AN INVESTIGATION INTO CURRENT POSSIBILITIES IN AUTOMATED NETWORK DESIGN

by

Osman H.M.O. Hegazi, B.Sc.
(Cairo University, 1964)

A thesis submitted to the University of Leicester
for the degree of Doctor of Philosophy

FEBRUARY 1977

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

To my

Mother, Wife, Son

and

The memory of my beloved late Father

SUMMARY

During the past decade, many techniques for computer-aided circuit design have been suggested and investigated, but none have been developed to the stage where the designer is redundant. The research described herein concerns the application of one technique, the method of coefficient matching, to the synthesis of lumped, linear, passive, 3-terminal networks with no mutual inductances. The author developed a program which, with further development, could perform the entire design process with no designer interaction. Further, the author considers problems where the classical synthesis methods are unsuitable, e.g. the synthesis of non-series-parallel networks with no series-parallel equivalent.

The coefficient matching procedure is based on selecting a starting network which yields the correct polynomial structure and achieving a solution by component value adjustment and network evolution. The closer the starting network to a feasible topology, the more rapid the convergence to a solution. It is shown that the suitability of a starting network can be analysed on the basis of the information obtained from the different but equivalent forms of the admittance functions. The significance of common factors is discussed and the influence of various types of common factors on the network realization is investigated.

For cases when the initial starting network is remote from any feasible solution, sophisticated techniques allowing substantial topological modification during network evolution are required. These techniques were developed by the author on the basis of element and node addition and elimination.

A Fortran IV program has been developed by the author welding together all these aforementioned techniques for topological modification. The

program makes large topological modification automatically during the design process. The effectiveness and efficiency of these techniques and the program are illustrated by a variety of synthesis examples.

ACKNOWLEDGEMENT

I wish to gratefully acknowledge the advice and encouragement which I received from my supervisor Dr. O.P.D. Cutteridge, Reader in Engineering at the University of Leicester.

I would also like to mention the many helpful and interesting discussions I have had with Drs. D.J. Wright, A.J. Krzeczkowski, and A.R. Banerjee.

Finally, I wish to express my gratitude to the Egyptian Government and Racal-Redac Ltd. for financial support over the period 1974 to 1977.

STATEMENT

The accompanying thesis submitted for the degree of Ph.D. entitled 'An Investigation into Current Possibilities in Automated Network Design' is based on work conducted by the author in the Department of Engineering of the University of Leicester mainly during the period between July 1973 and January 1977.

All the work recorded in this thesis is original unless otherwise acknowledged in the text or by references. None of the work has been submitted for another degree in this or any other university.

Signed: O. Hegazi Date: 1st of April 1977.

CONTENTS

CHAPTER 1

INTRODUCTION

Techniques for the synthesis of electrical networks can be subdivided, in general, into two categories, namely, interpolation techniques and realization techniques. Interpolation techniques are used to find network characteristics which can be realized exactly, and which approximate the desired performance of the required network sufficiently well. Realization techniques produce explicit networks from the exactly realizable characteristics obtained by the interpolation techniques. The electrical networks considered in the present work are lumped, linear, constant parameter, passive,3-terminal networks having no mutual inductance.

Before the advent of the modern high speed digital computers, realization techniques for these networks depended on what are called 'classical techniques[1,2,3]'. These techniques are based on a set of necessary and sufficient conditions[4-7]. All such techniques rely on 'series-parallel' decomposition of the given functions (series decomposition of impedance functions, parallel decomposition of admittance functions) to successively simplify them into two or more simpler forms. Decomposition is continued until the simplified functions are recognized as impedances or admittances of some realizable networks. These simplified networks are then reassembled to construct the required network.

In 1968, Fialkow[8] showed that some feasible performance functions cannot have a series-parallel realization, which means that they cannot be realized by employing the classical methods. In addition, classical techniques take no account of the constraints on the range of element values, presence of parasitics, and sensitivity of the network performance to slight perturbations in some of the element values. The aforementioned limitations of the classical techniques show the need of a new synthesis technique to overcome these limitations.

During the past decade, due to the fast development of high speed digital computers, new synthesis techniques have been developed[9-12] which are known as 'Computer-Aided Electrical Network Design'. These techniques, using computers and employing optimization techniques, aim to manipulate large numbers of interacting variables in an iterative fashion with respect to the design objectives, independent of the network configurations.

There are two main modes of operation for using a computer, namely:

i) The 'Interactive' or 'On-line' mode in which the designer is able to interfere during run time in order to take the critical decisions.

ii) The 'Automated' or 'Batch' mode where some 'algorithm' endeavours to replace 'insight' .

In the latter mode of operation, all decisions are made automatically by implementing in the program all expected possibilities. Potentially, batch mode will require more core storage than the interactive mode but it is less hazardous. Batch mode is the mode of operation which is used in running the programs used in this research or, in other words, automated design technique has been used in the present work.

In 1964, Calahan[13] realized the possibilities of using computers in filter design. Since then, a variety of new ideas for network synthesis using Computer-Aided Design techniques have been developed[14-20]. Many of these techniques are independent of the classical restraints or the types of network which can be considered. Further, these techniques proved to be feasible in filter design[12,13], design of distributed networks[14] and many other applications[15-20]. The demand for synthesis of highly complex circuits (e.g. integrated circuits) lead to further practical design techniques such as schemes for the calculation of network sensitivities, first presented by Hachtel and Rohrer[63].

