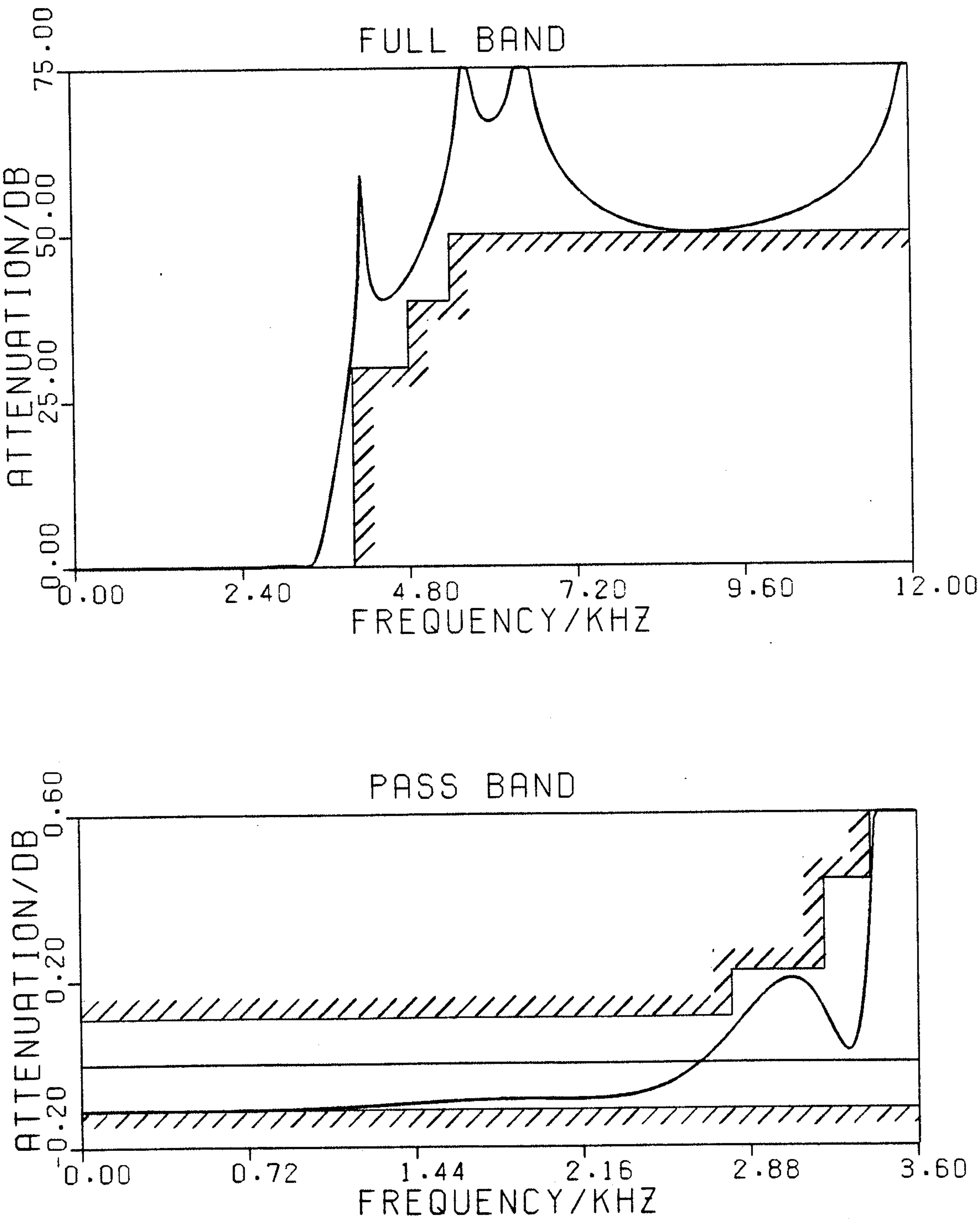
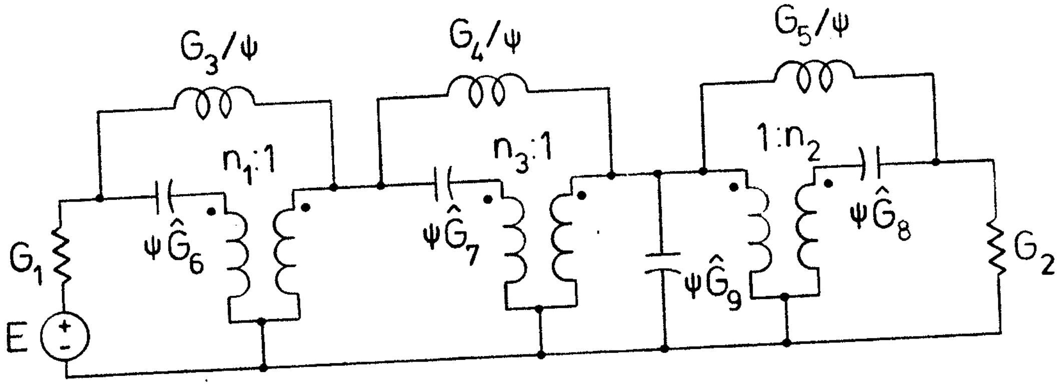


Figure 4.9 Frequency response for Example 4.



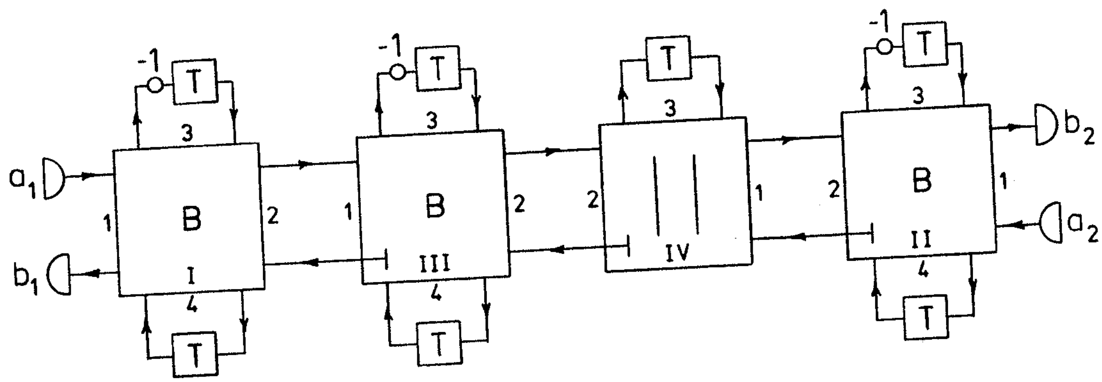


Figure 4.10 a) Alternate configuration of equivalent ladder filter involving Brune sections and b) a corresponding WDF (for Example 5).

Table 4.10 Initial Design Parameters for Example 5.

n	Parameters
1	0.11838
2	0.15225
3	0.11121
4	5.33344
5	0.31855
6	0.14233
7	3.27422
8	0.064193
9	0.38926
10	0.30922

Fig.4.12.

Since two adaptors do not have fixed flowgraphs, realization requirements for this design are greater than for the previous example. A shift-add implementation of Brune adaptors one and three is again impractical, as their scattering matrices have wordlengths of 18 and 23 bits respectively. The 18-bit scattering matrix would require 64 adders for its shift-add implementation, and so the stored-product method should be considered instead. Requirements for the adaptors two and four are 9 and 4 adders respectively.

Table 4.11 Final Conductances and Parameters for Example 5.

n	Conductances	Parameters		
		rational	CSDC	adaptor
1	2.0250	1/8	0.0 0 1	1
2	0.48693	5/32	0.0 0 1 0 1	1
3	0.82955	5/32	0.0 0 1 0 1	1
4	0.49727	5	1 0 1.0	1
5	0.26400	3/16	0.0 1 0 -1	3
6	0.62500	1/16	0.0 0 0 1	3
7	1.4027	9/2	1 0 0.1	3
8	0.4400	1/16	0.0 0 0 1	4
9	4.3750	45/128	0.1 0 -1 0 -1 0 1	2
10	5.5316	3/8	0.1 0 -1	2
11	1.1428			
12	0.81156			

Example 6

Consider a second choice of WDF configuration for realizing the network of Fig.4.8a), shown in Fig.4.12, in which the parallel adaptor has no reflection-free ports and the remaining adaptors are Brune sections with port two reflection-free. We attempt to apply the two constraints (4.19) to the conductances given in Table 4.7, yielding the set of initial parameters tabulated in Table 4.12. These initially caused an unacceptable deviation in frequency response, but by use of a continuous optimization, a satisfactory nominal parameter set was obtained. The function F_2 will be (4.19), as it was for Example 4, except that the set $\{ \beta \}$ of (3.18) for a port 2 reflection-free Brune adaptor will be substituted for the non-reflection-free adaptor. We apply the following strategy to minimize the hardware requirements: reduce the turns-ratio of adaptor three, followed by its fourth parameter, then the remaining parameters; optimize adaptor four; optimize the remaining adaptors. The parameter set of Table 4.13 was obtained. It just satisfies the specification, as shown in Fig.4.13. A total of 108 529 evaluations of F_1, 2 032 427 frequency points, and 192 evaluations of F_2 was required. Execution time was 2 minutes, 51 seconds.

Adder requirements for this design are as follows: for adaptor one, 7 adders for the flowgraph and 3 for the multipliers were needed; for adaptor four, 6 adders for the flowgraph

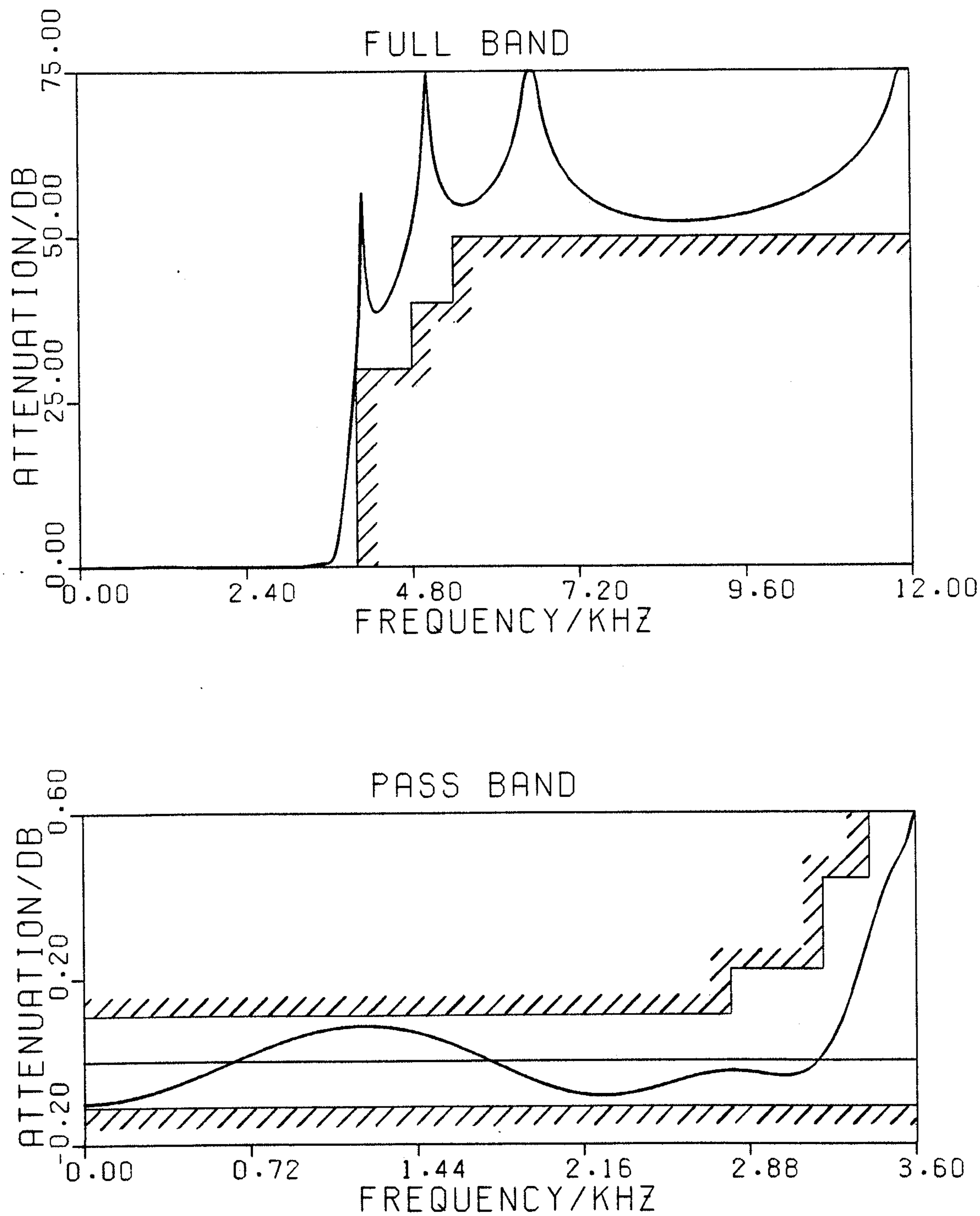


Figure 4.11 Frequency response for Example 5.

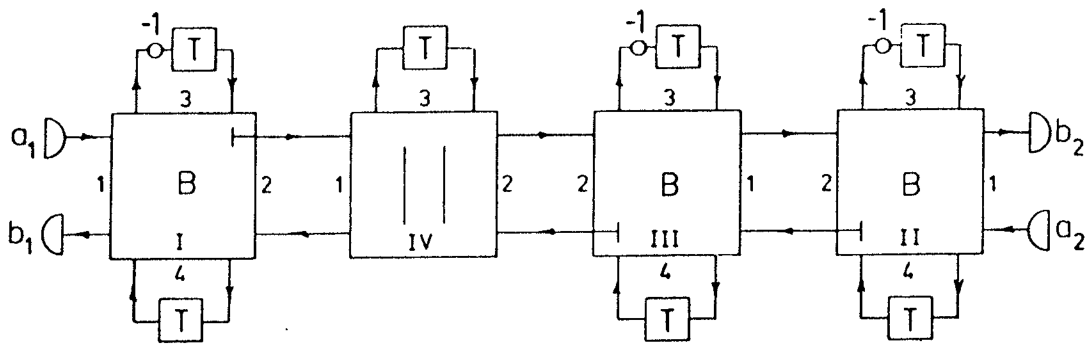


Figure 4.12 Alternate WDF for the network of Fig. 4.10.

Table 4.12 Initial Parameter Set for Example 6.

n	Parameters
1	0.11838
2	0.53495
3	0.065939
4	0.32779
5	0.26753
6	0.26677
7	5.02998
8	0.38926
9	0.30922

and 1 for the multipliers; for adaptor two, 5 adders for the flowgraph and 3 for the multipliers. A wordlength of 15 bits is required for adaptor three's scattering matrix. The adaptor can be implemented via the shift-add method using 34 adders, although the stored-product method would be more practical.

The above results indicate a significant reduction in hardware required for the implementation of cascade WDFs using Brune adaptors and parallel adaptors at a reasonable cost of computation. Results equivalent to the global optima for the 5th-order examples were found. The optimization procedure presented has been designed for cascade realizations including Brune adaptors. Some success has also been achieved for the WD lattice structure, as the following examples will show.

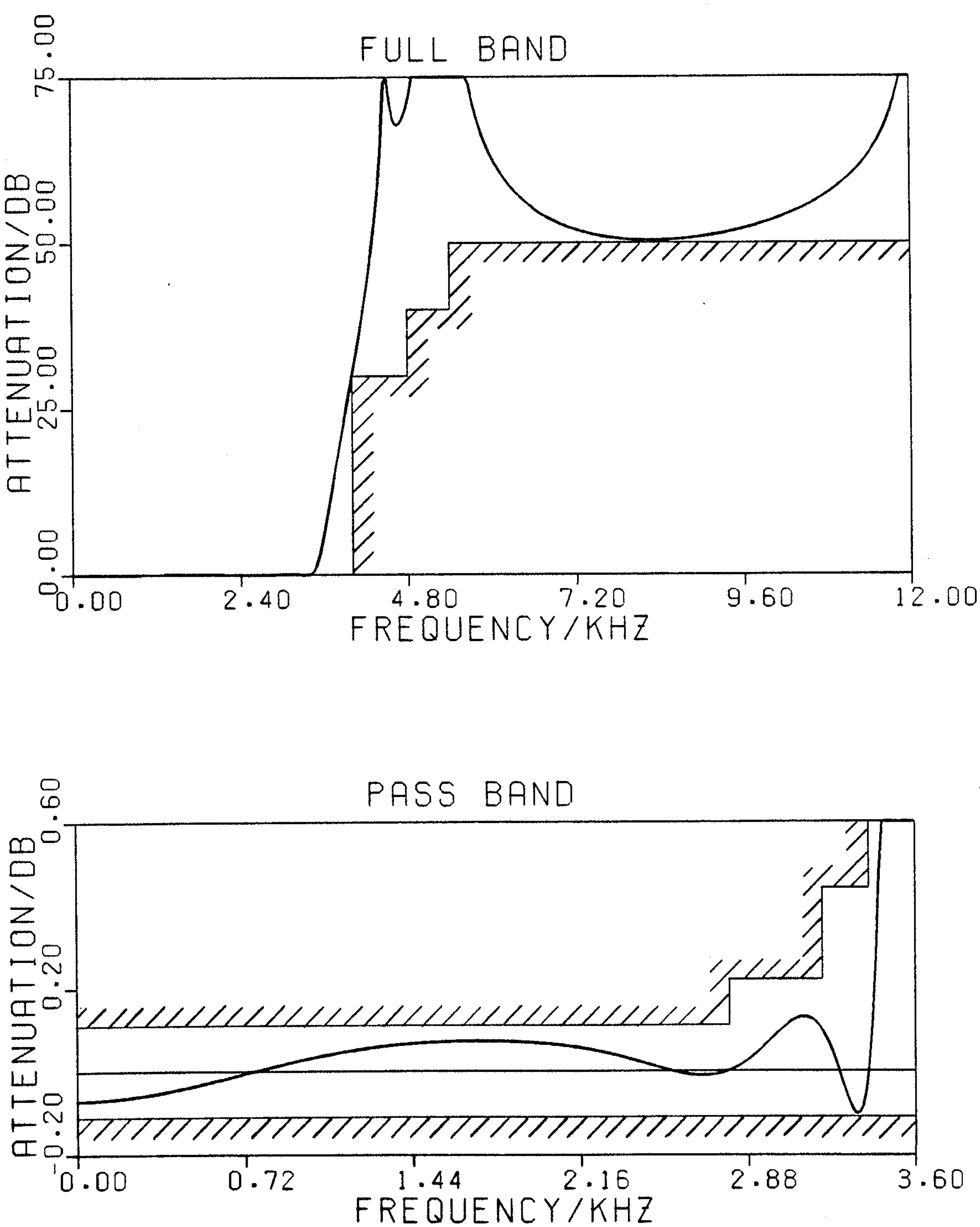


Figure 4.13 Frequency response for Example 6.