For the last seven years or so, a group of researchers has worked at the University of Leicester under the supervision of Dr. O.P.D. Cutteridge. They have considered a number of different approaches to Computer-Aided Network Design, both in interactive mode[21], and in batch mode[22-24]. One of these approaches, the 'Coefficient Matching' technique in batch mode has been widely used[22-27]. In the coefficient matching technique, the required network performance is specified in the form of a set of polynomials in the complex frequency variable p, with real positive valued coefficients. This technique was first suggested by Calahan[13,14]. Starting from an arbitrary initial network structure of the proper configuration and complexity to yield polynomials of the same order as the required network, a computer program employing optimization technique calculates the corresponding polynomial coefficients and formulates an error function which embodies the design criteria. This error function measures the difference between the performance required and that actually achieved. An optimization algorithm is used until either a complete matching between the required and the calculated performance is obtained or until some other pre-specified criteria are satisfied. These criteria give an indication of necessary modification to be made to the current network topology. After the modification takes place, and in some cases more than one modification may be necessary, optimization is repeated until finally a network is obtained which can realize the required performance. There are no constraints on the type of network configuration to be realized and any constraints on the element values can be easily included, hence, the Automated Network Design technique overcomes the well known severe limitations of the classical methods.

Further investigation of this technique would be of considerable interest to design engineers and circuit theorists. Furthermore, from a technological viewpoint, it is important to investigate how far the substantial power of modern computers can be of use for aiding or testing

a design, and how far the current stage of program development is removed from a fully automated design facility.

The development of a fully automated electrical network synthesis program, in general, is a very difficult and complex task. In the long term, the final goal of automated network design is to make it possible for the user to input the required network performance, specified in any form, directly to the program and have as the output one design, or more, which fulfils the requirements together, perhaps with the layout and integrated circuit mask. These techniques would be applicable for any type of linear network, active or passive and could include mutual inductance. In the short term, the development of an automatic program capable of synthesising RLC lumped linear networks having no mutual inductance, with no restrictions on the network configuration and regardless of the manner in which the performance is specified, is a first stepping stone towards the achievement of this goal.

In the direction of developing an automated network design program to synthesize 3-terminal lumped, linear networks with no mutual inductance, work has taken place[22-24], and still continuing, at the University of Leicester, using coefficient matching techniques. Before the beginning of the research described in this thesis, an Algol 60 program had been developed[24] with the following features:

(i) An algorithm capable of calculating, rapidly and efficiently, both the coefficients of the relevant network polynomials and their first derivatives with respect to all the possible elements in the network[28,29].

(ii) An option for selecting a variety of methods of error function representation[24,30].

(iii) A moderately successful optimization algorithm[31] using some of the special properties of multilinear functions.

(iv) Provision for modifying the network topology by adding or removing elements with the constraint that the order of the required set of polynomials be unchanged during the topological modifications.

(v) The facility of varying one common factor at a time.

The program was restricted to 3-terminal RC networks with a fixed number of nodes.

The research described in this thesis is a direct extension to the aforementioned work. The discussion in the present work is limited to 3-terminal, lumped, linear, passive networks with no mutual inductance. The network may consist of the combination of any two types of element (i.e. RL, LC or RC) or all the three types of element (RLC). Further the process of topological modification of the network has been extended to permit the addition or removal of nodes as well as elements and to allow for a change in the order of the required set of polynomials to occur whilst so doing. In other words, most of the limitations of the previous work have been overcome in respect of lumped, linear network synthesis. It is also possible, although is not considered in this thesis, to extend this technique to synthesize n-port problems.

Some of the basic principles which are of considerable importance in gaining a full understanding of the automated design technique are described in Chapter 2. In the design technique adopted in the present work, the network analysis is carried out on the basis of nodal analysis. Chapter 2, therefore describes the evaluation of network polynomials using the nodal admittance matrix. Further, the relevance of information which can be obtained from the various equivalent forms of representation

of these admittance functions, which was not known before, is pointed out. The significance of topological analysis and any common factors that may be present is also discussed. Common factors play an important role in the synthesis of electrical network. A new concept of the 'degree of connectivity' has been developed on the basis of the relationship between network polynomials and the network complexity by investigating some series-parallel networks only.

The coefficient matching technique and the optimization algorithm form the backbone of the automated design technique adopted in the present work. A review of these techniques and their important aspects are described in Chapter 3. For any 3-terminal lumped, linear RLC network, the optimization algorithm used in the present work would consistently obtain a feasible set of network element values satisfying the desired performance from any arbitrary set of starting values if the correct topology for the network was selected at the outset. In this case the solution is achieved readily without any difficulty and the automated design of the network presents no problem. However, it is unlikely that the designer would choose the correct topology for his starting network, especially if he does not know a solution or it is not possible to find one with alternative methods (e.g. classical methods). In this case some topological modifications may be necessary and should be carried out automatically. Thus the automated modifications of network topology, sometimes called automated network evolution, also form an essential part of the automated network design.

The concept of automated network evolution in the present work has been developed on the basis of the addition or removal of nodes as well as elements, and other possible variations to the design procedure, such as common factor variation, to increase the probability of satisfying the performance required. The variation of common factors and altering the order of the required set of functions might or might not accompany the

aforementioned modifications. The principles of automated network evolution are described in detail in Chapter 4.

The criteria for network modification have been implemented in a computer program written and developed by the author in FORTRAN IV. Various other modifications have been included in the Fortran program which were not present in the previously developed Algol 60 version, in order to increase the accuracy and efficiency. The Fortran program and its comparison with the Algol version have been described in detail in Chapter 5. The efficiency and reliability of the program and the feasibility of the coefficient matching technique in linear network synthesis have been demonstrated by some case studies in Chapters 6 and 7.

Chapter 6 deals with the synthesis of some two-element kind 3-terminal networks. Some non-series-parallel networks cannot have a series-parallel equivalent as Fialkow[8] pointed out. However, for such a network a very good approximate series-parallel realization may be achieved as is demonstrated by an example in Chapter 6. Another example is presented in that chapter which establishes the feasibility of the concept of the degree of connectivity (which was introduced in Chapter 2).

In Chapter 7, a case study of an RLC realization of a non-series-parallel network with no series-parallel equivalent is described. This realization employed the addition of an extra node, element addition, element removal and the variation of up to three common factors simultaneously. The RLC realization of a non-series-parallel network having no series-parallel equivalent, involving the addition of an extra node, employing automated network design has not been previously achieved. The examples described in Chapters 6 and 7 show the future potential of the automated network design technique.

Finally, the author's overall conclusions with respect to his work, including a discussion of the directions and areas of possible further investigations, are presented in Chapter 8.

CHAPTER 2

SYNOPSIS OF SOME RELEVANT THEORY OF LUMPED LINEAR ELECTRICAL NETWORKS

2.1. Introduction.

It was pointed out in Chapter 1 that coefficient matching forms the basis of the automated network design technique adopted in the present work. In this technique the coefficients of the network admittance functions, expressed as polynomials in p, the complex frequency, are matched.

Automated design is often dependent on the selection of the initial starting network. Thus automated design is greatly facilitated and simplified if the starting network is selected such that it provides a response close to that required. For a suitable starting network, the optimization algorithm will readily provide the network element values and provide a solution.

Topological analysis and the different but equivalent forms of representation for the network admittance functions are very helpful in estimating a good starting network. These are described in detail in this chapter. The significance of the equivalent forms of representation for network admittance functions was not known before.

A starting network estimated only on the basis of the above two considerations can, sometimes, be misleading. The reason for this seems to be due to the large difference in admittance level between input and output of some network polynomials (i.e. the relative magnitude of the coefficients). However, the constraints imposed by the relative magnitudes of the coefficients of the network polynomials can be overcome by using common factors. Therefore, the significance of common factors in the realization of series-parallel and non-series-parallel network is described in this chapter. Further, a new concept of a 'degree of

connectivity' has been introduced on the basis of the relationship between the common factors and the relative magnitudes of the coefficients of the network polynomials.

2.2. Nodal Analysis and the Network Admittance Functions.

The analysis of networks in the present work is by means of the characterisation of the networks by a nodal admittance matrix. The network performance employed is the three short-circuit admittance functions or the four principal polynomials of the network. To calculate these functions, consider the RLC transformerless 3-terminal network T shown in Fig.[2.1] upon a nodal basis.

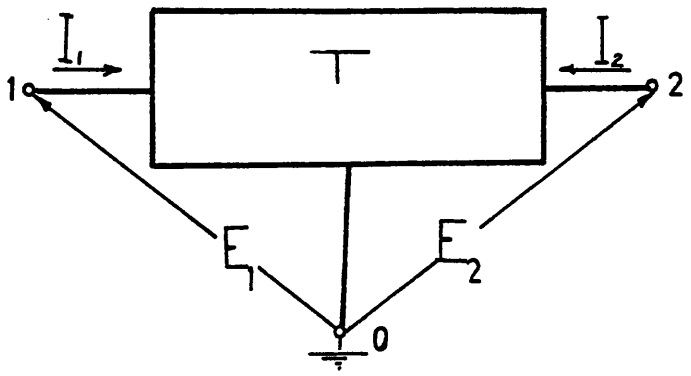


Figure [2.1] A three-terminal network

The ground terminal is taken as node 0 (reference node) and input and output terminals as nodes 1 and 2 respectively. The remaining nodes are identified so that each branch consists of at most a resistance, inductance and a capacitance in parallel. Hence the branch admittance between nodes i and j , y_{ij} $(i \neq j)$ is of the form $(ap + b + c/p)$, $a \geqslant 0$, $b \geqslant 0$ and $c \geqslant 0$, where

p is the complex frequency variable,

a is the value of capacitance in Farads,

b is the value of conductance in Siemens,

and c is the value of inverse inductance in Henrys^{-1}.

For passive, linear, lumped networks $y_{ij} = y_{ji}$. The admittance y_{ii} is defined as

$$y_{ii} = \sum_{\substack{j=o \\ i\neq j}}^{nl} y_{ij} \quad (i = 1,2,\ldots,nl), \tag{2.1}$$

where nl+1 is the total number of nodes including node 0 .

The nodal admittance matrix Y (nlxnl) , whose diagonal elements are y_{ii} and off-diagonal elements are $-y_{ij}$, yields the nodal admittance determinant Δ where

$$\Delta = \det Y \text{ (nlxnl)} . \tag{2.2}$$

The branch admittance y_{ij} , which is in the form $(ap + b + c/p)$, can be more conveniently represented by $\frac{1}{p}(ap^2 + bp + c)$. The common multiplier of all elements of a row $(\frac{1}{p})$ in the matrix Y may be taken out and y_{ij} becomes a quadratic in p with non-negative coefficients.

Writing $\Delta_{\ell k}$ for the cofactor of the ℓk'th element in Δ , and $\Delta_{\ell kpq}$ for the cofactor of the pq'th element in $\Delta_{\ell k}$, then the terminal currents I_1 and I_2 are related[1,2] to the terminal voltages E_1 and E_2 (see Fig.[2.1]) by

$$\left.\begin{aligned} I_1 &= Y_{11}E_1 + Y_{12}E_2 \\ I_2 &= Y_{21}E_1 + Y_{22}E_2 \end{aligned}\right\} \tag{2.3}$$

where

$Y_{11} = \dfrac{\Delta_{22}}{\Delta_{1122}} = \left.\dfrac{I_1}{E_1}\right|_{E_2=o}$ Input admittance when the output terminals are short-circuited.