Table 4.13 Final Conductances and Parameters for Example 6.

n	Conductances	Parameters		
		rational	CSDC	adaptor
1	0.20000	5/16	0.0 1 0 1	1
2	0.16034	3/8	0.1 0 -1	1
3	0.09091	1/4	0.0 1	3
4	0.083686	59/256	0.0 1 0 0 0 -1 0 -1	3
5	0.062741	6	1 0 -1 0.0	3
6	0.15152	1/16	0.0 0 0 1	4
7	0.18750	3/64	0.0 0 0 1 0 -1	4
8	0.080667	9/32	0.0 1 0 0 1	2
9	0.33333	7/16	0.1 0 0 -1	2
10	0.64583			
11	0.50452			
12	0.20615			

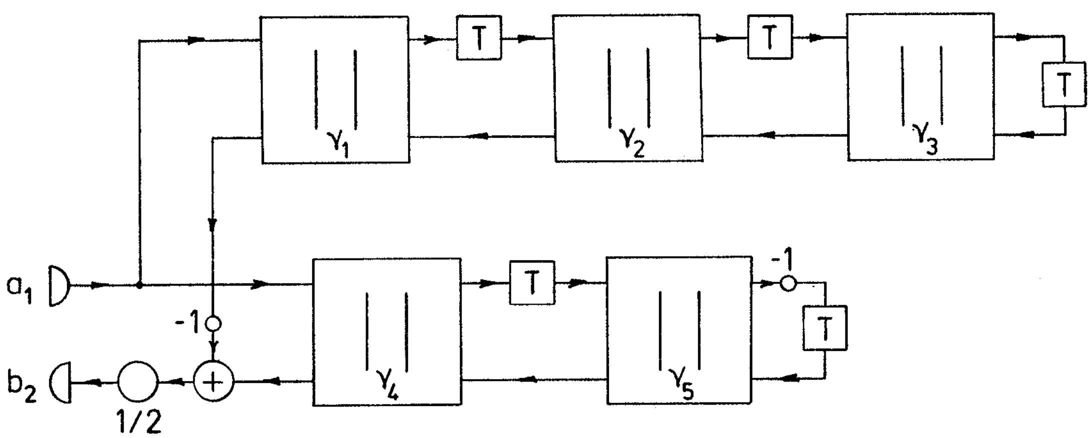


Figure 4.14 Fifth-order lattice WDF for Example 7.

4.5. WD Lattice Examples

Example 7

Consider again the digital filter specifications of Example 3, which may be met with a fifth-order elliptic transfer function. We will realize the filter as a WD lattice structure, as shown in Fig.4.14, which employs chains of unit elements for the first and second arms. A computer program was used to obtain an initial parameter set, given in Table 4.14, which was then optimized to maximize the design margin. For the wordlength objective function we shall use

$$F_2 = \sum_{i=1}^{5} D_i, \tag{4.24}$$

where D_i is the figure of merit for the i^{th} multiplier, given by (4.8). The minimization procedure is then applied, yielding the parameter set given in Table 4.14. A frequency response plot is presented in Fig.4.15. The design is equivalent to the one originally given by Wegener [45]. A total of 6823 evaluations of F_1 using 85 464 frequency points, and 59 evaluations of F_2 were required.

Adder requirements for this design are 16 for the flowgraph and 4 for the multipliers. The latter figure assumes that the appropriate parameter definition for each two-port adaptor will be used. (Proper choice of one of the three possible definitions always allows the multiplier component ± 1 to be eliminated [35].)

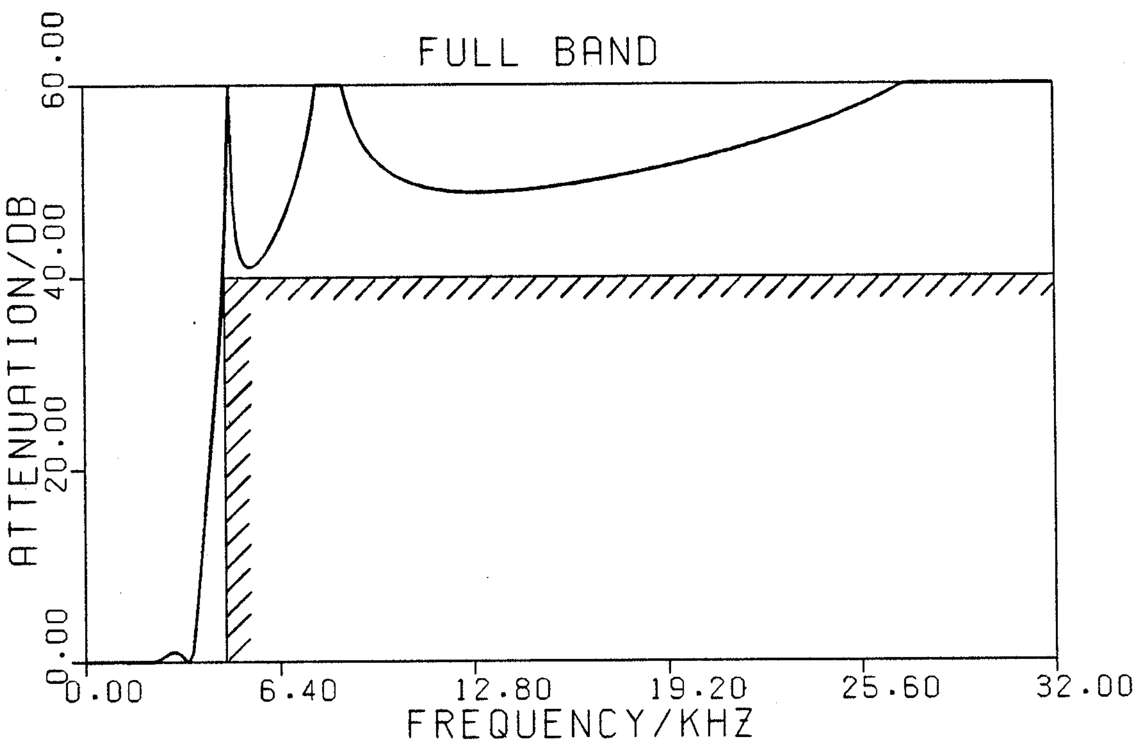
Table 4.14 Initial and Final Parameters for Example 7.

n	Initial	Final	
		rational	CSDC
1	-0.78418	-13/16	-1.0 1 0 -1
2	0.98438	63/64	1.0 0 0 0 0 -1
3	-0.95703	-61/64	-1.0 0 0 1 0 -1
4	-0.79590	-105/128	-1.0 1 0 -1 0 0 -1
5	0.96875	31/32	1.0 0 0 0 -1

Example 8

As the last example, consider the realization of a WD lattice filter of the form shown in Fig.4.17 which satisfies the specifications of § 4.4.2. An initial equiripple design, listed in Table 4.15, was obtained with the aid of tables [87]. It exhibits a sufficient design margin so that a preliminary optimization is not required. By applying the minimization procedure we obtain the design of Table 4.15, whose frequency response is plotted in Fig.4.17. A total of 108 529 evaluations of F_1 using 2 032 427 frequency points, and 192 evaluations of F_2 were required.

Adder requirements for this design are 22 adders for the flowgraph and 9 adders due to the multipliers. These results are comparable in terms of the number of shifts and adders to those presented by Wegener [45].



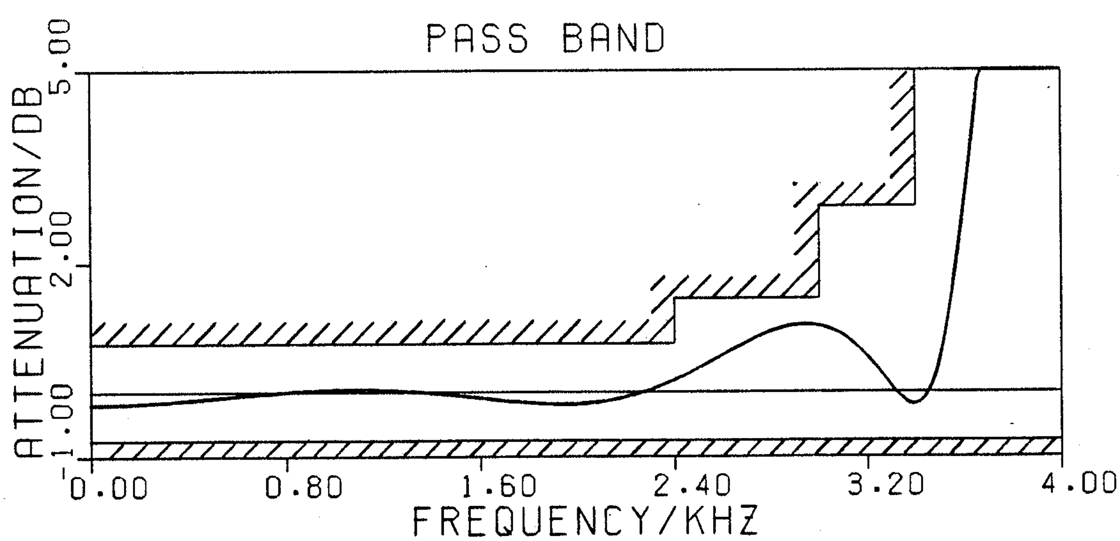


Figure 4.15 Frequency response for Example 7.

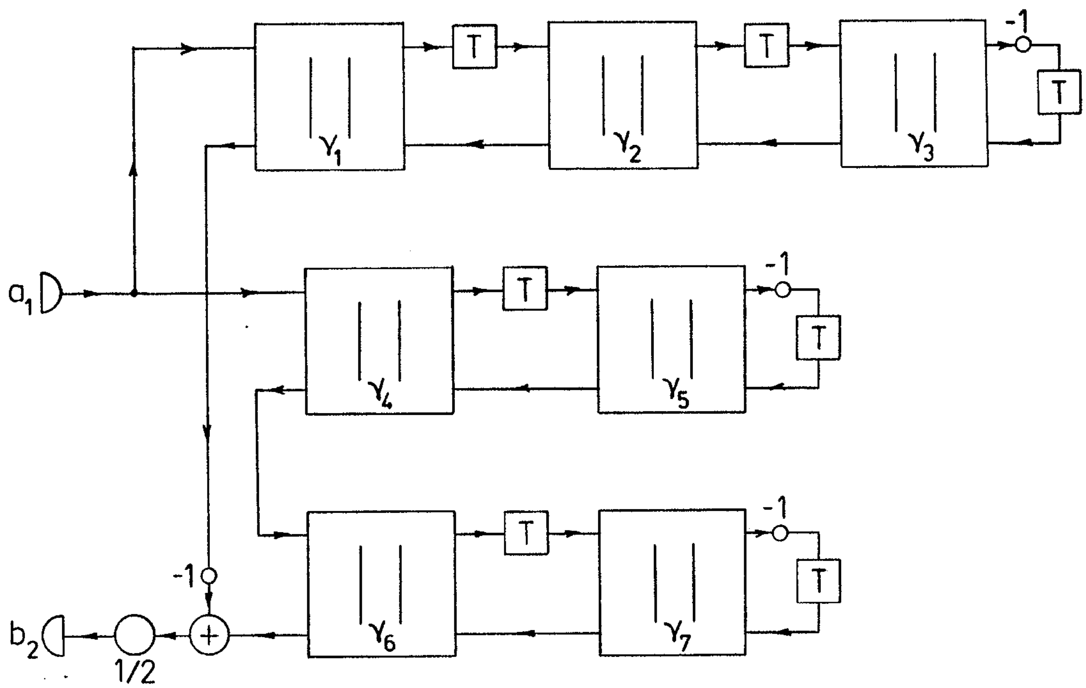


Figure 4.16 Seventh-order lattice WDF for Example 8.

Table 4.15 Initial and Final Parameters for Example 8.

n	Initial	Final	
		rational	CSDC
1	-0.52340	-1/2	0.-1
2	0.82871	27/32	1.0 0 -1 0 -1
3	-0.80580	-3/4	-1.0 1
4	-0.92647	-61/64	-1.0 0 0 1 0 -1
5	0.61713	19/32	0.1 0 1 0 -1
6	-0.57121	-35/64	0.-1 0 0 -1 0 1
7	0.83801	101/128	1.0 -1 0 0 1 0 1

The above two examples illustrate the potential of the minimization procedure for WD lattice filters. They also show the relative complexity of two filters realized using the WD lattice as compared to a cascade of Brune adaptors.

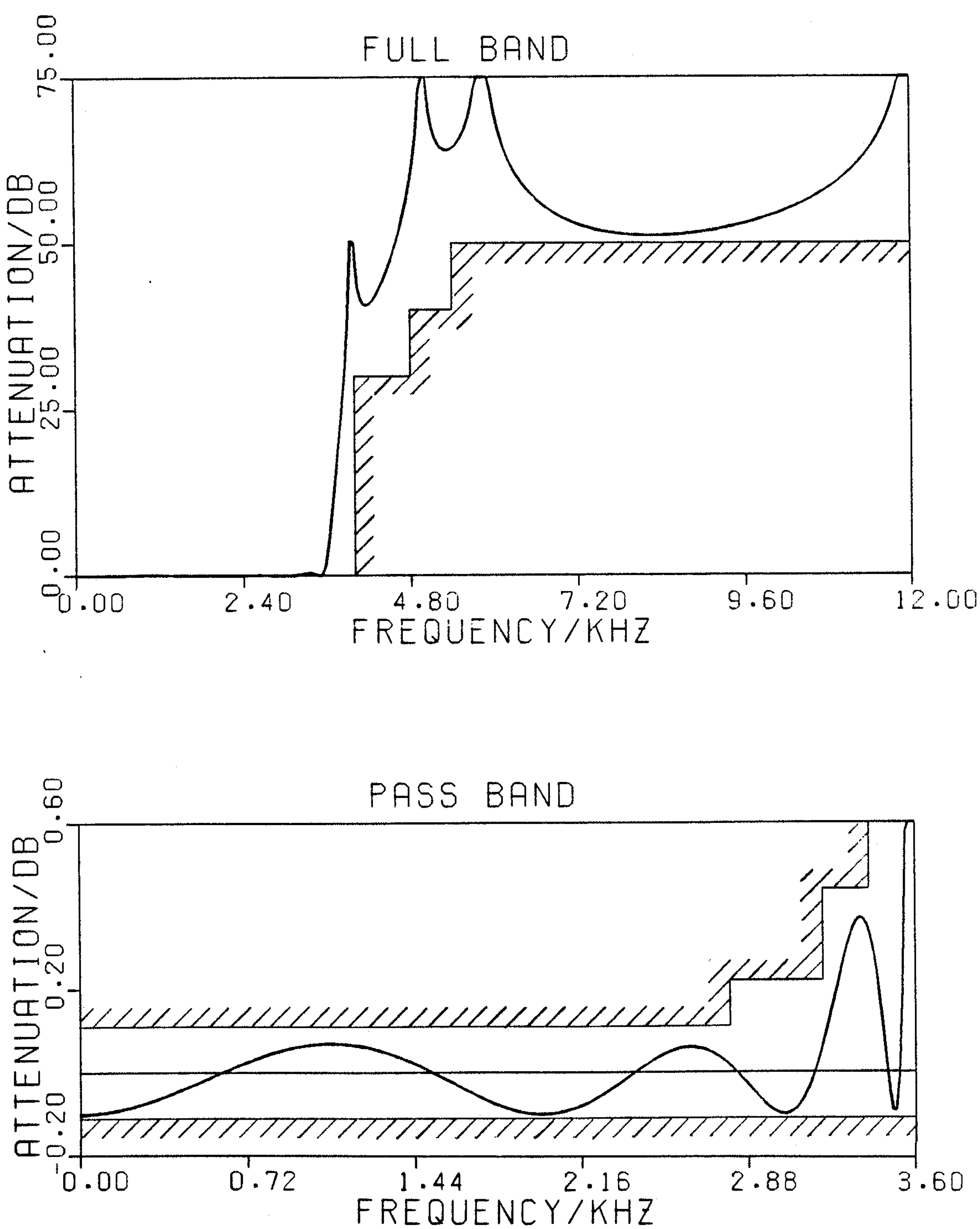


Figure 4.17 Frequency response for Example 8.

5. CONCLUSION

This thesis has proposed an approach to the minimization of WDF realization requirements through discrete optimization. Methods for the design of cascade WDFs involving a recently-introduced Brune adaptor, and for lattice WDFs were presented, both of which allow economical implementations when actual multipliers are replaced by binary shifts and additions. The problem of reducing the total number of operations for a given implementation realized using the shift-and-add method was formulated as an optimization problem. Objective functions were given describing two (relatively unrelated) properties of a digital filter: its ability to satisfy a desired transfer characteristic, and the relative number of shifts and additions required to effect the multiplications. An algorithm was then proposed to efficiently minimize the "hardware" objective function subject to the condition that the transfer characteristic remain within design specifications.

Most WD adaptors have fixed flowgraphs and so changes in the number of shifts and additions depend solely on the design parameter values. The general Brune adaptor, however, does not have a simple fixed flowgraph representation and so we instead considered implementing the adaptor as a matrix multiplication. Since the evaluation of the number of shifts and additions required to implement a matrix multiplication is prohibitively time-consuming, we turned to wordlength reduction of the matrix entries, presuming that a reduction in adder count would ensue. Also, overall wordlength reduction implies reduced costs for an alternate means of implementing a scattering matrix, namely the stored-product method. The problem was simplified by considering the wordlengths of the two matrices, **N** and **K**, of which **S** is composed.

The optimization algorithm was applied to a number of WD filters of fifth and seventh orders, based on ladder and lattice analog prototypes. Designs requiring a minimum number of shifts and additions were found for the fifth-order filters, and were verified using an exhaustive search. The two WD lattice filters given here compare favourably to previously-published results.

We have assumed that the above designs will be implemented using specialized hardware, or in some other way in which the use of distributed arithmetic is uniformly advantageous. Typical target technologies such as microprocessors and VLSI introduce criteria in the minimum-cost design problem that have not been considered here. We suggest that future work might include digital filter cost minimization schemes which take into greater account the restrictions imposed by various specific technologies, such as a particular microprocessor.

Appendix A. Summary of Brune Adaptor Design Equations

A Brune section and its equivalent ladder network are shown in Fig.A.1.