$Y_{12} = -\dfrac{\Delta_{21}}{\Delta_{1122}} = \left.\dfrac{I_1}{E_2}\right|_{E_1=o}$ Transfer admittance when the input terminals are short-circuited.

$$Y_{21} = -\frac{\Delta_{12}}{\Delta_{1122}} = \frac{I_2}{E_1}\Big|_{E_2=o}$$ Transfer admittance when the output terminals are short-circuited.

$$Y_{22} = \frac{\Delta_{11}}{\Delta_{1122}} = \frac{I_2}{E_2}\Big|_{E_1=o}$$ Output admittance when the input terminals are short-circuited.

Because the network is composed of reciprocal elements only, $y_{ij} = y_{ji}$, $\Delta_{12} = \Delta_{21}$ and $Y_{12} = Y_{21}$.

The three admittance y-parameters Y_{11}, Y_{12} and Y_{22} or the four principal polynomials Δ_{11}, Δ_{12}, Δ_{22} and Δ_{1122} are functions of p , the complex frequency, and their coefficients are multi-linear functions of the network elements. These functions determine completely the performance of the network as far as current and voltage at nodes 0, 1 and 2 are concerned, i.e. the external performance. Any two networks that satisfy the same set of performance functions are said to be equivalent. The determinant Δ and its cofactors Δ_{11}, Δ_{12} , Δ_{22} and Δ_{1122} satisfy Jacobi's theorem[1,2], namely

$$\Delta\Delta_{1122} = \Delta_{11}\Delta_{22} - \Delta^2_{12} \qquad (2.4)$$

When the technique of coefficient matching is discussed in this thesis, the coefficients referred to will be the coefficients of the various powers of p in the four principal polynomials.

The mesh impedance matrix can be used as an alternative to the nodal admittance matrix. The latter was preferred for the following reasons:

1) The nodal method automatically yields the external equations in the form desired for a 3-terminal network. The mesh method, as straightforwardly applied, yields external equations of a two-part and not necessarily of a 3-terminal network.

2) In 3-terminal networks, the number of meshes is, in general, more than the corresponding number of nodes for any network[2], i.e. using the nodal admittance matrix represents a network in a more compact form.

3) The nodal admittance matrix is a sparse matrix[12], and using the special characteristics of sparse matrices will take less core store, speed up the calculation and increase the accuracy.

2.3. Topological Analysis of Electrical Networks.

An alternative method for evaluating the four principal polynomials Δ_{11}, Δ_{12}, Δ_{22} and Δ_{1122} is the topological analysis method. With this method, it is particularly easy to see the relationship between the network elements and the coefficients of the network polynomials when the network contains a small number of the relevant 2-trees and 3-trees. Thus it could help in selecting a suitable starting network and in developing successful criteria for network evolution. In this section some basic definitions and theorems for graph theory are introduced (for more details see Shu-Park Chan[32]).

Let T be a three-element kind 3-terminal network with N nodes plus the reference node 0 . The input and output terminals are nodes 1 and 2 respectively. Suppose T is composed of n elements. Each element is connected between two of the N+1 nodes. Each node, with the possible exception of the three external nodes, is the intersection of at least two different types of element. Nodes 0, 1 and 2 may be connected to one element only. The following theorems and definitions may be considered.

Definition 1: A tree of the network T is a connected subnetwork that contains all the nodes but does not contain any loops.

Theorem 1 : Every tree contains exactly N elements.

Definition 2: A 2-tree of a tree is a subset of N-1 elements of that tree.

Definition 3: A 3-tree of a tree is a subset of N-2 elements of that tree.

Definition 4: A tree admittance product $T_1(y)$ is the product of the admittances of the N elements of that tree.

Definition 5: A 2-tree admittance product $T_2(y)$ is the product of the admittances of the N-1 elements of that 2-tree.

Definition 6: A 3-tree admittance product $T_3(y)$ is the product of the admittances of the N-2 elements of that 3-tree.

Theorem 2 : The determinant of the admittance matrix Δ is equal to the sum of the tree admittance products of all the trees of the network.

$$\Delta = \sum_{\text{all } i} T_1^i(y) \tag{2.5}$$

Theorem 3 : The cofactor Δ_{12} of Δ is equal to the sum of the 2-tree admittance products of all the 2-trees which satisfy the following conditions

a) There is a direct path between nodes 1 and 2.

b) There is no direct path between nodes 0 and 1 .

c) There is no direct path between nodes 0 and 2 .

$$\Delta_{12} = \sum_{\text{all } i} T_2^i(y,\Delta_{12}) \tag{2.6}$$

Theorem 4: The cofactor Δ_{11} of Δ is equal to the sum of the 2-tree admittance products of all the 2-trees which satisfy the following conditions:

a) There is a direct path between nodes 1 and 2 or, there is a direct path between nodes 0 and 2, but not both.

or b) There is no direct path between nodes 0 and 1.

$$\Delta_{11} = \Delta_{12} + \sum_{\text{all } i} T_2^i(y, \Delta_{11} - \Delta_{12}) \tag{2.7}$$

Theorem 5: The cofactor Δ_{22} of Δ is equal to the sum of the 2-tree admittance products of all the 2-trees which satisfy the following conditions:

a) There is a direct path between nodes 1 and 2 or, there is a direct path between nodes 0 and 1, but not both.

or b) There is no direct path between nodes 0 and 2.

$$\Delta_{22} = \Delta_{12} + \sum_{\text{all } i} T_2^i(y, \Delta_{22} - \Delta_{12}) \tag{2.8}$$

Theorem 6: The cofactor Δ_{1122} is equal to the sum of all the 3-tree admittance products of all the 3-trees which satisfy the following conditions:

a) There is no direct path between nodes 1 and 2.

b) There is no direct path between nodes 0 and 1.

c) There is no direct path between nodes 0 and 2.