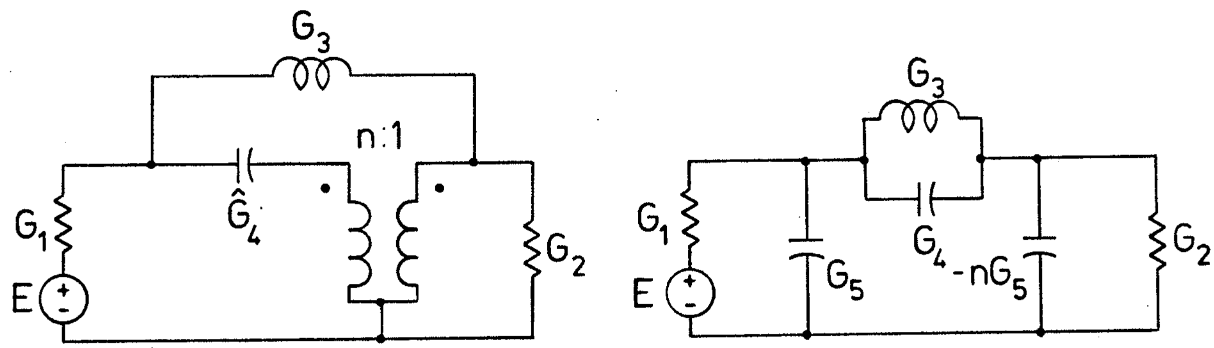


Figure A.1 A Brune section and its equivalent ladder network.

$$n = \frac{G_4}{G_4 + G_5}, \qquad \hat{G}_4 = G_4 + G_5$$

A summary of the various design and analysis equations for the Brune adaptor follow. Simplified designs have the constraint $G_1/G_5 = G_3/G_4$ imposed.

A.1 Non-reflection-free

$$G_T = \frac{1}{\dfrac{1}{G_1 + (1-n)\hat{G}_4} + \dfrac{1}{G_3 + n\hat{G}_4} + \dfrac{1}{G_2 - n(1-n)\hat{G}_4}}$$

definition 1

$$\beta_1 = n \qquad \frac{G_1}{\hat{G}_4} = \frac{1}{\beta_2\beta_4} - (1-n)$$

$$\beta_2 = \frac{G_T}{G_1 + (1-n)\hat{G}_4} \qquad \frac{G_2}{\hat{G}_4} = \frac{1}{\beta_3\beta_4} + n(1-n)$$

$$\beta_3 = \frac{G_T}{G_2 - n(1-n)\hat{G}_4} \qquad \frac{G_3}{\hat{G}_4} = \frac{1}{(1-\beta_2-\beta_3)\beta_4} - n$$

$$\beta_4 = \frac{\hat{G}_4}{G_T}$$

definition 2

$$\beta_1 = n \qquad \frac{G_1}{\hat{G}_4} = \frac{(1-n)\beta_2}{1-\beta_2}$$

$$\beta_2 = \frac{G_1}{G_1 + (1-n)\hat{G}_4} \qquad \frac{G_2}{\hat{G}_4} = n(1-n)\left[1 + \frac{\beta_4}{1 - \beta_4(n(1-\beta_2) + (1-n)(1-\beta_3))}\right]$$

$$\beta_3 = \frac{G_3}{G_3 + n\hat{G}_4} \qquad \frac{G_3}{\hat{G}_4} = \frac{n\beta_3}{1-\beta_3}$$

$$\beta_4 = \frac{G_T}{n(1-n)\hat{G}_4}$$

definition 3

$$\beta_1 = n \qquad \frac{G_1}{\hat{G}_4} = \frac{(1-n)\beta_2}{1-\beta_2}$$

$$\beta_2 = \frac{G_1}{G_1 + (1-n)\hat{G}_4} \qquad \frac{G_2}{\hat{G}_4} = n(1-n)\left[1 + \frac{\beta_3}{n(1-\beta_2)(1-\beta_3(1 + (1-n)\beta_4))}\right]$$

$$\beta_3 = \frac{G_T}{G_1 + (1-n)\hat{G}_4} \qquad \frac{G_3}{\hat{G}_4} = \frac{1}{(1-\beta_2)\beta_4} - n$$

$$\beta_4 = \frac{G_1 + (1-n)\hat{G}_4}{(1-n)(G_3 + n\hat{G}_4)}$$

simplified

$$\beta_1 = n \qquad \frac{G_2}{G_1} = \frac{\beta_1(2-\beta_3)}{\beta_3}$$

$$\beta_2 = \frac{G_1}{G_1 + (1-n)\hat{G}_4} \qquad \frac{G_3}{G_1} = \frac{\beta_1}{1-\beta_1}$$

$$\beta_3 = \frac{nG_1}{nG_1 + G_2} \qquad \frac{\hat{G}_4}{G_2} = \frac{1-\beta_2}{(1-\beta_1)\beta_2}$$

A.2 Port 1 Reflection-Free

$$\beta_1 = n \qquad \frac{\hat{G}_4}{G_1} = \frac{1-2\beta_2}{1-n}$$

$$\beta_2 = \frac{G_1 - (1-n)\hat{G}_4}{2G_1} \qquad \frac{G_2}{G_1} = \frac{2\beta_2}{1-2(1-n)\beta_2\beta_3} + n(1-2\beta_2)$$

$$\beta_3 = \frac{G_1}{(1-n)(G_2 + n\hat{G}_4)} \qquad \frac{G_3}{G_1} = \frac{1}{\beta_3(1-n)} - n\frac{\hat{G}_4}{G_1}$$

A.3 Port 2 Reflection-Free

definition 1

$$\beta_1 = n \qquad \frac{G_1}{G_2} = \frac{1}{n\beta_2} - \frac{(1-n)\hat{G}_4}{G_2}$$

$$\beta_2 = \frac{G_2}{n(G_1 + (1-n)\hat{G}_4)} \qquad \frac{G_3}{G_2} = \frac{1}{\frac{1}{1+2\beta_3(1-n)} - n\beta_2} - 2\beta_3$$

$$\beta_3 = \frac{n\hat{G}_4}{2G_2} \qquad \frac{\hat{G}_4}{G_2} = \frac{2\beta_3}{n}$$

definition 2

$$\frac{G_1}{G_2} = \frac{1}{\frac{1}{1+2\beta 3n} - \beta_2(1-n)} - 2\beta_3$$

$$\beta_1 = n$$

$$\beta_2 = \frac{G_2}{(1-n)(G_3 + n\hat{G}_4)} \qquad \frac{G_3}{G_2} = \frac{1}{\beta_2(1-n)} - \frac{2\beta_3 n}{1-n}$$

$$\beta_3 = \frac{(1-n)\hat{G}_4}{2G_2} \qquad \frac{\hat{G}_4}{G_2} = \frac{2\beta_3}{1-n}$$

simplified

$$\beta_1 = n \qquad \frac{G_1}{G_2} = \frac{1}{\beta_1}$$

$$\beta_2 = \frac{G_1}{G_1 + (1-n)\hat{G}_4} \qquad \frac{G_3}{G_2} = \frac{1}{1-\beta_1}$$

$$\frac{\hat{G}_4}{G_2} = \frac{1-\beta_2}{(1-\beta_1)\beta_1\beta_2}$$

Appendix B. Program Listing of Optimization Algorithm

A listing of the optimization program, input requirements and some sample data are given below.

```
C**********************************************************************
C                                                                     *
C S/R DPATTERN PERFORMS DISCRETE PATTERN SEARCH TO MINIMIZE WORDLENGTH *
C REQUIREMENTS WHILE SATISFYING FREQUENCY SPECIFICATIONS               *
C VERSION: JUN 06'85,   STARTED: SEPT 7, 1984                          *
C DESCRIPTION OF INPUT DATA:                                           *
C   PLOTTING:                                                          *
C     WC1  -      LOWEST FREQUENCY TO BE PLOTTED                       *
C     WC   -      HIGHEST FREQUENCY TO BE PLOTTED                      *
C     WS   -      STOPBAND EDGE                                        *
C     RW   -      PASSBAND RIPPLE                                      *
C     ASMIN -     MINIMUM ATTENUATION IN STOPBAND                      *
C   HI-RES F1 ROUTINE:                                                 *
C     NX   -      # OF FREQUENCY SAMPLE POINTS                         *
C     NPX  -      # OF PASSBAND RANGES                                 *
C     NSX         # OF STOPBAND RANGES                                 *
C     R,BTOL -    LIST OF FREQUENCY RANGES AND TOLERANCES              *
C   MAIN F1 ROUTINE:                                                   *
C     NPASS -     # OF PASSBAND POINTS                                 *
C     NSTOP -     # OF STOPBAND POINTS                                 *
C     F,TOL -     LIST OF FREQUENCY POINTS AND TOLERANCES              *
C   OPTIMIZATION STAGES:                                               *
C     NSTG   -    # OF STAGES                                          *
C     NINC,VINC,                                                       *
C      SFIX -     LIST OF # OF PARAMETERS INCLUDED IN A STAGE,         *
C                 INDICES OF THE PARMS. INCLUDED, AND                  *
C                 A LIST OF ND 1'S OR 0'S INDICATING WHICH PARMS. ARE  *
C                 CURRENTLY FIXED (1) OR FREE (0)                      *
C   CONTINUOUS OPTIMIZATION PARAMETERS:                                *
C     IDC    -    DEBUG PARAMETER (SEE DBUG)                           *
C     M0C    -    MAX. # OF VARIABLES (SEE M0)                         *
C     NOPTC  -    # OF CONTINUOUS OPTIMIZATION ATTEMPTS (SEE NOPT)     *
C     CNTRC  -    CONTRACTION FACTOR; SHOULD BE POWER OF 2             *
C     EXITC  -    LOG BASE 2 OF EXIT CRITERION                         *
C   MISCELLANEOUS :                                                    *
C     F1EXIT -    MAXIMUM ALLOWABLE VALUE OF F1                        *
C     M0     -    MAXIMUM # OF VARIABLES VARIED AT ONCE IN DPATTS      *
C     QMAX   -    MAXIMUM QUANTIZATION WORDLENGTH                      *
C     QRED   -    INDICATES WHETHER INITIAL BINARY FRACTIONS ARE TO BE *
C                 REDUCED (1) OR NOT (0)                               *
C     DBUG   -    A DEBUGGING PARAMETER FOR DPATTS, IN THE RANGE [-1,4] *
C                 CAUSING NO OUTPUT (-1) UP TO FULL OUTPUT (4)         *
C     ICOPT  -    INDICATES WHETHER CONTINUOUS OPTIMIZATION IS TO BE   *
C                 INCLUDED IN DPATTS (1) OR NOT (0)                    *
C     NRPT   -    # OF TIMES THE MAIN ROUTINE IS TO BE REPEATED,       *
C                 TO ALLOW DIFFERENT STARTING POINTS TO BE USED        *
C     NOPT   -    # OF TIMES TO REPEAT DPATTS AT EACH STAGE            *
C     NQOPT  -    # OF ATTEMPTS AT CONTINUOUS OPTIMIZATION IF          *
C                 INITIAL QUANTIZATION FAILS                           *
C     ISTOP  -    1 IF THE PROGRAM SHOULD BE STOPPED AFTER             *
C                 QUANTIZATION, ELSE 0; USED IN PRELIM. CHECKS OF DATA *
C                                                                     *
C**********************************************************************
      LOGICAL OPTOK, FAST
      INTEGER Q(20), ITITLE(144), QMAX, QM, OUT, DBUG, DIM, TITLE(80),
     &DIMMAX, ORD(20), PQ(20), P2M0, P21, FIX(20), QRED, INC(20), VINC(
     &20), SINC(9, 20), SFIX(9, 20), DO(20), QO(20)
      REAL GR(20), D(20), DX(20), DIN(20), DLB(20), DUB(20), TOL(100),
     &WPTS(100), AWPTS(100), C(128, 20), D0(20), R(2, 30), BTOL(30)
      REAL*8 DSEED, GRD(20)
      COMMON / FUNCS / ND, D, Q, NPASS, NSTOP, TOL, WPTS, AWPTS, F1EXIT,
     &DLB, DUB, ORD, FIX, IP, PI, FAST
      COMMON / FCN2 / INC, ICOPT
      COMMON / IOUT / IN, OUT, DBUG
      COMMON / PLT / ITITLE
      COMMON / PLT2 / WC, ASMIN, RW, WC1
      COMMON / SMFY / PQ
      COMMON / SRCH / C, M0, QMAX, F1EXT
      COMMON / DSRC / M0D
      COMMON / CONTIN / IDC, EXITC, NOPTC, CNTRC, M0C
      PI = 4. * ATAN(1.)
      PI2 = PI / 2.
      PI256 = PI / 256.
      IN = 5
      OUT = 6
      CALL UGETIO(1, NIN, NOUT)
      READ (IN,400) (TITLE(I), I = 1, 80)
```

```
      READ (NIN,410) (ITITLE(I), I = 1, 144)
      READ (IN,*) WC1, WC, WS, RW, ASMIN
      READ (IN,*) ND, DEN, (D(I), I = 1, ND)
      READ (IN,*) NX, NPX, NSX
      NTX = NPX + NSX
      DO 10 I = 1, NTX
10    READ (IN,*) R(1, I), R(2, I), BTOL(I)
      READ (IN,*) NPASS, NSTOP
      NSPEC = NPASS + NSTOP
      READ (IN,*) (WPTS(I), TOL(I), I = 1, NSPEC)
      READ (IN,*) NSTG
      IF (NSTG .EQ. 0) GO TO 50
      DO 40 J = 1, NSTG
      READ (IN,*) NINC, (VINC(I), I = 1, NINC), (SFIX(J, I), I = 1, ND)
      DO 20 I = 1, ND
20    SINC(J, I) = 0
      DO 30 I = 1, NINC
30    SINC(J, VINC(I)) = 1
40    CONTINUE
50    READ (IN,*) FIEXIT, M0, QMAX, QRED
      READ (IN,*) DBUG, ICOPT, NRPT, NOPT, NQOPT
      READ (IN,*) IDC, M0C, NOPTC, CNTRC, EXITC, ISTOP
C-----------------------------------------------------------------------
      M0D = M0
      DO 60 I = 1, ND
60    D(I) = D(I) / DEN
      WRITE (OUT,440) (TITLE(I), I = 1, 80), ND, (D(I), I = 1, ND)
      WRITE (OUT,450) WC1, WC, WS, RW, ASMIN, NPASS, NSTOP, DBUG, NRPT,
     &FIEXIT, M0, ICOPT
      WRITE (OUT,460) IDC, EXITC, NOPT, CNTRC, M0C, ISTOP, QMAX, NOPT,
     &NQOPT
      WRITE (OUT,470) NSTG
      DO 70 J = 1, NSTG
      WRITE (OUT,480) J, (SINC(J, I), I = 1, ND)
      WRITE (OUT,490) (SFIX(J, I), I = 1, ND)
70    CONTINUE
C-----------------------------------------------------------------------
C GENERATE SEARCH PATTERN MATRIX
C-----------------------------------------------------------------------
      DSEED = D(1)
      FIEXT = FIEXIT
      MX = 7
      P2M0 = 2 ** MX
      P21 = 1
      DO 90 I = 1, MX
      DO 80 J = 1, P2M0
80    C(J, I) = 1 - 2 * MOD((J - 1) / P21, 2)
      P21 = P21 * 2
90    CONTINUE
      DO 100 I = 1, NSPEC
100   AWPTS(I) = TAN(WPTS(I) * PI256)
C-----------------------------------------------------------------------
C PREPARE INITIAL PARAMETER SET. PERTURB INITIAL SET ON REPEAT.
C-----------------------------------------------------------------------
      DO 120 I = 1, ND
      D0(I) = D(I)
120   CONTINUE
      DO 380 IRPT = 1, NRPT
      WRITE (OUT,500) IRPT
      IF (IRPT .LE. 1) GO TO 140
      DO 130 I = 1, ND
      D(I) = D0(I) * ((GGUBFS(DSEED) - 0.5) * 0.4 + 1)
130   CONTINUE
      WRITE (OUT,510) (D(I), I = 1, ND)
140   DO 150 I = 1, ND
      FIX(I) = 0
150   ORD(I) = I
      NFE = 0
      NFPTS = 0
      FAST = .FALSE.
      FI = 0
      IP = 1
      CALL FUNCT(FI, D, ND, NFE, NFPTS)
      IP = 0
      CALL XD2G(ND, D, NG, GR, IER)
      CALL PLOT2(NG, GR, 1)
      CALL FUNCX(R, BTOL, NX, NPX, NSX, FI)
      WRITE (6,420) (GR(I), I = 1, NG)
      WRITE (6,430) (D(I), I = 1, ND)
      IF (FI .LE. FIEXIT) GO TO 180
C-----------------------------------------------------------------------
C IF SPEC NOT MET INITIALLY, OPTIMIZE
C-----------------------------------------------------------------------
      IQOPT = 0
160   WRITE (OUT,520) IQOPT
      SIDLEN = 1. / 2 ** 4
      DO 170 I = 1, ND
170   DX(I) = SIDLEN
```

```
      NFE1 = 0
      CALL OPT(IDC, DX, SIDLEN, EXITC, NOPTC, CNTRC, OPTOK, F1, NFE1,
     &NSRCH, IOPT)
      WRITE (OUT,530) F1, (D(I), I = 1, ND)
      CALL XD2G(ND, D, NG, GR, IER)
      WRITE (6,420) (GR(I), I = 1, NG)
      CALL PLOT2(NG, GR, 1)
      IF (F1 .GT. F1EXIT .OR. .NOT. OPTOK) GO TO 350
C----------------------------------------------------------------------
C IF SPEC HAS BEEN MET, CONTINUE
C----------------------------------------------------------------------
180   IF (ISTOP .EQ. 1) GO TO 390
C----------------------------------------------------------------------
C QUANTIZE PARAMETERS TO BINARY NUMBERS
C----------------------------------------------------------------------
      WRITE (OUT,540)
      MAXBIT = QMAX
      DO 190 I = 1, ND
      ORD(I) = 1
190   DIN(I) = D(I)
      IF (QRED .EQ. 1) GO TO 200
      QM = MAXBIT
      GO TO 220
200   QM = 0
      DO 210 I = 1, ND
      Q(I) = INT(ALOG(1 / ABS(D(I))) / ALOG(2.0) + 0.5)
210   QM = AMAX0(Q(I), QM)
220   DO 230 I = 1, ND
      Q(I) = QM
230   D(I) = AINT(DIN(I) * 2 ** Q(I) + SIGN(0.5, DIN(I))) / 2 ** Q(I)
      FAST = .TRUE.
      F1 = F1EXIT
      CALL FUNCT(F1, D, ND, NFE, NFPTS)
      WRITE (OUT,550) F1, (D(I), I = 1, ND)
      WRITE (OUT,560) (Q(I), I = 1, ND)
      IF (F1 .LE. F1EXIT) GO TO 240
      QM = QM + 1
      IF (QM .LE. MAXBIT) GO TO 220
      IQOPT = IQOPT + 1
      IF (IQOPT .LE. NQOPT) GO TO 160
      GO TO 360
C----------------------------------------------------------------------
C PARAMETERS SUCCESSFULLY QUANTIZED, REDUCE BINARY FRACTIONS IF QRED=1
C----------------------------------------------------------------------
240   CONTINUE
      IF (QRED .EQ. 0) GO TO 280
      DO 250 I = 1, ND
250   PQ(I) = 2 ** Q(I)
      DO 270 I = 1, ND
      DT = D(I) * PQ(I)
      IF (AINT(DT) .NE. DT) GO TO 270
260   DT = DT / 2
      IF (DT .NE. AINT(DT)) GO TO 270
      Q(I) = Q(I) - 1
      GO TO 260
270   CONTINUE
280   CONTINUE
C----------------------------------------------------------------------
C PREPARE FOR DISCRETE MINIMIZATION
C----------------------------------------------------------------------
      WRITE (OUT,550) F1, (D(I), I = 1, ND)
      WRITE (OUT,560) (Q(I), I = 1, ND)
      DO 290 I = 1, ND
      INC(I) = 1
      ORD(I) = 1
290   CONTINUE
      WRITE (OUT,570)
      IP = 1
      CALL FUNC2(F2, Q, D, ND, NFE)
      IP = 0
      NFE1 = 0
      NFE2 = 0
      NFP1 = 0
      NFE1C = 0
C----------------------------------------------------------------------
C LOOP FOR STAGES
C----------------------------------------------------------------------
      DO 320 ISTG = 1, NSTG
      DO 300 I = 1, ND
      INC(I) = SINC(ISTG, I)
      FIX(I) = SFIX(ISTG, I)
300   CONTINUE
      WRITE (OUT,600) ISTG, (INC(I), I = 1, ND)
      WRITE (OUT,610) (FIX(I), I = 1, ND)
C----------------------------------------------------------------------
C PERFORM DISCRETE OPTIMIZATION
C----------------------------------------------------------------------
      CALL DOPT(DBUG, NOPT, OPTOK, F1, F2, NFEVAL, NFPTS, NFEV2, NFEVC,
```

```
     &DO, QO)
      NFE1 = NFE1 + NFEVAL
      NFE2 = NFE2 + NFEV2
      NFP1 = NFP1 + NFPTS
      NFE1C = NFE1C + NFEVC
      DO 310 I = 1, ND
310   ORD(I) = I
      F1 = 1000
      CALL FUNCT(F1, D, ND, NFE, NFPTS)
320   CONTINUE
C----------------------------------------------------------------------
C AFTER STAGES, PRINT STATISTICS AND PLOT FREQ. RESPONSE
C----------------------------------------------------------------------
      IP = 1
      FAST = .FALSE.
      DO 330 I = 1, ND
330   ORD(I) = I
      DO 340 I = 1, ND
340   INC(I) = 1
      WRITE (OUT,620)
      CALL FUNC2(F2, Q, D, ND, NFE)
      CALL FUNCX(R, BTOL, NX, NPX, NSX, F1)
      CALL FUNCT(F1, D, ND, NFE, NFPTS)
      CALL XD2G(ND, D, NG, GR, IER)
      WRITE (OUT,650) (GR(I), I = 1, NG)
      NF1T = NFE1 + NFE1C
      WRITE (OUT,660) NFE1, NFE1C, NF1T, NFP1, NFE2
      CALL PLOT2(NG, GR, 1)
      IP = 0
      GO TO 380
C----------------------------------------------------------------------
C ERROR MESSAGES
C----------------------------------------------------------------------
350   WRITE (OUT,680)
      GO TO 370
360   WRITE (OUT,690)
370   IP = 1
      FAST = .FALSE.
      CALL FUNCT(F1, D, ND, NFE, NFPTS)
380   CONTINUE
C----------------------------------------------------------------------
C DONE
C----------------------------------------------------------------------
390   WRITE (OUT,670)
      STOP
C----------------------------------------------------------------------
400   FORMAT (80A1)
410   FORMAT (72A1)
420   FORMAT ('0DPM: GR=' / (' ', 10G13.5))
430   FORMAT ('0D=' / (' ', 10G13.5))
440   FORMAT ('1' / ' ', 72('-') / ' DIGITAL FILTER WORDLENGTH MINIMIZAT
     &ION' / ' ', 72('-') / '0TITLE: ', 80A1 / '0', 'NUMBER OF PARAMETER
     &S:', I3 / '0INPUT PARAMETERS:' / '0', 10F12.5)
450   FORMAT ('0', 'WC1:', F12.5, T20, 'WC:', F12.5, T40, 'WS:', F12.5,
     &T60, 'RW:', F12.5, T80, 'ASMIN:', F12.5, T100, 'NPASS:', I4, T120,
     &'NSTOP:', I4 / '0DBUG:', I3, T20, 'NRPT:', I2, T40, 'F1EXIT:',
     &G13.5, T60, 'M0:', I4, T80, 'ICOPT:', I4)
460   FORMAT ('0IDC:', I5, T20, 'EXITC:', F12.5, T40, 'NOPT:', I5, T60,
     &'CNTRC:', F12.5, T80, 'MOC:', I5, T100, 'ISTOP:', I5, T120, 'QMAX:
     &', I5, / '0NOPT:', I3, T10, 'NQOPT:', I3)
470   FORMAT ('0NSTG:', I4)
480   FORMAT ('0STAGE ', I3, T15, 'INCLUDED:', T30, 20I3)
490   FORMAT ('0', T15, 'FIXED:', T30, 20I3)
500   FORMAT ('0', 72('-') / '0TRIAL # ', I2)
510   FORMAT ('0GENERATED STARTING POINT:', ( / '0', 10G13.5))
520   FORMAT ('0PRELIMINARY LINEAR OPT. REQUIRED, #', I3)
530   FORMAT ('0CONT. OPT. PARAMETER SET: F1=', G13.5 / ('0', 10G13.5))
540   FORMAT ('0QUANTIZATION STEP')
550   FORMAT ('0QUANTIZED PARAMETER SET:  F1=', G13.5, ( / '0', 10G13.5)
     &)
560   FORMAT ('0QUANTIZATION WORDLENGTH:', ( / '0', 10(4X, I2, 7X)))
570   FORMAT ('0INITIAL F2:')
580   FORMAT ('0UPPER STABILITY BOUND:', ( / '0', 10G13.5))
590   FORMAT ('0LOWER STABILITY BOUND:', ( / '0', 10G13.5))
600   FORMAT ('0STAGE ', I2 / '0INCLUDED: ', T15, 20I3)
610   FORMAT ('0FIXED: ', T15, 20I3)
620   FORMAT ('0>> FINAL VALUES')
630   FORMAT ('0G-RATIOS, INCLUDING TERMINATIONS:' / ('0', 5G25.16))
640   FORMAT ('0 ', 10I12 / ('0', 10I12))
650   FORMAT ('0G-RATIOS:' / ('0', 10G13.5))
660   FORMAT ('0# F1 EVAL DISCRETE', I10, ' # F1 EVAL CONT.', I10, ' # F
     &1 EVAL TOTAL:', I10, ' # FREQ.PTS.USED:', I10, ' # F2 EVAL:', I10)
670   FORMAT ('1')
680   FORMAT ('0ABORTED: INITIAL F1 IS GREATER THAN F1EXIT')
690   FORMAT ('0ABORTED: CANNOT QUANTIZE TO MEET SPEC')
      END
C**********************************************************************
C                                                                     *
```

```
C S/R BITS CALCULATES THE CSDC REPRESENTATION OF N AND THE NUMBER        *
C OF NON-ZERO BITS REQUIRED.              STARTED 85 APR 02              *
C                                                                        *
C*************************************************************************
      SUBROUTINE BITS(NIN, Q, BITCNT, K)
      IMPLICIT INTEGER(A - Z)
      INTEGER K(1)
      N = NIN
      BITCNT = 1
      K(1) = MOD(N, 2)
      N = N / 2
      Q1 = Q + 1
      DO 10 I = 2, Q1
      K(I) = MOD(N, 2)
      N = N / 2
      IF (K(I) .EQ. 0) GO TO 10
      BITCNT = BITCNT + 1
      IF (K(I - 1) .EQ. 0) GO TO 10
      K(I - 1) = - K(I - 1)
      N = N + K(I)
      K(I) = 0
      BITCNT = BITCNT - 1
10    CONTINUE
      RETURN
      END
C*************************************************************************
C                                                                        *
C S/R COPT PERFORMS A CONTINUOUS OPTIMIZATION TO REDUCE F1 AFTER         *
C A DSRCH STAGE.     FEB 21'85,   SEPT20'84                              *
C                                                                        *
C*************************************************************************
      SUBROUTINE COPT(F, X, N, Q, NFE, NFPTS, NFE1)
      LOGICAL FAST
      INTEGER SFIX(20), SORD(20), Q(20)
      REAL X(1), DO(20), DXO(20), SD(20), DX(20)
      COMMON / FUNCS / ND, D(20), IQ(20), NPASS, NSTOP, TOL(100), WPTS(
     &100), AWPTS(100), F1EXIT, DLB(20), DUB(20), IORD(20), IFIX(20), IP
     &, P1, FAST
      COMMON / FCN2 / INC(20), ICOPT
      COMMON / IOUT / IN, IO, ID1
      COMMON / CONTIN / ID, EXIT, NOPT, CNTRCT, M0C
      IF (ICOPT .NE. 1) RETURN
      FO = F
      DO 10 I = 1, ND
      DX(I) = PWR(IQ(I))
10    SD(I) = D(I)
      DO 20 I = 1, N
      DX(IORD(I)) = PWR(Q(I))
      D(IORD(I)) = X(I)
      CALL STABND(ND, D, IORD(I), DLB, DUB)
      IF (D(IORD(I)) .LE. DLB(IORD(I))) GO TO 120
      IF (D(IORD(I)) .GE. DUB(IORD(I))) GO TO 120
20    CONTINUE
      DO 30 I = 1, ND
      SFIX(I) = IFIX(I)
      SORD(I) = IORD(I)
      IF (INC(I) .EQ. 1) IFIX(I) = 1
30    CONTINUE
      NDIM = ND
      NFIX = 0
      I = 1
40    IF (IFIX(IORD(I)) .EQ. 0) GO TO 50
      NFIX = NFIX + 1
      IT = IORD(I)
      IORD(I) = IORD(NDIM)
      IORD(NDIM) = IT
      NDIM = NDIM - 1
      GO TO 60
50    I = I + 1
60    IF (I .LE. NDIM) GO TO 40
      STEP = 4.
      F = 1000.
      DO 70 I = 1, ND
      DO(I) = D(IORD(I))
      DXO(I) = DX(IORD(I)) * STEP
70    CONTINUE
      DXMIN = 0.
      IF (NDIM .LE. 0) GO TO 90
      DO 80 I = 1, NDIM
      DXMIN = AMAX1(DXMIN, DX(IORD(I)))
80    CONTINUE
90    CALL PATTRN(NDIM, DO, DXO, DXMIN, CNTRCT, F, NFE1, NSRCH, NFPTS1)
      NFE = NFE + NFE1
      NFPTS = NFPTS + NFPTS1
      DO 100 I = 1, ND
      D(IORD(I)) = DO(I)
      IFIX(I) = SFIX(I)
      IORD(I) = SORD(I)
```

```
100     CONTINUE
        IF (F .GT. F1EXIT) GO TO 130
        DO 110 I = 1, ND
        X(I) = D(IORD(I))
110     CONTINUE
        RETURN
120     F = 100
130     DO 140 I = 1, ND
140     D(I) = SD(I)
        RETURN
        END
C*****************************************************************
C                                                                *
C S/R DOPT PERFORMS A DISCRETE CONSTRAINED OPTIMIZATION. AUG 30,1984 *
C VERSION: 85 FEB 21                                             *
C                                                                *
C*****************************************************************
        SUBROUTINE DOPT(DBUG, NOPT, OPTOK, F1, F2, NFEVAL, NFPTS, NFE2V,
       &NCFEV, DN, QO)
        LOGICAL OPTOK, FAST
        INTEGER Q(20), QO(20), ORD(20), FIX(20), SORD(20, 20), DBUG, QBEST
       &(20), OINC(20), QSAVE(20), DN(1)
        REAL D(20), DX(20), DO(20), DLB(20), DUB(20), S(20), TOL(100),
       &WPTS(100), AWPTS(100), G(20), DBEST(20), DSAVE(20)
        REAL*8 DSEED
        COMMON / FUNCS / ND, D, Q, NPASS, NSTOP, TOL, WPTS, AWPTS, F1EXIT,
       &DLB, DUB, ORD, FIX, IP, P1, FAST
        COMMON / FCN2 / INC(20), ICOPT
        ROUND(X) = AINT(X + SIGN(0.5, X))
        DSEED = D(1)
        OPTOK = .TRUE.
        IQM = 20
        NFEVAL = 0
        NFPTS = 0