The notation used for the 2-tree and 3-tree admittance products first appeared in Krzeczkowski[24]. Although this notation is not standard, it is straight forward. The subscript indicates a tree, 2-tree or 3-tree

respectively. The y in the brackets indicates that T is a function of the admittance of the network elements.

From the aforementioned theorems and definitions we can draw the following conclusions:

1) For symmetrical networks $Y_{11} = Y_{22}$ (hence $\Delta_{11} = \Delta_{22}$) and the input and output terminals must be connected to similar elements. Further, input and output terminals must be subjected to a symmetrical topological modification, if any takes place.

2) From theorems 3,4 and 5 above, the 2-trees of Δ_{12} are included in the 2-trees of Δ_{11} and Δ_{22}. Any coefficient in the cofactors Δ_{11}, Δ_{12} and Δ_{22} is equal to the sum of the 2-tree admittance products of all the relevant 2-trees composed of the elements which can yield the corresponding power of p. Thus if the coefficients of any power of p in Δ_{11}, Δ_{12} and Δ_{22} are equal, they should correspond to identical 2-trees.

2.4. Equivalent Forms of the Admittance Functions.

In the present work the performance of the network is specified by the three short-circuit admittance parameters (Y_{11}, Y_{12} and Y_{22}) written as a ratio of two polynomials or the four principal polynomials of the network considered in Section 2.2. However there is some relevant information which can be obtained from the other equivalent forms of representation of these admittance functions. This information may help in selecting a more suitable starting network and in providing a better understanding of the automated network design technique.

For RC 3-terminal network, the three short-circuit admittance functions Y_{11}, Y_{12} and Y_{22} may be formulated as a set of poles, zeros and multiplicative constants. In this case, the zeros of the polynomials Δ_{11}, Δ_{12} and Δ_{22} become the zeros of their respective network admittances.

The zeros of the polynomial Δ_{1122} become the poles of the short-circuit admittance functions. For example Y_{11} can be written as

$$Y_{11} = \frac{\Delta_{22}}{\Delta_{1122}} = K \frac{a_n p^n + a_{n-1} p^{n-1} + \ldots\ldots + a_o}{b_m p^m + b_{m-1} p^{m-1} + \ldots\ldots + b_o}$$

$$= K \frac{(p-k_1)(p-k_2)\ldots\ldots(p-k_n)}{(p-\ell_1)(p-\ell_2)\ldots\ldots(p-\ell_m)} \qquad (2.9)$$

where

$a_i, i=0,1,\ldots,n$ are non-negative coefficients of Δ_{22}

$b_j, j=0,1,\ldots,m$ are non-negative coefficients of Δ_{1122}

$m \leqslant n$

$k_i, i=1,2,\ldots,n$ are the zeros of Y_{11}

$\ell_j, j=1,2,\ldots,m$ are the poles of Y_{11}

K is the positive valued multiplicative constant.

An alternative form of writing the admittance functions is the partial-fraction expansion, where

$$Y_{11} = \frac{\Delta_{22}}{\Delta_{1122}} = q_\infty p + q_o + \sum_{i=1}^{m} \frac{q_i p}{(p-\ell_i)}$$

$$-Y_{12} = \frac{\Delta_{12}}{\Delta_{1122}} = c_\infty p + c_o + \sum_{i=1}^{m} \frac{c_i p}{(p-\ell_i)}$$

$$Y_{22} = \frac{\Delta_{11}}{\Delta_{1122}} = d_\infty p + d_o + \sum_{i=1}^{m} \frac{d_i p}{(p-\ell_i)} \qquad (2.10)$$

where

ℓ_i $(i=1,\ldots,m)$ are the finite poles of Y_{11}, $-Y_{12}$ and Y_{22} respectively.

q_∞, c_∞ and d_∞ are the residues at the pole at zero of Y_{11}/p, $-Y_{12}/p$ and Y_{22}/p respectively.

q_o, c_o and d_o are the residues at the pole at infinity of Y_{11}, $-Y_{12}$ and Y_{22} respectively.

q_i, c_i and d_i are the residues at the finite poles of Y_{11}/p, $-Y_{12}/p$ and Y_{22}/p respectively ($q_i \geqslant 0$, $d_i \geqslant 0$ and c_i may be negative).

For RC networks the residues q_i, d_i and c_i are real. For RLC networks the residues are real if the poles are real and complex conjugate for complex-conjugate poles[33]. For RC networks in general if q, c and d are the residues of the poles in Y_{11}, $-Y_{12}$ and Y_{22} respectively, then the residues q, c and d satisfy the residue condition [1,2], namely

$$q\, d \geq c^2. \tag{2.11}$$

A special case of equation (2.11) is the compact residue condition where

$$q\, d = c^2 \ , \ c \neq 0. \tag{2.12}$$

Further, if

$$q = d = |c| \ , \tag{2.13}$$

then the residues are compact and equal.

From the topological analysis (Section 2.3) and equations (2.12) and (2.13) we can conclude the following:

1) If the residues at the pole at zero are non-zero, compact and equal, then the coefficients of p^0 in Δ_{11} , Δ_{12} and Δ_{22} must be equal.

2) If the residues at the pole at infinity are non-zero, compact and equal, the polynomials Δ_{11}, Δ_{12} and Δ_{22} must be of the same order and the coefficients of their highest power must be equal.

In any admittance function (i.e. RLC and two-element kind networks), if there is a factor in common between the numerator and the denominator, then the residue at the finite pole corresponding to this common factor is zero. Further, if the four principal polynomials Δ_{11}, Δ_{12} , Δ_{22} and Δ_{1122} have a common factor, then the residue at the finite pole corresponding to this common factor is zero in each of the three admittance functions. Of course, strictly speaking, such admittance functions do not possess poles at these points.

The role played by common factors in network synthesis is discussed in the next section.

To summarize, the equivalent forms of representation of the network admittance functions can be helpful in estimating a good starting network, if the starting network is selected such that:

1) It yields the same spread of powers as the required set of polynomials.

2) It yields the same number of poles and zeros.

3) It yields the same number of residues. The relevant residues in the calculated and required set of functions should have the same sign.

Example 1

Suppose that the network shown in Fig.[2.2] is selected to synthesize the admittance functions of equations (2.14) (originally given by Lucal[34])

$$\left.\begin{aligned} Y_{11} = \frac{\Delta_{22}}{\Delta_{1122}} &= \frac{6p^4+343p^3+1092p^2+773p+6}{6(p^3+6p^2+11p+6)} \\ &= p + \frac{1}{6} + \frac{p}{p+1} + \frac{15p}{p+2} + \frac{35p}{p+3} \\ -Y_{12} = \frac{\Delta_{12}}{\Delta_{1122}} &= \frac{p^4+p^3+2p^2+p+1}{(p^3+6p^2+11p+6)} \\ &= p + \frac{1}{6} - \frac{p}{p+1} + \frac{15p/2}{p+2} - \frac{35p/3}{p+3} \\ Y_{22} = \frac{\Delta_{11}}{\Delta_{1122}} &= \frac{36p^4+533p^3+1572p^2+1183p+36}{36(p^3+6p^2+11p+6)} \\ &= p + \frac{1}{6} + \frac{p}{p+1} + \frac{15p/4}{p+2} + \frac{35p/9}{p+3} \end{aligned}\right\} \quad (2.14)$$

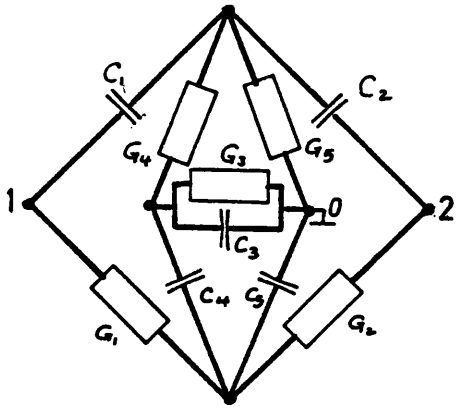


Figure [2.2] Starting network to synthesize equation (2.14)

Although the network shown in Fig.[2.2] yields the required spread of powers, it cannot yield residues of the same sign as those in $-Y_{12}$, for any range of positive element values (see Section 6.5). Further, for starting element values all equal to unity, there is a common factor (p+1) in Δ_{11}, Δ_{12} , Δ_{22} and Δ_{1122} . This information cannot be obtained by considering the four principal polynomials in the rational function form only.

The equivalent forms representing the short-circuit admittance functions can be of a great help in developing the synthesis technique. In case the technique adopted for network synthesis fails to achieve a realization, the equivalent forms may help in investigating the cause of failure (see Section 6.5). Moreover, the equivalent forms might give a better idea of how far the current realization is removed from a final solution.

2.5. Common Factors.

In Chapter 1 it was pointed out that the aim of the coefficient matching technique is the development of a network and finding suitable element values which realize the required set of network polynomials exactly to within a multiplicative constant. In Section 2.2, it was pointed out that the network performance employed in the present work is

the short-circuit admittance functions Y_{11}, $-Y_{12}$ and Y_{22} where

$$Y_{11} = \frac{\Delta_{22}}{\Delta_{1122}}, \quad -Y_{12} = \frac{\Delta_{12}}{\Delta_{1122}} \text{ and } Y_{22} = \frac{\Delta_{11}}{\Delta_{1122}} \tag{2.15}$$

A more general form to write equation (2.15) is

$$Y_{11} = \frac{K\Delta_{22}}{K\Delta_{1122}}, \quad -Y_{12} = \frac{K\Delta_{12}}{K\Delta_{1122}} \text{ and } Y_{22} = \frac{K\Delta_{11}}{K\Delta_{1122}} \tag{2.16}$$

where K is a multiplicative factor. If K is non-zero, positive and finite, it will not affect the properties of the admittance functions in any way. Similarly, the introduction of common factors in the numerator and denominator of the admittance functions, will not affect the external characteristics. For example, if common factors of the form $(p+\alpha_1)(p+\alpha_2)....(p+\alpha_k)$ are introduced in equation (2.16) then,

$$Y_{11} = \frac{\Delta_{22}}{\Delta_{1122}} = \frac{K\Delta_{22}(p+\alpha_1)(p+\alpha_2).....(p+\alpha_k)}{K\Delta_{1122}(p+\alpha_1)(p+\alpha_2).....(p+\alpha_k)}, \text{ etc.} \tag{2.17}$$

Equations (2.15), (2.16) and (2.17) are completely equivalent. However an RC network yielding the functions of equation (2.17) would have k more nodes than a network yielding equation (2.15) or (2.16). Classical methods of network synthesis often generate many excess common factors before the required functions are realized.

Elementary network theory requires that at least one common factor be presented in order to synthesize the short-circuit admittance functions (originally given by Fialkow[8])

$$\left.\begin{aligned} Y_{11} = Y_{12} &= \frac{(p+1)(1197p^3+56613.14p^2+2836 8.584p+191.184)}{(p+1)(800000p^2+408000p+3840)} \\ -Y_{12} &= \frac{(p+1)(3p^3-1.14p^2+197.176p+77.616)}{(p+1)(800000p^2+408000p+3840)} \end{aligned}\right\} \tag{2.18}$$

The reason for this is the presence of the negative coefficient in the numerator of the transfer function. In this case common factors play a decisive role[8,35].