        NCFEV = 0
        NFE2V = 0
        NFE = 0
        DO 10 I = 1, ND
10      DX(I) = PWR(IQM)
        CALL RELSEN(DX, NFE, NFPTS, S)
        NFEVAL = NFEVAL + NFE
        WRITE (6,170) NFEVAL, NFPTS
        DO 20 I = 1, ND
        DSAVE(I) = D(I)
        QSAVE(I) = Q(I)
20      SORD(I, 1) = ORD(I)
        IF (NOPT .LT. 2) GO TO 50
        DO 40 I = 2, NOPT
        CALL GGPER(DSEED, ND, ORD)
        DO 30 J = 1, ND
30      SORD(I, J) = ORD(J)
40      CONTINUE
50      FBEST = 1.E50
        DO 150 IOPT = 1, NOPT
        DO 60 I = 1, ND
        D(I) = DSAVE(I)
        Q(I) = QSAVE(I)
        ORD(I) = SORD(IOPT, I)
60      CONTINUE
        NDIM = ND
        NFIX = 0
        I = 1
70      IF (FIX(ORD(I)) .EQ. 0 .AND. INC(ORD(I)) .EQ. 1) GO TO 80
        NFIX = NFIX + 1
        IT = ORD(I)
        ORD(I) = ORD(NDIM)
        ORD(NDIM) = IT
        NDIM = NDIM - 1
        GO TO 90
80      I = I + 1
90      IF (I .LE. NDIM) GO TO 70
        DO 100 I = 1, ND
        DO(I) = D(ORD(I))
        QO(I) = Q(ORD(I))
        OINC(I) = INC(ORD(I))
        IF (OINC(I) .EQ. 1) CALL FNDWL(DO(I), IQM, QO(I))
        DX(I) = PWR(QO(I))
100     CONTINUE
        WRITE (6,210) (ORD(I), I = 1, ND)
        F1 = F1EXIT
        CALL DPATTS(NDIM, DO, QO, DX, F1, F2, OINC, NFE, NSRCH, NFP, NF2,
       &NCFE)
        NFEVAL = NFEVAL + NFE
        NCFEV = NCFEV + NCFE
        NFPTS = NFPTS + NFP
        NFE2V = NFE2V + NF2
        DO 110 I = 1, NDIM
        D(ORD(I)) = DO(I)
```

```
      Q(ORD(I)) = QO(I)
110   CONTINUE
      IF (F1 .GT. F1EXIT .OR. F2 .GT. FBEST) GO TO 130
      FBEST = F2
      DO 120 I = 1, ND
      DBEST(I) = D(I)
      QBEST(I) = Q(I)
120   CONTINUE
130   DO 140 I = 1, ND
      QO(I) = MAX0(1, 2 ** Q(I))
      DO(I) = D(I) * QO(I)
      DN(I) = DO(I)
140   CONTINUE
      WRITE (6,180) IOPT, F1, F2, (DO(I), I = 1, ND)
      WRITE (6,190) (QO(I), I = 1, ND)
      WRITE (6,200) (D(I), I = 1, ND)
150   CONTINUE
      DO 160 I = 1, ND
      D(I) = DBEST(I)
      Q(I) = QBEST(I)
160   CONTINUE
      RETURN
170   FORMAT ('0DOPT: NFEVAL AFTER RELSEN=', G13.6, ' NFPTS=', G13.6)
180   FORMAT ('0OPTIMIZATION ', I2 / '0F1= ', G13.5 / '0F2= ', G13.5 / '
     &0PARAMETER  SET:' / ('0', 20F8.0))
190   FORMAT ('0', 20I8)
200   FORMAT ('0', 10G13.5)
210   FORMAT (' DOPT: ORD=', 20I4)
      END
C**************************************************************************
C                                                                         *
C S/R DPATTS PERFORMS A DISCRETE PATTERN SEARCH TO MINIMIZE A WORD-       *
C LENGTH-BASED FUNCTION.                                                  *
C VERSION 85 MAR 15,  STARTED 84 OCT 12                                   *
C                                                                         *
C**************************************************************************
      SUBROUTINE DPATTS(N, X, Q, E, F1, F, INC, NFE, NSRCH, NFPTS, NFE2,
     &NCFE)
      LOGICAL IMPROV
      INTEGER Q(1), QE(20), QMAX, INC(20)
      REAL XE(20), X(1), DX(1), E(20), C(128, 20), EE(20)
      COMMON / IOUT / IN, IO, ID
      COMMON / SRCH / C, IM0, QMAX, F1EXIT
      COMMON / DSRC / M0
      NFE = 0
      NSRCH = 0
      NFE2 = 0
      NFPTS = 0
      NCFE = 0
      CALL FUNCT(F1, X, N, NFE, NFPTS)
      CALL FUNC2(F, Q, X, N, NFE2)
      NFE = 0
      NFE2 = 0
      NFPTS = 0
      MAXRPT = 5
      NRPT = 0
      M01 = M0
      M0 = MIN0(M0, N)
      IF (ID .LT. 0) GO TO 10
      WRITE (IO,90) N, ID, M0, F1, F
      WRITE (IO,100) (X(I), I = 1, N)
      WRITE (IO,130) (Q(I), I = 1, N)
10    FE = F
      DO 20 I = 1, N
      XE(I) = X(I)
      QE(I) = Q(I)
      EE(I) = E(I)
20    CONTINUE
      CALL DSRCH(F1, FE, XE, QE, EE, N, INC, NFE, NFE2, NSRCH, NCFE,
     &NFPTS, IMPROV)
      IF (ID .LT. 2) GO TO 30
      WRITE (IO,120) F, F1, FE, NSRCH, NFE, NFE2, NCFE, NFPTS, (X(I), I
     & = 1, N)
      WRITE (IO,150) (Q(I), I = 1, N)
      WRITE (IO,140) (XE(I), I = 1, N)
      WRITE (IO,160) (QE(I), I = 1, N)
      WRITE (IO,170) (EE(I), I = 1, N)
30    IF ( .NOT. IMPROV) GO TO 70
40    CONTINUE
      IF (ID .GE. 2) WRITE (IO,110)
      DO 50 I = 1, N
      XV = X(I)
      X(I) = XE(I)
      Q(I) = QE(I)
      E(I) = EE(I)
      XE(I) = XE(I) + SIGN0(EE(I), XE(I) - XV)
      IF (INC(I) .EQ. 1) CALL FNDWL(XE(I), QMAX, QE(I))
      EE(I) = PWR(QE(I))
```

```
50     CONTINUE
       F = FE
       CALL DSRCH(F1, FE, XE, QE, EE, N, INC, NFE, NFE2, NSRCH, NCFE,
      &NFPTS, IMPROV)
       IF (ID .LT. 2) GO TO 60
       WRITE (IO,120) F, F1, FE, NSRCH, NFE, NFE2, NCFE, NFPTS, (X(I), I
      & = 1, N)
       WRITE (IO,150) (Q(I), I = 1, N)
       WRITE (IO,140) (XE(I), I = 1, N)
       WRITE (IO,160) (QE(I), I = 1, N)
       WRITE (IO,170) (EE(I), I = 1, N)
60     IF (IMPROV) GO TO 40
       IF (ID .GE. 2) WRITE (IO,180)
       GO TO 10
70     CONTINUE
       IF (ID .LT. 0) GO TO 80
       NFE1 = 0
       F1 = 1000
       CALL FUNCT(F1, X, N, NFE1, NFE1)
       WRITE (IO,190) F, F1, NFE, NSRCH, NFE2, NCFE, NFPTS, (X(I), I = 1,
      &N)
       WRITE (IO,130) (Q(I), I = 1, N)
       M0 = M01
80     RETURN
90     FORMAT ('1' / / / 1X, 70('-') / ' DISCRETE PATTERN SEARCH FOR ', '
      &NONLINEAR MINIMIZATION' / 1X, 70('-') / / / 5X, 'INPUT DATA:' / 5X
      &, 11('-') / / 5X, 'N      =', I3 / 5X, 'ID     =', I3 / 5X, 'M0     =
      &', I3 / 5X, 'F1     =', G13.5 / 5X, 'F2     =', G13.5)
100    FORMAT ('0INPUT VECTOR:' / ('0', 10G13.5))
110    FORMAT ('0EXTRAPOLATION:')
120    FORMAT (' F=', G13.5, ' F1=', G13.5, ' FE=', G13.5, ' NSRCH=', I4,
      &' NFE=', I4, ' NFE2=', I4, ' NCFE=', I11, ' NFPTS=', I11 / ' X=',
      &10G13.5)
130    FORMAT ('0WORDLENGTHS:' / ('0', 10I13))
140    FORMAT (' XE', 10G13.5)
150    FORMAT (' Q=', 10I13)
160    FORMAT (' QE', 10I13)
170    FORMAT (' E=', 10G13.5)
180    FORMAT ('0RETREAT:')
190    FORMAT ('-', 4X, 'FINAL OBJECTIVE FUNCTION VALUE:', G13.5 / '0',
      &8X, 'FINAL SPEC. FUNCTION VALUE:', G13.5 / '0', 4X, 'NUMBER OF FUN
      &CTION EVALUATIONS:', I6 / '0', 11X, 'NUMBER OF SEARCH STAGES:', I6
      & / '0', 12X, 'NUMBER OF WL FCN EVALS:', I6 / '0', 10X, 'NUMBER OF
      &CONTIN.F.EVALS:', I6 / '0', 11X, 'NUMBER OF FREQ PTS USED:', I13 /
      &'0FINAL POINT:' / ('0', 10G13.5))
       END
C*********************************************************************
C                                                                    *
C S/R DSRCH PERFORMS MULTIVARIATE PATTERN SEARCH PROBE MOVEMENT.      *
C VERSION: 85 FEB 21                                                 *
C                                                                    *
C*********************************************************************
       SUBROUTINE DSRCH(F1, F0, X, Q, D, N, INC, NFE1, NFE2, NSR, NCFE,
      &NFPTS, IMPROV)
       LOGICAL IMPROV
       INTEGER Q(1), P2M, QS(20), QB(20), QMAX, INC(20)
       REAL X(1), D(1), DS(20), C(128, 20), XS(20), XB(20)
       COMMON / IOUT / IN, IO, ID
       COMMON / SRCH / C, IM0, QMAX, F1EXIT
       COMMON / DSRC / M0
       IMPROV = .FALSE.
       NCFE0 = 0
       NSR = NSR + 1
       FE = F0
       F = F0
       F1B = F1
       NFEAS = 0
       IF (ID .LT. 3) GO TO 10
       WRITE (IO,170) F0, F1B, (Q(I), I = 1, N)
       WRITE (IO,190) (X(I), I = 1, N)
10     F1 = 1000
       CALL FUNCT(F1, X, N, NFE1, NFPTS)
       CALL FUNC2(FE, Q, X, N, NFE2)
       IF (FE .GT. F) GO TO 40
       IF (FE .EQ. F .AND. F1 .GE. F1B) GO TO 40
       IF (F1 .LE. F1EXIT) GO TO 20
       CALL COPT(F1, X, N, Q, NCFE, NFPTS, NCFE0)
       IF (F1 .GT. F1EXIT) GO TO 40
20     NFEAS = NFEAS + 1
       DO 30 I = 1, N
       XB(I) = X(I)
       QB(I) = Q(I)
30     CONTINUE
       F = FE
       F1B = F1
40     IF (ID .LT. 4) GO TO 50
       I = 0
       WRITE (IO,180) I, I, I, FE, F, F1, NCFE0, (Q(I0), I0 = 1, N)
```

```
      IF (ID .GE. 4) WRITE (IO,190) (X(IO), IO = 1, N)
50    DO 150 M = 1, M0
      P2M = 2 ** M
      DO 140 K = 1, N
      DO 120 J = 1, P2M
      NCFE0 = 0
      DO 60 I = 1, M
      L = MOD(I + K - 2, N) + 1
      DS(I) = D(L)
      XS(I) = X(L)
      QS(I) = Q(L)
      D(L) = D(L) * C(J, I)
      X(L) = X(L) + D(L)
      IF (INC(L) .EQ. 1) CALL FNDWL(X(L), QMAX, Q(L))
60    CONTINUE
      F1 = 1000
      CALL FUNCT(F1, X, N, NFE1, NFPTS)
      CALL FUNC2(FE, Q, X, N, NFE2)
      IF (FE .GT. F) GO TO 90
      IF (FE .EQ. F .AND. F1 .GE. F1B) GO TO 90
      IF (F1 .LE. F1EXIT) GO TO 70
      CALL COPT(F1, X, N, Q, NCFE, NFPTS, NCFE0)
      IF (F1 .GT. F1EXIT) GO TO 90
70    NFEAS = NFEAS + 1
      DO 80 I = 1, N
      XB(I) = X(I)
      QB(I) = Q(I)
80    CONTINUE
      F = FE
      F1B = F1
90    IF (ID .LT. 4) GO TO 100
      WRITE (IO,180) J, K, M, FE, F, F1, NCFE0, (Q(IO), IO = 1, N)
      IF (ID .GE. 4) WRITE (IO,190) (X(IO), IO = 1, N)
100   DO 110 I = 1, M
      L = MOD(I + K - 2, N) + 1
      X(L) = XS(I)
      D(L) = DS(I)
      Q(L) = QS(I)
110   CONTINUE
120   CONTINUE
      IF (NFEAS .LT. 1) GO TO 140
      DO 130 I = 1, N
      X(I) = XB(I)
      Q(I) = QB(I)
      D(I) = PWR(QB(I))
130   CONTINUE
      IMPROV = .TRUE.
      NFEAS = 0
140   CONTINUE
      IF (F .LT. F0) GO TO 160
150   CONTINUE
      F0 = F
      F1 = F1B
      RETURN
160   F1 = F1B
      F0 = F
      RETURN
170   FORMAT ('0DSRCH; F0:', G13.5, 'F1B:', G13.5, ' Q:', 20I3)
180   FORMAT (' J:', I2, ' K:', I2, ' M:', I2, ' FE:', G13.5, ' F:',
     &G13.5, ' F1:', G13.5, ' NCFE0:', G13.5, ' Q:', 20I3)
190   FORMAT (' X:', 10G13.5)
      END
C*****************************************************************
C                                                                *
C S/R FNDWL FINDS THE SHORTEST WORDLENGTH REQUIRED TO REPRESENT X1.  *
C VERSION MAR  26, 1985,  STARTED OCT 12, 1984                    *
C                                                                *
C*****************************************************************
      SUBROUTINE FNDWL(X1, L0, L1)
      L1 = L0
      X = X1 / PWR(L0)
10    X = X / 2.
      IF (X .NE. AINT(X)) GO TO 20
      L1 = L1 - 1
      IF (X .NE. 0.) GO TO 10
      L1 = 0
20    RETURN
      END
C*****************************************************************
C                                                                *
C S/R FUNCT EVALUATES THE OBJECTIVE FUNCTION BASED ON FREQUENCY  *
C VERSION FEB 21'85,  OCT 25'84                                  *
C                                                                *
C*****************************************************************
      SUBROUTINE FUNCT(F, DO, N, NFE, NFPTS)
      LOGICAL FAST
      INTEGER ORD(20), Q(20), FIX(20)
      REAL DO(1), FR(100)
```

```
      COMMON / FUNCS / ND, D(20), Q, NPASS, NSTOP, TOL(100), WPTS(100),
     &AWPTS(100), FIEXIT, DLB(20), DUB(20), ORD, FIX, IP, PI, FAST
      COMMON / FREQPT / NG, G(20), W, FRI
      NFE = NFE + 1
      IF (N .LE. 0) GO TO 20
      DO 10 I = 1, N
10    D(ORD(I)) = DO(I)
20    DO 30 I = 1, ND
      CALL STABND(ND, D, I, DLB, DUB)
      IF (D(I) .GE. DUB(I)) GO TO 120
      IF (D(I) .LE. DLB(I)) GO TO 120
30    CONTINUE
      CALL XD2G(ND, D, NG, G, IER)
      IF (IER .EQ. 1) GO TO 130
      FCOMP = F
      FRMAX = 0
      FP = 0
      FRMIN = 0
      DO 40 I = 1, NPASS
      W = AWPTS(I)
      NFPTS = NFPTS + 1
      CALL FRQPT
      FRI = - 20.0 * ALOG10(FRI)
      FR(I) = FRI
      FRMIN = AMIN1(FRMIN, FRI)
      FRMAX = AMAX1(FRMAX, FRI)
      FP = AMAX1(FP, (FRI - FRMIN) / TOL(I), ABS(FRMIN / TOL(NPASS)))
C     FP=AMAX1(FP,(FRI-FRMIN)/TOL(I))
      IF (FP .GT. FCOMP .AND. FAST) GO TO 100
40    CONTINUE
      IF (FRMIN .GE. 0) GO TO 60
      DO 50 I = 1, NPASS
      FP = AMAX1(FP, (FR(I) - FRMIN) / TOL(I))
      IF (FP .GT. FCOMP .AND. FAST) GO TO 100
50    CONTINUE
60    ATTMIN = 1000
      NS1 = NPASS + 1
      NS2 = NPASS + NSTOP
      FS = 0.
      DO 70 I = NS1, NS2
      W = AWPTS(I)
      NFPTS = NFPTS + 1
      CALL FRQPT
      FRI = - 20.0 * ALOG10(FRI)
      FR(I) = FRI
      ATTMIN = AMIN1(FRI, ATTMIN)
      DFS = FRI - FRMIN
      IF (DFS .EQ. 0.) DFS = 1.E - 10
      FS = AMAX1(FS, ABS(TOL(I) / DFS))
      IF (FS .GT. FCOMP .AND. FAST) GO TO 110
70    CONTINUE
      F = AMAX1(FP, FS)
      IF (IP .NE. 1) GO TO 90
      WRITE (6,140) FRMIN, FRMAX, ATTMIN, F
      WRITE (6,150)
      DO 80 I = 1, NS2
      RELATT = FR(I) - FRMIN
      WRITE (6,160) I, WPTS(I), FR(I), RELATT, TOL(I)
80    CONTINUE
90    RETURN
100   F = FP
      RETURN
110   F = FS
      RETURN
120   F = 500.
      RETURN
130   F = 501.
      RETURN
140   FORMAT ('0MIN.ATTEN.:', G13.5, 'MAX.PASS.:', G13.5, 'MIN.STOP.:',
     &G13.5, 'OBJ.FCN.:', G13.5)
150   FORMAT ('-', 4X, 'POINT', 4X, 'FREQ', 10X, 'ATTEN(ABS)', 9X, 'ATTE
     &N(REL)', 7X, 'TOL')
160   FORMAT (' ', 6X, I2, 1X, 5(2X, G13.5, 2X))
      END
C****************************************************************
C                                                               *
C S/R FUNCX EVALUATES THE FREQUENCY RESPONSE OBJECTIVE FUNCTION WITH  *
C HIGH RESOLUTION. OCT. 10'84                                   *
C                                                               *
C****************************************************************
      SUBROUTINE FUNCX(R, BTOL, N, NP, NS, F)
      LOGICAL FAST
      INTEGER ORD(20), Q(20), FIX(20)
      REAL R(2, 30), BTOL(30), AT(300)
      COMMON / FUNCS / ND, D(20), Q, NPASS, NSTOP, TOL(100), WPTS(100),
     &AWPTS(100), FIEXIT, DLB(20), DUB(20), ORD, FIX, IP, PI, FAST
      COMMON / FREQPT / NG, G(20), W, FRI
      CALL XD2G(ND, D, NG, G, IER)
```

```
      FPMIN = 1000.
      DO 10 I = 1, N
      WD = (I - 1.) / N
      W = TAN(WD * PI / 2.)
      CALL FRQPT
      FRI = - 20. * ALOG10(FRI)
      AT(I) = FRI
      IF (FRI .GT. FPMIN) GO TO 10
      WPMIN = WD * 128.
      FPMIN = FRI
10    CONTINUE
      DO 20 I = 1, NP
      W = TAN(R(2, I) * PI / 256.)
      CALL FRQPT
      FRI = - 20. * ALOG10(FRI)
      AT(I + N) = FRI
      IF (FRI .GT. FPMIN) GO TO 20
      WPMIN = R(2, I)
      FPMIN = FRI
20    CONTINUE
      DO 30 I = 1, NS
      W = TAN(R(1, I + NP) * PI / 256.)
      CALL FRQPT
      FRI = - 20. * ALOG10(FRI)
      AT(I + N + NP) = FRI
      IF (FRI .GT. FPMIN) GO TO 30
      WPMIN = R(1, I + NP)
      FPMIN = FRI
30    CONTINUE
      FP = 0.
      DO 50 I = 1, NP
      L1 = INT(R(1, I) * N / 128.) + 2
      L2 = INT(R(2, I) * N / 128.) + 1
      FPMAX = - 1000.
      DO 40 J = L1, L2
      IF (AT(J) .LE. FPMAX) GO TO 40
      FPMAX = AT(J)
      WPMAX = (J - 1) * 128. / N
40    CONTINUE
      FP = AMAX1(FP, (FPMAX - FPMIN) / BTOL(I))
      TP = (AT(N + I) - FPMIN) / BTOL(I)
      IF (TP .LE. FP) GO TO 50
      FP = TP
      WPMAX = R(2, I)
      FPMAX = AT(N + I)
50    CONTINUE
      FS = 0.
      I1 = NP + 1
      I2 = NP + NS
      DO 70 I = I1, I2
      L1 = INT(R(1, I) * N / 128.) + 2
      L2 = INT(R(2, I) * N / 128.) + 1
      FSMIN = 1000
      DO 60 J = L1, L2
      IF (AT(J) .GE. FSMIN) GO TO 60
      FSMIN = AT(J)
      WSMIN = (J - 1) * 128. / N
60    CONTINUE
      FS = AMAX1(FS, BTOL(I) / ABS(FSMIN - FPMIN))
      TS = BTOL(I) / ABS(AT(N + I) - FPMIN)
      IF (TS .LE. FS) GO TO 70
      FS = TS
      WSMIN = R(1, I)
      FSMIN = AT(N + I)
70    CONTINUE
      F = AMAX1(FP, FS)
      WRITE (6,80) N, F, FP, WPMAX, FPMAX, FS, WSMIN, FSMIN, FPMIN,
     &WPMIN
      RETURN
80    FORMAT ('-OBJ.FCN(', I4, ' POINTS)=', G13.5 / '0F IN PB.:', G13.5,
     &T30, 'AT:', G13.5, T50, 'VALUE:', G13.5 / '0F IN SB.:', G13.5, T30
     &, 'AT:', G13.5, T50, 'VALUE:', G13.5 / '0ATT.MIN.:', G13.5, T30, '
     &AT:', G13.5)
      END
C***********************************************************************
C                                                                      *
C S/R OPT PERFORMS A CONTINUOUS CONSTRAINED OPTIMIZATION.              *
C VERSION: APR 02'85 ,  STARTED JULY 19'84                             *
C                                                                      *
C***********************************************************************
      SUBROUTINE OPT(DBUG, DX, SIDLEN, EXIT, NOPT, CNTRCT, OPTOK, F,
     &NFEVAL, NSRCH, IOPT)
      LOGICAL OPTOK, FAST
      INTEGER Q(20), ORD(20), FIX(20), SORD(20, 20), DBUG, ORD1(20)
      REAL D(20), DO(20), DLB(20), DUB(20), S(20), DXO(20), TOL(100),
     &WPTS(100), AWPTS(100), G(20), FS(20), DX(20), DBEST(20)
      REAL*8 DSEED
      COMMON / FUNCS / ND, D, Q, NPASS, NSTOP, TOL, WPTS, AWPTS, FIEXIT,
```

```
     &DLB, DUB, ORD, FIX, IP, P1, FAST
      DSEED = ABS(D(1))
      NFE = 0
      NFPTS = 0
      FBEST = 1000
      DO 10 I = 1, ND
      DBEST(I) = D(I)
      IF (FIX(I) .EQ. 0) GO TO 10
      CALL STABND(ND, D, I, DLB, DUB)
      IF (D(I) .GE. DUB(I)) GO TO 160
      IF (D(I) .LE. DLB(I)) GO TO 160
10    CONTINUE
      CALL RELSEN(DX, NFE, NFPTS, S)
      DO 20 I = 1, ND
      SORD(1, I) = ORD(I)
      ORD(I) = I
20    CONTINUE
      IF (NOPT .LT. 2) GO TO 50
      DO 40 I = 2, NOPT
      CALL GGPER(DSEED, ND, ORD1)
      DO 30 J = 1, ND
30    SORD(I, J) = ORD1(J)
40    CONTINUE
50    NFEVAL = 0
      DXMIN = 1. / 2 ** EXIT
      OPTOK = .TRUE.
      IOPT = 1
60    CONTINUE
      DO 70 I = 1, ND
70    ORD(I) = SORD(I, I)
      NDIM = ND
      NFIX = 0
      I = 1
80    IF (FIX(ORD(I)) .EQ. 0) GO TO 90
      NFIX = NFIX + 1
      IT = ORD(I)
      ORD(I) = ORD(NDIM)
      ORD(NDIM) = IT
      NDIM = NDIM - 1
      GO TO 100
90    I = I + 1
100   IF (I .LE. NDIM) GO TO 80
      DO 110 I = 1, ND
      DO(I) = DBEST(ORD(I))
      DXO(I) = DX(ORD(I))
110   CONTINUE
      FAST = .TRUE.
      CALL PATTRN(NDIM, DO, DXO, DXMIN, CNTRCT, F, NFE, NSRCH, NFPTS)
      NFEVAL = NFEVAL + NFE
      FS(IOPT) = F
      DO 120 I = 1, ND
120   D(ORD(I)) = DO(I)
      WRITE (6,190) IOPT, F, (D(I), I = 1, ND)
      IF (F .GE. FBEST) GO TO 140
      FBEST = F
      DO 130 I = 1, ND
      DBEST(ORD(I)) = DO(I)
130   CONTINUE
140   IOPT = IOPT + 1
      IF (FBEST .GT. F1EXIT .AND. IOPT .LE. NOPT) GO TO 60
      F = FBEST
      DO 150 I = 1, ND
150   D(I) = DBEST(I)
      RETURN
160   OPTOK = .FALSE.
      RETURN
170   FORMAT (' ORD:', 10G13.5)
180   FORMAT (' FIX:', 10G13.5 / (' ', 10G13.5))
190   FORMAT ('0OPTIMIZATION ', I2 / '0F= ', G13.5 / '0PARAMETER SET:' /
     &('0', 10G13.5))
200   FORMAT ('0OPT1; ORD:' / ('0', 10G13.5))
210   FORMAT ('0OPT1; DO:' / ('0', 10G13.5))
220   FORMAT ('0OPT2; DO:' / ('0', 10G13.5))
      END
C*****************************************************************
C                                                                *
C S/R PATTRN PERFORMS MULTIVARIATE CONTINUOUS PATTERN SEARCH      *
C VERSION 85 FEB 21,  OCT 5'84                                   *
C                                                                *
C*****************************************************************
      SUBROUTINE PATTRN(N, X, DX, DXMIN, CNTRCT, F, NFE, NSRCH, NFPTS)
      REAL XE(20), X(1), DX(1)
      COMMON / IOUT / IN, IO, ID1
      COMMON / SRCH / C(128, 20), MOD, IQMAX, F1EXIT
      COMMON / CONTIN / ID, EXIT, NOPT, CNTRC1, M0