For some examples, such as the set of admittance functions given by equations (2.14), theory gives no obvious reason or guide to the number, if any, of excess factors required. The best realization for this set of functions published by Cutteridge[36] had fourteen elements and two common factors. Later, Hansen and Wanet[37] presented an LC realization whose RC equivalent has two linear common factors, and used one less network element than did Cutteridge's realization. The reason that this set of functions required common factors seems to be due to the large difference in the admittance level between the input and output. The relationship between the coefficient of the network polynomials and the required number of excess factors is established by the author and the concept of a 'degree of connectivity' is introduced later on in this section. Thus common factors play an important role in the synthesis of electrical networks in general[35]. In order to understand this role, let us consider the effect of introducing a common factor in a set of four principal polynomials Δ_{11} , Δ_{12}, Δ_{22} and Δ_{1122} and their corresponding network T .

$$\left.\begin{aligned}
\Delta_{11} &= a_o+a_1p+a_2p^2+\ldots\ldots+a_np^n \quad (a_i\geq 0, i=1,\ldots,n)\\
\Delta_{12} &= b_o+b_1p+b_2p^2+\ldots\ldots+b_np^n \quad (b_i \text{ may be negative, } i=1,\ldots,n)\\
\Delta_{22} &= c_o+c_1p+c_2p^2+\ldots\ldots+c_np^n \quad (c_i\geq 0, i=1,\ldots,n)\\
\Delta_{1122} &= d_o+d_1p+d_2p^2+\ldots\ldots+d_mp^m \quad (d_i\geq 0, i=1,\ldots,m, m\leq n)
\end{aligned}\right\} \qquad (2.19)$$

If a common factor of the form $(p+\alpha)$, $\alpha \geq 0$ is introduced in equations (2.19) then,

$$\left.\begin{aligned}
\hat{\Delta}_{11} &= (p+\alpha)\Delta_{11} = a_o\alpha+(a_o+\alpha a_1)p+\ldots\ldots+(a_{n-1}+\alpha a_n)p^{n+1}\\
\hat{\Delta}_{12} &= (p+\alpha)\Delta_{12} = b_o\alpha+(b_o+\alpha b_1)p+\ldots\ldots+(b_{n-1}+\alpha b_n)p^{n+1}\\
\hat{\Delta}_{22} &= (p+\alpha)\Delta_{22} = c_o\alpha+(c_o+\alpha c_1)p+\ldots\ldots+(c_{n-1}+\alpha c_n)p^{n+1}\\
\hat{\Delta}_{1122} &= (p+\alpha)\Delta_{1122} = d_o\alpha+(d_o+\alpha d_1)p+\ldots\ldots+(d_{m-1}+d_m)p^{m+1}.
\end{aligned}\right\} \qquad (2.20)$$

The introduction of a common factor has the following effects:

A] Analytical Effects

i) A change in the value of each coefficient. Thus in the case of negative coefficient (equations (2.18)) a common factor of a proper value [27,35] $\left(\frac{1.14}{3} < \alpha < \frac{191.176}{1.14}\right)$ will mask the presence of the negative coefficient.

ii) A change in the ratio between the relevant coefficients in the different cofactors, e.g. consider the relevant coefficient in Δ_{11} and Δ_{12} in equations (2.19) and (2.20), then

$$\frac{a_o}{b_o} \rightarrow \frac{\alpha a_o}{\alpha b_o} = \frac{a_o}{b_o}$$

$$\frac{a_1}{b_1} \rightarrow \frac{a_o + \alpha a_1}{b_o + b_1} \quad .$$

Further, the ratio between the coefficients of the same cofactor is also changed e.g. for Δ_{11} and Δ_{11}

$$\frac{a_o}{a_1} \rightarrow \frac{\alpha a_o}{a_o + \alpha a_1}$$

$$\frac{a_{n-1}}{a_n} \rightarrow \frac{a_{n-2} + \alpha a_{n-1}}{a_{n-1} + \alpha a_n}$$

iii) The number of the non-zero coefficients has increased by four (one in each cofactor).

iv) No change in the external characteristics.

B] Topological Effects.

To multiply the four principal polynomials by the same common factor necessitates the addition of an extra node, thus increasing the number of possible network elements. This in turn will result in an increase in the number of alternative topologies to choose from. Further it will result in an increase in the number of trees, 2-trees and 3-trees in

general. Thus, the possibility of sharing fewer elements between the various 2-trees and 3-trees is increased.

To summarize, the introduction of a common factor produces the following:

1) Changes in the values and the number of the non-zero coefficients.

2) Changes in the ratio between the relevant coefficients in the different cofactors as well as the ratio between the coefficients of the same cofactor.

3) An increase in the number of nodes, thus an increase in the number of possible alternative topologies (i.e. possible network elements).

4) As a result of 3) , the number of possible 2-trees and 3-trees is increased.

5) Since the coefficients are the admittance products of the relevant 2-trees and 3-trees, thus the possible difference in the relative magnitude of the coefficients can be increased.

6) As a result of 4) and 5) network polynomials with large difference in the admittance level between the input and output will probably not be realizable with a minimum number of nodes. However, a realization may exist if one or more common factors (hence nodes) are introduced.

The concept of the 'degree of connectivity' was developed based upon the above items. In order to define the degree of connectivity consider the following example.