      NFE = 0
      NSRCH = 0
      NFPTS = 0
```

```
      M01 = M0
      M0 = MIN0(M0, N)
      IF (ID .LT. 0) GO TO 10
      CALL FUNCT(F, X, N, NFE, NFPTS)
      WRITE (IO,150) N, ID, DXMIN, M0, CNTRCT, F
      IF (N .LE. 0) GO TO 10
      WRITE (IO,160) (X(I), I = 1, N)
      WRITE (IO,170) (DX(I), I = 1, N)
10    F = 1000
      CALL FUNCT(F, X, N, NFE, NFPTS)
      IF (N .LE. 0) GO TO 140
      FE = F
      DO 20 I = 1, N
20    XE(I) = X(I)
      CALL SEARCH(FE, XE, DX, N, NFE, NSRCH, NFPTS)
      IF (ID .LT. 3) GO TO 30
      WRITE (IO,180) F, FE, NSRCH, NFE, (X(I), I = 1, N)
      WRITE (IO,200) (XE(I), I = 1, N)
30    IF (FE .GE. F) GO TO 80
40    CONTINUE
      DO 50 I = 1, N
      XV = X(I)
      X(I) = XE(I)
      XE(I) = 2. * XE(I) - XV
50    CONTINUE
      F = FE
      IF (ID .LT. 3) GO TO 60
      WRITE (IO,210)
      WRITE (IO,200) (XE(I), I = 1, N)
60    CALL FUNCT(FE, XE, N, NFE, NFPTS)
      CALL SEARCH(FE, XE, DX, N, NFE, NSRCH, NFPTS)
      IF (ID .LT. 2) GO TO 70
      WRITE (IO,190) F, FE, NSRCH, NFE, (X(I), I = 1, N)
      WRITE (IO,200) (XE(I), I = 1, N)
70    IF (NFE .GT. 2000) GO TO 100
      IF (FE .LT. F) GO TO 40
      GO TO 10
80    CONTINUE
      DO 90 I = 1, N
90    IF (ABS(DX(I)) .GT. DXMIN) GO TO 120
100   IF (ID .LT. 0) GO TO 110
      WRITE (IO,220) F, NFE, NSRCH, (X(I), I = 1, N)
      WRITE (IO,170) (DX(I), I = 1, N)
110   M0 = M01
      RETURN
120   DO 130 I = 1, N
130   DX(I) = DX(I) / CNTRCT
      IF (ID .GE. 2) WRITE (IO,230) (DX(I), I = 1, N)
      GO TO 10
140   IF (ID .GE. 0) WRITE (IO,240) F
      M0 = M01
      RETURN
150   FORMAT ('1' / / / 1X, 70('-') / ' MULTIVARIATE PATTERN SEARCH FOR
     &', 'NONLINEAR MINIMIZATION' / 1X, 70('-') / / / 5X, 'INPUT DATA:'
     & / 5X, 11('-') / / 5X, 'N      =', I3 / 5X, 'ID     =', I3 / 5X, 'DX
     &MIN =', F8.4 / 5X, 'M MAX =', I3 / 5X, 'CNTRCT=', F8.4 / 5X, 'F IN
     &IT=', F8.4)
160   FORMAT ('0INPUT VECTOR:' / ('0', 10G13.5))
170   FORMAT ('0DX VECTOR:' / ('0', 10G13.5))
180   FORMAT ('0RETREAT:' / ' F=', G13.5, T20, 'FE=', G13.5, T40, 'NSRCH
     &=', I4, T60, 'NFE=', I4 / ' X=', 10G13.5 / ' '10G13.5)
190   FORMAT ('0EXTRAPOLATION:' / ' F=', G13.5, T20, 'FE=', G13.5, T40,
     &'NSRCH=', I4, T60, 'NFE=', I4 / ' X=', 9G13.5 / ' ', 10G13.5)
200   FORMAT (' XE=', 9G13.5 / ' ', 10G13.5)
210   FORMAT (' AFTER EXTRAPOLATION:')
220   FORMAT ('-', 4X, 'FINAL OBJECTIVE FUNCTION VALUE:', G13.5 / '0',
     &4X, 'NUMBER OF FUNCTION EVALUATIONS:', I6 / '0', 11X, 'NUMBER OF S
     &EARCH STAGES:', I6 / '0FINAL POINT:' / ('0', 10G13.5))
230   FORMAT ('0DX REDUCED TO:', 9G13.5 / ('0', 10G13.5))
240   FORMAT ('-', 4X, 'OBJECTIVE FUNCTION VALUE:', G13.5 / '0', 4X, 'NO
     & FREE PARAMETERS')
      END
C*****************************************************************
C                                                                *
C S/R PLOT PLOTS THE FREQUENCY RESPONSE OF THE FILTER.           *
C OCT   3'84      IT=0:  PLOT ONLY                               *
C                 IT=1:  PLOT AND TABLE                          *
C                 IT=2:  TABLE ONLY                              *
C                                                                *
C*****************************************************************
      SUBROUTINE PLOT(RANGE, IT)
      INTEGER IMAG4(5151), ITITLE(144), ICHAR(10), Q(1)
      REAL ATTEN(1024, 1), FREQ(1024), RANGE(4)
      COMMON / PLT / ITITLE
      COMMON / FREQPT / NG, G(20), W, FRI
      DATA ICHAR(1) / 1H  /
      IY = 1024
      N = 128
```

```
      M = 1
      INC = 1
      IOPT = 1
      PI = 4 * ATAN(1.)
      ATTMIN = 1.E50
      DO 10 I = 1, N
      W = (I - 1) * RANGE(2) / N
      FREQ(I) = W
      W = TAN(W * PI / 2. / N)
      CALL FRQPT
      ATTEN(I, 1) = - 20. * ALOG10(FR1)
      IF (ATTMIN .LT. ATTEN(I, 1)) GO TO 10
      ATTMIN = ATTEN(I, 1)
      MATT = I
10    CONTINUE
      IF (IT .LE. 0) GO TO 30
      WRITE (6,70)
      N4 = N / 4
      DO 20 I = 1, N4
      WRITE (6,80) (FREQ((J - 1) * N4 + I), ATTEN((J - 1) * N4 + I, 1),
     &J = 1, 4)
20    CONTINUE
30    DO 40 I = 1, N
40    ATTEN(I, 1) = AMIN1(ATTEN(I, 1), RANGE(4))
      DO 50 I = 1, N
      ATTEN(I, 1) = ATTEN(I, 1) - ATTMIN
50    CONTINUE
      IF (IT .GE. 2) GO TO 60
      CALL USPLT(FREQ, ATTEN, IY, N, M, INC, ITITLE, RANGE, ICHAR, IOPT,
     &IMAG4, IER)
      WRITE (6,90) (RANGE(I), I = 1, 4), FREQ(MATT), ATTMIN
60    RETURN
70    FORMAT ('1' / '0', 40X, 'FILTER ATTENUATION (AS PLOTTED)' / '-',
     &10X, 4(2X, 'FREQ', 8X, 'ATTEN(DB)', 4X) / '0')
80    FORMAT (' ', 10X, 4(F8.0, 5X, F9.4, 5X))
90    FORMAT ('-RANGES:   FREQ:', 2G13.5, ' ATTEN:', 2G13.5, 'MIN. ATTEN
     &. AT FREQ.:', G13.5, ' ATTEN.:', G13.5)
      END
C****************************************************************
C                                                               *
C S/R PLOT2 PLOTS THE PASSBAND AND THE FULL BAND                *
C                                                               *
C****************************************************************
      SUBROUTINE PLOT2(NG1, G1, IT)
      REAL RANGE(4), G1(1)
      COMMON / PLT2 / WC, ASMIN, RW, WC1
      COMMON / FREQPT / NG, G(20), W, FR1
      NG = NG1
      DO 10 I = 1, NG
10    G(I) = G1(I)
      RANGE(1) = 0.
      RANGE(2) = 128.
      RANGE(3) = 0.
      RANGE(4) = ASMIN * 1.5
      CALL PLOT(RANGE, IT)
      IF (IT .GE. 2) RETURN
      RANGE(1) = WC1
      RANGE(2) = WC * 17. / 16.
      RANGE(3) = 0.
      RANGE(4) = 1.25 * RW
      CALL PLOT(RANGE, 0)
      RETURN
      END
C****************************************************************
C                                                               *
C S/R PWR COMPUTES  PWR=1/2**I                                  *
C                                                               *
C****************************************************************
      FUNCTION PWR(I)
      IA = IABS(I)
      IF (IA .GE. 31) GO TO 10
      PWR = 2 ** IA
      IF (I .GE. 0) PWR = 1. / PWR
      RETURN
10    PWR = 1. / 2147483647
      RETURN
      END
C****************************************************************
C                                                               *
C S/R RELSEN COMPUTES THE RELATIVE SENSITIVITY OF EACH FILTER   *
C PARAMETER BASED ON THE OVERALL OBJECTIVE FUNCTION AND CREATES A *
C VECTOR INDEXING THEM IN ORDER OF DECREASING SENSITIVITY. JUNE 20,1984*
C VERSION: FEB 21'85                                            *
C                                                               *
C****************************************************************
      SUBROUTINE RELSEN(DXIN, NFE, NFPTS, S)
      LOGICAL XCHG, FAST
      INTEGER ORD(20), FIX(20), Q(20), DXCNT
```

```
      REAL D(20), S(1), DLB(20), DUB(20), TOL(100), WPTS(100), AWPTS(100
     &), DXIN(1)
      COMMON / FUNCS / ND, D, Q, NPASS, NSTOP, TOL, WPTS, AWPTS, F1EXIT,
     &DLB, DUB, ORD, FIX, IP, PI, FAST
      J = 0
      DXI = 20.
      DO 10 I = 1, ND
10    DXI = AMIN1(DXIN(I), DXI)
      DO 20 I = 1, ND
20    ORD(I) = I
      DO 60 I = 1, ND
      F = 1000
      FA = F
      FB = F
      DXCNT = 0
      IF (FIX(I) .EQ. 0) GO TO 30
      S(I) = 0.
      GO TO 60
30    DX = DXI
      DS = D(I)
40    D(I) = D(I) + DX
      CALL FUNCT(FA, D, ND, NFE, NFPTS)
      IF (FA .LT. 500) GO TO 50
      D(I) = D(I) - 2. * DX
      CALL FUNCT(FA, D, ND, NFE, NFPTS)
      IF (FA .LT. 500) GO TO 50
      D(I) = D(I) + DX
      DX = DX / 2.
      DXCNT = DXCNT + 1
      IF (DXCNT .GE. 20) GO TO 100
      GO TO 40
50    D(I) = DS
      CALL FUNCT(F, D, ND, NFE, NFPTS)
      S(I) = ABS((F - FA) / DX)
60    CONTINUE
      L = ND
70    XCHG = .FALSE.
      DO 80 I = 2, L
      J = ORD(I)
      J1 = ORD(I - 1)
      IF (S(J) .LE. S(J1)) GO TO 80
      XCHG = .TRUE.
      ORD(I) = J1
      ORD(I - 1) = J
80    CONTINUE
      IF ( .NOT. XCHG) GO TO 90
      L = L - 1
      IF (L .GE. 2) GO TO 70
90    CONTINUE
      RETURN
100   WRITE (6,110) F, FA, FB, DX, (D(I), I = 1, ND)
      RETURN
110   FORMAT ('-ABORT DURING RELSEN: DX REDUCED TOO MANY TIMES' / '0F=',
     &G13.5, ' FA=', G13.5, ' FB=', G13.5, ' DX=', G13.5 / '0D VECTOR:'
     & / ('0', 10G13.5))
      END
C*****************************************************************
C                                                                *
C S/R SEARCH PERFORMS MULTIVARIATE PROBE MOVEMENT FOR WEGENER'S SEARCH.*
C VERSION FEB 21'85,   OCT 5'84                                  *
C                                                                *
C*****************************************************************
      SUBROUTINE SEARCH(F0, X, DX, N, NFE, NSR, NFP)
      INTEGER P2M, QMAX
      REAL X(1), DX(1), E(20), B(20), C(128, 20)
      COMMON / SRCH / C, MOD, QMAX, F1EXIT
      COMMON / IOUT / IN, IO, ID1
      COMMON / CONTIN / ID, EXIT, NOPT, CNTRCT, M0
      IF (ID .GE. 4) WRITE (IO,110) (X(I), I = 1, N)
      FE = F0
      F = F0
      DO 70 M = 1, M0
      P2M = 2 ** M
      DO 60 K = 1, N
      DO 50 J = 1, P2M
      DO 10 I = 1, M
      L = MOD(I + K - 2, N) + 1
      E(I) = DX(L)
      B(I) = X(L)
      DX(L) = SIGN(DX(L), C(J, I))
      X(L) = X(L) + DX(L)
10    CONTINUE
      CALL FUNCT(FE, X, N, NFE, NFP)
      IF (ID .LT. 4) GO TO 20
      WRITE (IO,90) NFE, K, M, J, F, FE, (X(I), I = 1, N)
      WRITE (IO,100) (DX(I), I = 1, N)
20    IF (FE .GE. F) GO TO 30
      F = FE
```

```
      GO TO 60
30    DO 40 I = 1, M
      L = MOD(I + K - 2, N) + 1
      X(L) = B(I)
      DX(L) = E(I)
40    CONTINUE
50    CONTINUE
60    CONTINUE
      IF (F .LT. F0) GO TO 80
70    CONTINUE
80    NSR = NSR + 1
      F0 = F
      RETURN
90    FORMAT (' NFE:', I5, ' K:', I3, ' M:', I3, ' J:', I3, ' F:', G13.5
     &, ' FE:', G13.5 / ' X:', 10G13.5 / ' ', 10G13.5)
100   FORMAT (' DX:', 10G13.5 / ' ', 10G13.5)
110   FORMAT ('0BASIS POINT:', 9G13.5 / (' ', 10G13.5))
      END
C*****************************************************************
C                                                                *
C S/R SIGNO PERFORMS TRANSFER OF SIGNWITH ZERO FEATURE.          *
C                                                                *
C*****************************************************************
      FUNCTION SIGNO(MAG, SGN)
      REAL MAG
      IF (SGN .EQ. 0.) GO TO 10
      SIGNO = SIGN(MAG, SGN)
      RETURN
10    SIGNO = 0.
      RETURN
      END
C*****************************************************************
C                                                                *
C S/R SMPLFY FINDS THE MINIMUM WORDLENGTH REQUIRED TO REPRESENT D. *
C                                                                *
C*****************************************************************
      SUBROUTINE SMPLFY(ND, D, Q)
      INTEGER Q(1), PQ(20)
      REAL D(1)
      COMMON / SMFY / PQ
      DO 20 I = 1, ND
      DT = D(I) * PQ(I)
      IF (AINT(DT) .NE. DT) GO TO 20
10    DT = DT / 2
      IF (DT .NE. AINT(DT)) GO TO 20
      Q(I) = Q(I) - 1
      GO TO 10
20    CONTINUE
      RETURN
      END
C=================================================================
C THE FOLLOWING SUBROUTINES ARE CASE-DEPENDENT AND USER-SUPPLIED.
C=================================================================
C*****************************************************************
C                                                                *
C S/R FRQPT EVALUATES THE FREQUENCY RESPONSE AT ONE POINT FOR A  *
C 7TH ORDER ELLIPTIC FILTER.   SEPT 29'84                        *
C                                                                *
C*****************************************************************
      SUBROUTINE FRQPT
      COMMON / FREQPT / NGR, GR(20), W, FP
      COMPLEX Y(10), Z(3)
      W2 = W * W
      Y(1) = CMPLX(GR(1), W * GR(9))
      Z(1) = CMPLX(0., W / (GR(3) - W2 * GR(6)))
      Y(2) = CMPLX(0., W * GR(10))
      Z(2) = CMPLX(0., W / (GR(4) - W2 * GR(7)))
      Y(3) = CMPLX(0., W * GR(11))
      Z(3) = CMPLX(0., W / (GR(5) - W2 * GR(8)))
      Y(4) = CMPLX(GR(2), W * GR(12))
      Y(5) = 1. + Z(1) * Y(1)
      Y(6) = Y(1) + Y(2) * Y(5)
      Y(7) = 1. + Z(3) * Y(4)
      Y(8) = Y(4) + Y(3) * Y(7)
      Y(9) = Y(5) + Z(2) * Y(6)
      Y(10) = Y(6) * Y(7) + Y(8) * Y(9)
      FP = CABS((GR(1) + GR(2)) / Y(10))
      RETURN
      END
C*****************************************************************
C                                                                *
C S/R FUNC2 EVALUATES THE WORDLENGTH OBJECTIVE FUNCTION FOR A    *
C 7TH ORDER ELLIPTIC FILTER, MIDDLE BRUNE PORT 2 REFLECTION-FREE, *
C N1 AND N2 CONSTRAINED                                          *
C VERSION: 85 MAY  15,  STARTED 84 OCT 22                        *
C                                                                *
C*****************************************************************
      SUBROUTINE FUNC2(F2, QIN, X, N, NFE)
```

```
      IMPLICIT INTEGER(I - N, Q)
      LOGICAL FAST
      INTEGER QIN(1), Q(20), ORD(20), FIX(20), BM(32)
      REAL X(20), K(2, 2, 2)
      COMMON / FUNCS / ND, P(20), Q, NPASS, NSTOP, TOL(100), WPTS(100),
     &AWPTS(100), FIEXIT, DLB(20), DUB(20), ORD, FIX, IP, PI, FAST
      COMMON / FCN2 / INC(20), ICOPT
      QMAX = 26
      DO 10 I = 1, N
      P(ORD(I)) = X(I)
10    Q(ORD(I)) = QIN(I)
      F2 = 0
      K1 = 0
      K2 = 0
      K3 = 0
      K4 = 0
C--------------------------------------------------------------------------
      DO 20 I = 1, 2
      IF (INC(I) .EQ. 0) GO TO 20
      PQ = PWR(Q(I))
      CALL BITS(IFIX(P(I) / PQ), Q(I), IB, BM)
      K1 = IB / PQ + K1
20    CONTINUE
      F2 = F2 + K1
C-------------------------------------------------------------------------
30    ISUM = INC(3) + INC(4) + INC(5)
      IF (ISUM .EQ. 0) GO TO 60
      IF (ISUM .EQ. 3) GO TO 50
      DO 40 I = 3, 5
      IF (INC(I) .EQ. 0) GO TO 40
      PQ = PWR(Q(I))
      CALL BITS(IFIX(P(I) / PQ), Q(I), IB, BM)
      K2 = IB / PQ + K2
40    CONTINUE
      F2 = F2 + K2
      GO TO 60
50    C1 = P(3) * P(5)
      C2 = C1 + 0.5
      C3 = 1 - 2 * (1 - P(3)) * P(4) * C2
      C4 = 1 - 2 * C1 * P(4)
      C6 = (1 - C2) * C3
      K(1, 1, 1) = C4 * C6
      K(2, 1, 1) = C2 * C4
      K(1, 2, 1) = C3 * P(5) * (2 * P(4) * (1 - P(3)) + C3) + C6 * (1 -
     &C4)
      K(2, 2, 1) = C2 - C3 * P(5) - K(2, 1, 1)
      CALL FNDWL(K(1, 1, 1), QMAX, QA)
      CALL FNDWL(K(2, 1, 1), QMAX, QB)
      CALL FNDWL(K(1, 2, 1), QMAX, QC)
      CALL FNDWL(K(2, 2, 1), QMAX, QD)
      K2 = MAX0(QA, QB, QC, QD)
      F2 = F2 + 1 / PWR(K2)
C--------------------------------------------------------------------------
60    DO 70 I = 6, 7
      IF (INC(I) .EQ. 0) GO TO 70
      PQ = PWR(Q(I))
      CALL BITS(IFIX(P(I) / PQ), Q(I), IB, BM)
      K3 = IB / PQ + K3
70    CONTINUE
      F2 = F2 + K3
C--------------------------------------------------------------------------
80    DO 90 I = 8, 9
      IF (INC(I) .EQ. 0) GO TO 90
      PQ = PWR(Q(I))
      CALL BITS(IFIX(P(I) / PQ), Q(I), IB, BM)
      K4 = IB / PQ + K4
90    CONTINUE
      F2 = F2 + K4
C--------------------------------------------------------------------------
100   NFE = NFE + 1
      IF (IP .EQ. 0) GO TO 110
      WRITE (6,120) F2, K1, K2, K3, K4, ((K(I, J, 1), J = 1, 2), I = 1,
     &2)
110   RETURN
120   FORMAT ('0F2: ', G13.6, '  SECTIONS:', 4G13.6 / '0K MATRIX:',
     &2G13.6 / 12X, 2G13.6)
      END
C************************************************************************
C                                                                       *
C S/R STABND EVALUATES THE STABILITY BOUNDS FOR ONE VARIABLE.           *
C 7TH-ORDER ELLIPTIC, MIDDLE BRUNE PORT 2 RF                            *
C VERSION: 85 APR 02                                                    *
C                                                                       *
C************************************************************************
      SUBROUTINE STABND(NX, X, I, XOLB, XOUB)
      REAL X(1), XOLB(1), XOUB(1)
      IF (I .EQ. 5) GO TO 10
      XOLB(I) = 0.
```

```
      XOUB(1) = 1.
      RETURN
10    XOLB(1) = 0.
      XOUB(1) = 100.
      RETURN
      END
C***********************************************************************
C                                                                      *
C S/R XD2G TRANSFORMS DESIGN PARAMETERS INTO G-RATIOS FOR 7TH-ORDER    *
C ELLIPTIC FILTER WITH 2 CONSTRAINTS,  MIDDLE BRUNE PORT 2 R.F.        *
C VERSION APR 02, 1985,  STARTED  OCT 12'84                            *
C                                                                      *
C***********************************************************************
      SUBROUTINE XD2G(NP, P, NGR, GR, IER)
      REAL P(20), GR(20), WI, FP
      IER = 0
      GR(10) = 1. - P(6) - P(7)
      GR(9) = (1 - P(2)) * P(6) / P(1) / P(2)
      C = P(1) * GR(9)
      GR(6) = C / (1. - P(1))
      C1 = (1. - P(1))
      GR(3) = P(6) / C1
      GR(1) = P(6) / P(1)
      GR(10) = GR(10) - C
      C3 = 2 * P(5) * P(7)
      C = C3 * P(3)
      GR(7) = C / (1. - P(3))
      C1 = P(4) * (1. - P(3))
      GR(4) = P(7) / C1
      C2 = P(7) + C
      T2 = 1. - C2 / GR(4)
      IF (T2 .EQ. 0) GO TO 10
      C4 = C2 / T2 - C3
      GR(4) = GR(4) - GR(7)
      GR(10) = GR(10) - C
      GR(2) = C4 / P(8)
      GR(5) = C4 / (1. - P(8))
      GR(8) = (1. - P(9)) / P(9)
      GR(12) = GR(8) * GR(2)
      GR(8) = GR(8) * GR(5)
      GR(11) = C3 - P(8) * GR(12)
      NGR = NP + 3
      RETURN
10    IER = 1
      NGR = NP + 3
      RETURN
      END
//GO.SYSIN DD *
WEGENER 7TH-ORD ELLIPTIC, PORT 2 REFL-FREE, EXAMPLE 6
ATTENUATION CHARACTERISTIC
FREQUENCY                                DB    ATTENUATION
0.  36.26  42.7  .55  50.                /WC1,WC,WS,RW,ASMIN
9  1.  .112164  .502707  .252173  .217023  6.539907
256  3  3                                /NX,NPX,NSX
 0.    29.86  .22                        /R(1,1),R(2,1),BTOL(1)
29.86  34.13  .33
34.13  36.26  .55
42.7   51.2   30.0
51.2   57.6   40.0
57.6  127.    50.0
19  8                                    /NPASS,NSTOP
36.26      .55                           /WPTS(1),TOL(1)
36.        .55
35.5       .55
35.        .55
34.13      .33
34.        .33
33.        .33
32.        .33
31.        .33
29.86      .22
29.        .22
27.        .22
25.        .22
23.        .22
21.        .22
17.        .22
13.        .22
11.        .22
42.7     30.
48.      30.
51.2     40.
54.      40.
57.6     50.
66.      50.
80.      50.
95.      50.
7                                        /NSTG
```

```
1  3          0 0 0 0 0 0 0 0 0     /NINC,VINC(I),SFIX(J,I)
1  5          0 0 1 0 0 0 0 0 0
1  4          0 0 1 0 1 0 0 0 0
3  3 4 5      0 0 0 0 0 0 0 0 0
2  6 7        0 0 1 1 1 0 0 0 0
2  1 2        0 0 1 1 1 1 1 0 0
2  8 9        1 1 1 1 1 1 1 0 0
1  4   12  0                        /F1EXIT,M0,QMAX,QRED
4  1    1  1  3                     /DBUG,ICOPT,NRPT,NOPT,NQOPT
-1 6    3  2. 9. 1                  /IDC,M0C,NOPTC,CNTRC,EXITC,ISTOP
/*
```

Appendix C. Design and Analysis Equations for Ladder-Based Examples

The design and analysis equations (given in the left and right columns, respectively) used for the ladder-based design examples are included here in algorithm form.

C.1 5th-Order Elliptic Ladder Filter

$$\beta_1 = \frac{G_1}{G_1 + G_7}$$

$$\beta_2 = \frac{G_2}{G_2 + G_9}$$

$$\beta_3 = \frac{G_5}{G_5 + G_7}$$

$$\beta_4 = \frac{G_6}{G_6 + G_9}$$

$$\beta_5 = \frac{\beta_1}{1 + \dfrac{G_8+\beta_4(G_2+G_9)}{\beta_3(\beta_3(G_1+G_7))}}$$

$$\beta_6 = \frac{\beta_4\beta_5 G_2}{\beta_1 G_1}$$

$$G_1 = \frac{B_5}{\beta_3}$$

$$G_2 = \frac{\beta_6}{\beta_4}$$

$$G_3 = \frac{\beta_5}{(1-\beta_3)}$$

$$G_4 = \frac{\beta_6}{(1-\beta_4)}$$

$$c_1 = \frac{1}{\beta_1}-1$$

$$c_2 = \frac{1}{\beta_2}-1$$

$$G_5 = c_1 G_3$$

$$G_6 = c_2 G_4$$

$$G_7 = c_1 G_1$$

$$G_9 = c_2 G_1$$

$$G_8 = 1-\frac{\beta_5}{\beta_1}-\frac{\beta_6}{\beta_2}$$

C.2 7th-Order Elliptic Filter, Middle Brune Non-Reflection-Free.

$$n_1 = \frac{G_6}{G_6 + G_9}$$

$$n_2 = \frac{G_8}{G_8 + G_{12}}$$

$$n_3 = \frac{G_7}{G_7 + G_{11} + n_2 G_{12}}$$

$$\beta_1 = n_1$$

$$c_1 = \beta_1 G_1$$

$$\beta_2 = \frac{G_1}{G_1 + G_9}$$

$$c_5 = G_{11} + n_2 G_{12}$$

$$c_2 = G_{10} + n_1 G_9 + n_3 c_5$$

$$G_7 = \beta_4$$

$$c_3 = 1 - \beta_4$$

$$c_4 = 1/\beta_7$$

$$c_1 = c_4/\beta_5 - c_3$$

$$c_2 = c_4/\beta_6 + \beta_4 c_3$$

$$G_4 = \frac{c_4}{1-\beta_5-\beta_6}-\beta_4$$

$$G_2 = c_1/\beta_8$$

$$G_5 = \frac{c_1}{1-\beta_8}$$

$$\beta_3 = \frac{c_1}{c_1 + c_2} \beta_4 = n_2$$

$$c_3 = c_1 + c_2$$

$$c_4 = \cfrac{1}{\cfrac{1}{n_2 G_2 + c_5} + \cfrac{1}{G_4 + G_7} + \cfrac{1}{c_3 - n_3 c_5}}$$

$$c_8 = (1-\beta_9)/\beta_9$$
$$G_8 = c_8 G_5$$
$$G_{12} = c_8 G_2$$
$$G_{11} = c_3 - \beta_8 G_{12}$$
$$c_1 = \beta_3 c_2$$

$$\beta_5 = \frac{c_4}{n_2 G_2 + c_5}$$

$$\beta_6 = \frac{c_4}{c_3 - n_3 c_5}$$

$$\beta_7 = (G_7 + c_5)/c_4$$

$$\beta_8 = n_3$$

$$\beta_9 = \frac{G_2}{G_2 + G_{12}}$$

$$G_{10} = c_2 - c_1 - \beta_4 c_3$$
$$c = (1-\beta_2)c_1/\beta_2$$
$$G_9 = c/\beta_1$$
$$G_6 = \frac{c}{1-\beta_1}$$
$$G_3 = \frac{c_1}{1-\beta_1}$$
$$G_1 = c_1/\beta_1$$
$$G_{10} = G_{10} - c$$

C.3 7th-Order Elliptic Filter, Middle Brune Port 1 Reflection-Free.

$$\beta_1 = \frac{G_6}{G_6 + G_9}$$

$$\beta_2 = \frac{G_8}{G_8 + G_{12}}$$

$$c_1 = G_{10} + \beta_1 G_9$$

$$\beta_3 = \frac{G_7}{G_7 + c_1}$$

$$G_6 = \beta_1 \beta_2$$
$$G_9 = (1-\beta_1)\beta_4$$
$$G_1 = 1/\beta_2 - G_9$$
$$G_3 = 1/(1-\beta_2-\beta_3) - G_6$$

$$c_2 = G_6 + G_9$$
$$c_3 = c_1 + G_7$$
$$c_4 = G_8 + G_{12}$$
$$c_5 = G_{11} + \beta_9 G_{12} + \beta_5 c_1$$

$$c_6 = 1/\beta_3 - \beta_1 G_9$$
$$c_1 = c_6(1-2\beta_6)$$
$$c_2 = \beta 7(1-\beta_5)$$
$$c_3 = \frac{c_1}{1-\beta_5}$$

$$c_6 = \cfrac{1}{\cfrac{1}{G_2 + G_{12}} + \cfrac{1}{G_5 + G_8}} - \beta_9 G_{12}$$

$$c_7 = c_5 + c_6$$

$$c_8 = c_1 + \cfrac{1}{\cfrac{1}{G_4 + G_7} + \cfrac{1}{c_7 - \beta_5 c_1}}$$

$$G_7 = \beta_5 c_3$$
$$G_4 = c_6/c_2 - G_7$$

$$c_9 = \cfrac{1}{\cfrac{1}{G_1 + (1-\beta_1)c_2} + \cfrac{1}{G_3 + \beta_1 c_2} + \cfrac{1}{c_8 - \beta_1(1-\beta_1)c_2}}$$

$$c_8 = 2\beta_6 \frac{c_6}{1 - 2c_2\beta_6}$$

$$c_7 = c_8\beta_8$$

$$\beta_2 = \frac{c_9}{G_1 + (1-\beta_1)c_2}$$

$$\beta_3 = \frac{c_9}{c_8 - \beta_1(1-\beta_1)c_2}$$

$$\beta 4 = c_2/c_9$$

$$\beta_6 = (c_8 - (1-\beta_5)c_3)/2c_8$$

$$G_5 = \frac{c_7}{1 - \beta_9}$$

$$G_8 = (1-\beta_{10})G_5/\beta_{10}$$

$$G_2 = c_7/\beta_9$$

$$c_3 = c_3 - G_7$$

$$\beta_7 = \frac{c_8}{(1-\beta_5)(G_4 + \beta_5 c_3)}$$

$$\beta_8 = \frac{c_6}{c_5 + c_6}$$

$$\beta_{10} = \frac{G_2}{G_2 + G_{12}}$$

$$c_4 = \beta_1 G_9$$

$$G_{10} = c_3 - c_4$$

$$c_5 = (1-\beta_9)G_8$$

$$G_{12} = c_5/\beta_9$$

$$G_{11} = c_8 - c_7 - c_5 - \beta_5 c_3$$

C.4 7th Order Elliptic Filter, Middle Brune Port 2 Reflection-Free.

$$\beta_1 = n_1$$

$$\beta_2 = \frac{G_1}{G_1 + G_5}$$

$$\beta_3 = n_3$$

$$c_5 = G_{11} + n_3 G_{12}$$

$$G_{10} = 1 - \beta_6 - \beta_7$$

$$G_9 = (1-\beta_2)\beta_6/\beta_1\beta_2$$

$$c = G_1\beta_9$$

$$G_6 = \frac{c}{1 - \beta_1}$$

$$c_2 = \cfrac{1}{\cfrac{1}{n_3 G_2 + c_5} + \cfrac{1}{G_4 + G_7}} - n_3 c_5$$

$$\beta_4 = \frac{c_2}{(1-n_3)(G_4 + G_7)}$$

$$\beta_5 = c_5/2c_2$$

$$c_3 = G_{10} + n_1 G_9 + n_3 c_5$$

$$c_1 = 1 - \beta_1$$

$$G_3 = \beta_6/c_1$$

$$G_1 = \beta_6/\beta_1$$

$$G_{10} = G_{10} - c$$

$$c_3 = 2\beta_5\beta_7$$

$$c = c_3\beta_3$$

$$G_7 = \frac{c}{1 - \beta_3}$$

$$c_4 = c_1 + c_2 + c_3$$

$$\beta_6 = c_1/c_4$$

$$\beta_7 = c_2/c_4$$

$$c_1 = \beta_4(1-\beta_3)$$

$$G_4 = \beta_7/c_1$$

$$c_2 = \beta_7 + c$$

$$c_4 = \frac{c_2}{1 - c_2/G_4} - c_3$$

$$G_4 = G_4 - G_7$$

$$G_{10} = G_{10} - c$$

$$G_2 = c_4/\beta_8$$

$$\beta_8 = n_2$$

$$G_5 = \frac{c_4}{1 - \beta_8}$$

$$\beta_9 = \frac{G_2}{G_2 + G_{12}}$$

$$G_8 = (1-\beta_9)/\beta_9$$

$$G_{12} = G_2 G_8$$

$$G_8 = G_8 G_5$$

$$G_{11} = c_3 - \beta_8 G_{12}$$

Appendix D. Frequency Response Algorithms for Fifth- and Seventh-Order Ladder Elliptic Filters

D.1 5th-Order Elliptic Ladder Filter Network.

$$Y_1 = G_1 + j\omega G_7 \qquad Y_2 = \omega G_8$$
$$Y_3 = G_2 + \omega G_9 \qquad Y_4 = 1 + Z_1 Y_1$$
$$Z_1 = \frac{\omega}{G_3 - \omega^2 G_5} \qquad Y_5 = Y_1 + Y_2 Y_4$$
$$Y_6 = Y_4 + Z_2 Y_5$$
$$Z_2 = \frac{\omega}{G_4 - \omega^2 G_6} \qquad F = \left| \frac{G_1 + G_2}{Y_5 + Y_3 Y_6} \right|$$

D.2 7th-Order Elliptic Ladder Filter Network.

$$Y_1 = G_1 + \omega G_9 \qquad Y_4 = G_2 + \omega G_{12}$$
$$Z_1 = \frac{\omega}{G_3 - \omega^2 G_6} \qquad Y_5 = 1 + Z_1 Y_1$$
$$Y_6 = Y_1 + Y_2 Y_5$$
$$Y_2 = \omega G_{10} \qquad Y_7 = 1 + Z_3 Y_4$$
$$Z_2 = \frac{\omega}{G_4 - \omega^2 G_7} \qquad Y_8 = Y_4 + Y_3 Y_7$$
$$Y_9 = Y_5 + Z_2 Y_6$$
$$Y_3 = \omega G_{11} \qquad Y_{10} = Y_6 Y_7 + Y_8 Y_9$$
$$Z_3 = \frac{\omega}{G_5 - \omega^2 G_8} \qquad F = \left| \frac{G_1 + G_2}{Y_{10}} \right|$$

References

1. B. Liu, "Effect of Finite Ford Length on the Accuracy of Digital Filters: A Review," *IEEE Trans. Circuit Th.*, vol. CT-18, pp.670-677, Nov. 1971.
2. A.V. Oppenheim, C.J. Weinstein, "Effects of Finite Register Length in Digital Filtering and the Fast Fourier Transform," *Proc. IEEE*, vol. 60, pp. 957-967, Aug. 1972.
3. J.W.K. Lam, V. Ramachandran, M.N.S. Swamy, "Comparison of the Effects of Quantization on Digital Filters," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-30, no. 6, pp. 1010-1013, Dec. 1982.
4. A. Fettweis, "Roundoff Noise and Attenuation Sensitivity in Digital Filters with Fixed-Point Arithmetic," *IEEE Trans. Circuit Th.*, vol. CT-20, no. 2, pp. 174-175, Mar. 1973.
5. T.A.C.M. Claasen, W.F.G. Mecklenbräuker, J.B.H. Peek, "Effects of Quantization and Overflow in Recursive Digital Filters," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-24, pp. 517-529, Dec. 1976.
6. R.E. Crochiere, "Digital Ladder Structures and Coefficient Sensitivity," *IEEE Trans. Audio Electroacoust.*, vol. AU-20, no. 4, pp. 240-318, 1972.
7. E. Avenhaus, "A Proposal to Find Suitable Canonical Structures for the Implementation of Digital Filters with Small Coefficient Wordlength," *Nachrichtentech. Z.*, vol. 25, pp. 377-382, Aug. 1972.
8. T.L. Chang, "On Low-Roundoff Noise and Low-Sensitivity Digital Filter Stuctures," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-29, pp. 1077-1080, Oct. 1981.
9. J.B. Jackson, "On the Interaction of Roundoff Noise and Dynamic Range in Digital Filters," *Bell Sys. Tech. J.*, vol. 49, no. 2, pp. 159-184, Feb. 1970.
10. A. Fettweis, "On the Connection Between Multiplier Word Length Limitation and Roundoff Noise in Digital Filters," *IEEE Trans. Circuit Th.*, vol. CT-19, no.5, pp. 486-491, Sept. 1972.
11. W.L. Mills, C.T. Mullis, R.A. Roberts, "Low Roundoff Noise and Normal Realizations of Fixed Point IIR Digital Filters," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-29, pp. 893-903, Aug. 1981.
12. B.W. Bowmar, J.C. Hung, "Minimum Roundoff Noise Digital Filters with Some Powers-of-Two Coefficients," *IEEE Trans. Circuits Syst.*, vol. CAS-31, no. 10, pp. 833-840, Oct. 1984.
13. M. Arjmand, R.A. Roberts, "Reduced Multiplier, Low Roundoff Noise Digital Filters," in *Proc. IEEE Conf. Acoust. Speech, Signal Proc.*, Washington,D.C., Apr. 1979.
14. C.T. Mullis, R.A. Roberts, "Synthesis of Minimum Roundoff Noise Fixed Point Digital Filters," *IEEE Trans. Circuits Syst.*, vol. CAS-23, pp. 551-562, Sept. 1976.
15. C.W. Barnes, "Computationally Efficient Second-Order Digital Filter Sections with Low Roundoff Noise Gain," *IEEE Trans. Circuits Syst.*, vol. CAS-31, no. 10, pp. 841-847, Oct. 1984.
16. V. Tavsanoglu, L. Thiele, "Optimal Design of State-Space Digital Filters by Simultaneous Minimization of Sensitivity and Roundoff Noise," *IEEE Trans. Circuits Syst.*, vol. CAS-31, no. 10, pp. 884-888, Oct. 1984.
17. K. Meerkötter, W.Wegener, "A New Second-Order Digital Filter Without Parasitic Oscillations," *Arch. Elek. Übertragung*, vol. AEÜ-29, no. 7-8, pp. 312-314, 1975.

18. K. Meerkötter, "Realization of Limit Cycle-Free Second-Order Digital Filters," in *Proc. IEEE Int. Symp. Circuits Syst.*, pp. 295-298, 1976.

19. T.L. Chang, "Suppression of Limit Cycles in Digital Filters Designed with One Magnitude-Truncation Quantizer," *IEEE Trans. Circuits Syst.*, vol. CAS-28, no. 2, pp. 107-111, Feb. 1981.

20. A.T. Fam, C.W.Barnes, "Non-Minimal Realizations of Fixed-Point Digital Filters that are Free of All Finite Word-Length Limit-Cycles," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-27, no. 2, pp. 149-153, Apr. 1979.

21. H.J. Butterweck, A.C.P. van Meerz, G. Verkroost, "New Second-Order Digital Filter Sections Without Limit Cycles," *IEEE Trans. Circuits Syst.*, vol. CAS-31, no. 2, pp. 141-146, Feb. 1984.

22. T.A.C.M. Claasen, W.F.G. Mecklenbräuker, J.B.H Peek, "Second-Order Digital Filter with Only One Magnitude-Truncation Quantiser and Having Practically No Limit Cycles," *Electron. Lett.*, vol.9, pp. 531-532, Nov. 1, 1973.

23. E.D. Deluca, G.O Martens, "A New Coefficient Quantization Scheme to Suppress Limit Cycles in State-Variable Digital Filters," in *Proc. 6th European Conf. on Circuit Theory and Design*, 1983.

24. C.W.Barnes, A.T. Fam, "Minimum Norm Recursive Digital Filters that are Free of Overflow Limit Cycles," *IEEE Trans. Circuits Syst.*, vol. CAS-24, no.10, pp. 569-574, Oct. 1977.

25. A.G. Bolton, "A Two's Complement Overflow Limit Cycle Free Digital Filter Structure," *IEEE Trans. Circuits Syst.*, vol. CAS-31, no. 12, pp. 1045-1046, Dec. 1984.

26. W.L. Mills, C.T. Mullis, R.A. Roberts, "Digital Filter Realizations without Overflow Oscillations," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-26, No.4, pp. 334-338, Aug. 1978.

27. B. Eckhardt, W. Winkelkemper, "Implementation of a Second Order Digital Filter Section with Stable Overflow Behavior," *Nachrichtentech. Z.*, vol.26, pp. 282-284, June 1973.

28. A. Fettweis, "Digital Filter Structures Related to Classical Filter Networks," *Arch. Elek. Ubertragung*, vol. AEU-25, pp. 79-89, 1971.

29. A. Fettweis, "Canonic Realizations of Ladder Wave Digital Filters," *Int. J. Circuit Th. and Appl.*, vol.3, pp. 321-332, 1975.

30. A. Fettweis, "On Sensitivity and Roundoff Noise in Wave Digital Filters," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-22, pp. 383-384, Oct. 1974.

31. A. Fettweis, "Pseudopassivity, Sensitivity, and Stability of Wave Digital Filters," *IEEE Trans. Circuit Th.*, vol. CT-19, pp. 668-673, Nov. 1972.

32. A. Fettweis, "Reciprocity, Inter-Reciprocity, and Transposition in Wave Digital Filters," *Circuit Th. & Appl.*, pp. 323-337, 1973.

33. A. Fettweis, "Some Principles of Designing Digital Filters Imitating Classical Filter Structures," *IEEE Trans. Circuit Th.*, vol. CT-18, no. 2, pp. 314-316, Mar. 1971.

34. A. Fettweis, "Wave Digital Filters with Reduced Number of Delays," *Circuit Th. & Appl.*, vol. 2, pp. 319-330, 1974

35. A. Fettweis, K. Meerkötter, "On Adaptors for Wave Digital Filters," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-23, pp. 516-525, Dec. 1975.

36. K. Meerkötter, "Beiträge zur Theorie der Wellendigitalfilter," Ph.D. Dissertation, Dept. of EE., Ruhr-Universität, Bochum, F.R. of Germany, Jan 1979.

37. A. Sedlmeyer, A. Fettweis, "Digital Filters with True Ladder Configuration," *Circuit Th. & Appl.*, vol. 1, pp. 5-10, 1973.

38. A. Fettweis, "Realizability of digital filter networks," *Arch. Elek. Übertragung*, vol. AEÜ-30, pp. 90-96, Feb. 1976.

39. A. Fettweis, K. Meerkötter, "Suppression of Parasitic Oscillations in Wave Digital Filters," *IEEE Trans. Circuits Syst.*, vol. CAS-22, no. 3, pp. 239-246, Mar. 1975.

40. A.T Ashley, "Minimal Wave Digital Filter Realizations: The N-port Approach," Ph.D. Thesis, Dept. of EE., U. of Manitoba, Winnipeg, Canada, R3T 2N2, May 1978.

41. A. Fettweis, H. Levin, A. Sedlmeyer, "Wave Digital Lattice Filters," *Circuit Th. & Appl.*, vol. 2, pp. 203-211, 1974.

42. H.H. Lê, "Wave Digital Adaptors for Brune, Darlington C and D, and Twin T Sections," Ph.D. thesis, University of Manitoba, Canada, Aug. 1977.

43. M.R. Jarmasz, "Design of Canonic Wave Digital Filters Which Suppress All Zero-Input Parasitic Oscillations," M.Sc. Thesis, University of Manitoba, Canada, 1983.

44. M. Suzuki, N. Miki, N. Nagai, "New Wave Digital Filters for Basic Reactance Sections," *IEEE Trans. Circuits Syst.*, vol. CAS-32, no. 4, pp. 337-348, Apr. 1985.

45. W. Wegener, "Entwurf von Wellendigitalfiltern mit minimalem Realisierungsaufwand," Ph.D. Dissertation, Ruhr-Universität, Bochum, F.R. Germany, 1979.

46. K.A. Owenier, "Optimization of Wave Digital Filters with Reduced Number of Multipliers," *Arch. Elek. Übertragung*, vol. AEÜ-30, pp. 387-393, 1976.

47. W. Wegener, "Wave Digital directional Filters with Reduced Number of Multipliers and Adders," *Arch. Elek. Übertragung*, vol. AEÜ-33, no. 6, pp. 239-243, 1979.

48. W. Wegener, "Wave Digital Directional Filters with Reduced Number of Multipliers and Adders," *Arch. Elek. Übertragung*, vol. AEÜ-33, no. 6, pp. 239-243, 1979.

49. W. Wegener, "On the Design of Wave Digital Lattice Filters with Short Coefficient Word Lengths and Optimal Dynamic Range," *IEEE Trans. Circuits Syst.*, vol. CAS-25, pp. 1091-1098, Dec. 1978.

50. C. Charalambous, M.J. Best, "Optimization of Recursive Digital Filters with Finite Word Lengths," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-22, pp. 424-431, Dec 1974.

51. B. Liu, A. Peled, "Heuristic Optimization of the Cascade Realization of Fixed-Point Digital Filters," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-23, pp. 464-473, Oct. 1975.

52. M. Suk, S.K. Mitra, "Computer-Aided Design of Digital Filters with Finite Wordlength," *IEEE Trans. Audio Electroacoust.*, vol. AU-20, pp. 356-363, Apr. 1972.

53. T.P. Lee, W.T. Nye, A.L. Tits, "Design of Digital Filters Using Interactive Optimization," *IEEE Trans. Circuits Syst.*, vol. CAS-30, no. 11, pp. 821-824, Nov. 1983.

54. H.-K. Kwan, "On the Problem of Designing IIR Digital Filters with Short Coefficient Wordlengths," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-27, no. 6, pp. 620-624, Dec. 1979.

55. F. Brglez, "Digital Filter Design with Short Word-Length Coefficients," *IEEE Trans. Circuits Syst.*, vol. CAS-25, no. 12, pp. 1044-1050, Dec. 1978.

56. R. Jain, J. Vandeville, H. De Man, "Discrete Optimization for the CAD of Arbitrary Integrated Digital Filters," in *Proc. 6th European Conf. Circuit Th. & Design*, Stuttgart, pp. 414-416, 1983.

57. R. Bolton, P.C. Craig, L.C. Westphal, "Computer-Aided Design of Recursive Digital Filters with Coefficients Having Restricted Minimal Representation," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-29, no. 6, pp. 1205-1208, Dec. 1981.

58. E. Avenhaus, "On the Design of Digital Filters with Coefficients of Limited Word Length," *IEEE Trans. Audio Electroacoust.*, vol. AU-20, pp. 206-212, Aug 1972.

59. P. Chambon, A. Desblache, "Integer Coefficients Optimization of Digital Filters," in *Proc. IEEE Symp. Circuits & Syst.*, Munchen, pp. 461-464, Apr. 1976.

60. R. Boite, H. Dubois, H. Leich, "Optimisation of Digital Filters in the Discrete Space of Coefficients," *Electron. Lett.*, vol. 10, pp. 179-180, May 1974.

61. K. Hofer, "Minimization of the Number of Additions in Digital Filters Without Multipliers," *Proc. 6th European Conf. Circuit Th. & Design*, Stuttgart, pp. 417-419, 1983.

62. N.I. Smith, "A Random-Search Method for Designing Finite-Wordlength Recursive Digital Filters," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-27, pp. 40-46, Feb. 1979.

63. K. Steiglitz, "Designing Short-Word Recursive Digital Filters," *Proc. 9th Allerton Conf. Circuits and System Theory*, Monticello, IL, pp. 778-788, Oct. 1971

64. R. Nouta, "The Jaumann Structure in Wave-Digital Filters," *Circuit Th. & Appl.*, vol. 2, pp. 163-174, 1974.

65. G.O. Martens, "Voltage Scattering Matrix Analysis of N-Ports With Applications to Wave-Digital Filters," *Proc. Int. Conf. Circuits & Computers*, pp. 1180-1183, 1980.

66. G.O. Martens, K. Meerkötter, "On n-Port Adaptors for Wave Digital Filters with Application to a Bridged-Tee Filter," *Proc. IEEE Int. Symp. Circuits Syst.*, pp. 514-517, 1976.

67. G.O. Martens, H.H. Lê, "Wave Digital Adaptors for Reciprocal Second-Order Sections," *IEEE Trans. Circuits Syst.*, vol. CAS-25, pp. 1077-1083, Dec. 1978.

68. E.D. DeLuca, "Minimal Wave Digital Filters Based On Strictly Pseudopassive N-port Adaptors," M.Sc. Thesis, University of Manitoba, Canada, 1983.

69. G. Verkroost, H.-J. Butterweck, "Suppression of Parasitic Oscillations in Wave Digital Filters and Related Structures by Means of Controlled Rounding," *Arch. Elek. Übertragung*, vol. AEÜ-30, no.5, pp. 181-186, May 1976.

70. R.E. Kalman, J.E. Bertram, "Control System Analysis and Design Via the 'Second Method' of Lyapunov-II Discrete-Time Systems," *Trans. ASME, J. Basic Engrg.*, pp. 394-400, June 1960.

71. A. Fettweis, "Canonic Realization of Ladder Wave Digital Filters," *Circuit Th. & Appl.*, vol.3, pp. 321-332, 1975.

72. G.O. Martens, M.R. Jarmasz, "Design of Digital Filters with a Diagonal Lyapunov Function Based on Elliptic Reference Filters," *Proc. IEEE Int. Symp. Circuits & Syst.*, Rome, pp. 527-530, May 1982.

73. V. Belevitch, *Classical Network Theory*, San Francisco: Holden-Day, 1968.

74. A. Fettweis, "Scattering Properties of Real and Complex Lossless 2-ports," *IEE Proc.*, vol. 128, pt.G, no.4, pp. 147-148, Aug. 1981.

75. A. Fettweis, "On the Scattering Matrix and the Scattering Matrix of Multidimensional Lossless Two-ports," *Arch. Elek. Übertragung*, vol. AEÜ-36, no.9, pp. 374-381, 1982.

76. L. Gazsi, "Explicit Formulas for Lattice Wave Digital Filters," *IEEE Trans. Circuits Syst.*, vol. CAS-32, no. 1, pp. 68-88, Jan 1985.

77. J.-P. Thiran, "On Unit Element Structures for Wave Digital Filters," *IEEE Trans. Circuits Syst.*, vol. CAS-24, no.1, pp. 20-28, Jan. 1977.

78. K. Meerkötter, "Canonic Realization of Wave Digital Filters Involving Unit Elements," *Circuit Th. & Appl.*, vol.5, pp. 395-402, 1977.

79. P.I. Richards, "Resistor Transmission Line Circuits," *IRE Proc.*, pp. 217-220, Feb. 1948.

80. G.W. Reitweisner, "Binary arithmetic," in *Advances in Computers*, F.L.Alt ed., vol.1., pp. 231-308, New York: Academic Press, 1960.

81. L.B. Jackson, A.G. Lindgren, Y. Kim, "Optimal Synthesis of Second-Order State-Space Structures for Digital Filters," *IEEE Trans. Circuits Syst.*, vol. CAS-26, pp. 149-153, Mar. 1979.

82. P.R. Moon, G.O. Martens, "A Digital Filter Structure Requiring Only m-bit Delays, Shifters, Inverters, and m-bit Adders Plus Simple Logic Circuitry," *IEEE Trans. Circuits Syst.*, vol. CAS-27, no.10, pp. 901-908, Oct. 1980.

83. A. Peled, B. Liu, "A New Hardware Realization of Digital Filters," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-22, no. 6, pp. 456-462, Dec. 1974.

84. A. Crosier, D.J. Esteban, M.E. Levilion, V.Rizo, "Digital Filters for PCM Encoded Signals," *U.S. Patent 3 777 130*, Dec. 1973.

85. L. Gazsi, "Single Chip Filter Bank with Wave Digital Filters," *IEEE Trans. Acoust. Speech, Signal Proc.*, vol. ASSP-30, pp.709-718, Oct. 1982.

86. R. Hooke, T.A. Jeeves, "Direct Search Solution of Numerical and Statistical Problems," *J. Ass. Comput. Mach.*, vol.8, pp. 212-229, Apr. 1961.

87. A.I. Zverev, *Handbook of Filter Synthesis*, New York: Wiley, Englewood Cliffs, N.J. 1967.

88. J.A. Nelder, R. Mead, "A Simplex Method for Function Minimization," *Computer Journal*, no.7, pp. 308-313, 1965.

89. L. Wanhammer, "An Approach to LSI Implementation of Wave Digital Filters," Ph.D. Dissertation, Linkoping University, Sweden, April 1981.

90. R. Saal, *Handbook of Filter Design*, AEG Telefunken, Backnang, F.R. Germany, 1979.

91. A. Fettweis, "Multiplier-Free Modulation Schemes for PCM to FDM and Audio to FDM Conversion," *Arch. Elek. Übertragung*, vol. AEU-32, pp. 477-485, 1978.