Example 2

Suppose that equations (2.19) is to be realized by an RC network. In order to generate network polynomials of the same degree as the required polynomials, the necessary minimum number of nodes, in addition to the reference node, is equal to $(m+2)$, where m is the maximum order of p in Δ_{1122}.

For a network with a minimum number of nodes, the number of alternative topologies which can yield the required spread of powers is finite. Thus, the number of 2-trees and 3-trees is finite. The set of required network polynomials often imposes some constraints on the required topology (e.g. symmetry, compactness, etc...). Hence a network with a minimum number of nodes may never yield the required topology. However, these constraints may be overcome if the number of nodes is increased, as the number of possible topologies will increase.

Let us consider that the degree of connectivity is some measure of the number of different topologies which are possible for a network with a given number of nodes. Then a network composed of the minimum number of nodes has a low degree of connectivity. A network which is composed of more than the minimum number of nodes required is a network with a higher degree of connectivity. The degree of connectivity is to be increased (increase the number of extra nodes) until the constraints imposed by the required set of network polynomials are satisfied.

More discussion is given in Appendix A and an example is given in Section 6.5 to establish the existence of the degree of connectivity.

2.7. Summary.

In the introduction of this chapter, the main aim was explained as follows. To use automated network design techniques efficiently, all the possible information from the given set of network polynomials should be considered. The investigation was valuable as the following new

contributions were achieved.

1) The significance of considering the various different but equivalent forms of admittance functions was pointed out. This is very helpful in estimating a good starting network. Further it might help in explaining the odd behaviour of the optimization algorithm.

2) The role played by the common factors in the network synthesis was established on the basis of the relationship between the coefficients of the network polynomials and the required degree of the network complexity. Further a new concept of the degree of connectivity was introduced to justify the existence of such a relation.

To summarize the main results of this chapter, some of which are original, we can say the following.

The maximum spread of powers of p, the complex frequency, within the set of four principal polynomials will decide the minimum number of nodes of the required network. The configuration of a suitable starting network is estimated on the basis of the information obtained from the various equivalent forms of the admittance functions and the topological analysis of a variety of trial networks intuitively conceived as possibly yielding network polynomials of the correct order. Any difficulties may hopefully be overcome by using common factors. Common factors play a decisive role in the synthesis of non-series-parallel networks possessing no series-parallel equivalent.

CHAPTER 3

REVIEW OF THE COEFFICIENT MATCHING TECHNIQUE AND THE OPTIMIZATION ALGORITHM USED

3.1. Introduction.

In order to introduce a network design problem to the computer, to solve it automatically, i.e. without human intervention, it is necessary that the desired network specifications should be expressed in analytical form. A simple analytical form is always preferable to a complicated form as the complexity of the algorithm to solve the network synthesis problem is often in direct proportion to the method of specifying the required response. One of the methods for specifying the required response accurately, and fairly simple to program, is the rational function form in p.

In some synthesis problems, the specification may not be given analytically as a rational function but may be presented in the form of a curve or a table etc. In this case, it is convenient to transform the required specification into some other equivalent specification, where the latter is simpler to handle. It was demonstrated that a synthesis problem specified in the time domain may be transformed to the frequency domain without a loss of generality[38]. Furthermore, network specification given in the form of frequency response may be approximated, to a reasonable degree of accuracy, by a rational function in p, the complex frequency[39]. Although such transformations may not always be possible, or easy, they are very important in order to employ computers automatically.

To solve a synthesis problem starting from the rational function form, e.g. the short-circuit admittance functions, there are two methods:-

1) By comparing the zeros of the polynomials of the desired and calculated performance functions (pole-zero matching).

2) By comparing the coefficients of the polynomials of the desired and calculated performance functions (coefficient matching).

Although pole-zero matching and coefficient matching techniques are very similar (instead of matching the roots of the polynomials, we match their coefficients), the pole-zero matching technique would suffer from involving additional computation[21]. The coefficient matching technique is the technique adopted in the research described in this thesis and is considered in some detail in this chapter.

The coefficient matching technique was first introduced by Calahan[14,15] who acknowedged that its general features were suggested to him by Orchard. This technique assumes that the desired network performance functions are given in the form of polynomials in p, whose coefficients are to be matched exactly by some network function to be designed. While not all the performance functions of the synthesis problems are given in rational function form, they can often be transferred to that form within a reasonable degree of accuracy. Calahan[14,15] and Temes[9,15] have stated that, using coefficient matching techniques in electrical network synthesis is not only convenient and efficient but also "versatile" and has "extraordinary convergence properties". The power of the technique, which is sufficiently demonstrated further on in this thesis, is a further justification for formulating the synthesis problem in such a form.

In the coefficient matching technique, a reliable optimization algorithm is to be employed in order to match the required and calculated network performances iteratively. The current state of the art in optimization techniques is such that it is not always possible to solve all the complicated and highly specialized problems that can occur in network synthesis. Practically all currently available optimization techniques only guarantee the convergence to a 'local minimum' not necessarily to a

'global minimum' (exact solution).

A designer planning to employ optimization techniques in network synthesis is faced with two alternatives:

1) To adapt the currently available optimization algorithms, making maximum use of the special characteristics of the problem under consideration, to greatly improve the existing techniques[40-42] (e.g. gradient descent, quasi-Newton, etc.).

2) To develop a new optimization technique to solve the problem. Some of these techniques which are yet in the developing stages are adaptive random techniques[43-45], heuristic methods[46-48], analytic approaches[48-51], etc.... .

Since the development of a new optimization technique is a complete research project by itself and is not the theme of this research, the alternative employed is that which is first discussed.

In Calahan's original formulation of coefficient matching, the coefficients of the desired and calculated polynomials were matched by constructing a simple linear error function to embody the design criteria. Later on, a more satisfactory method for error representation was developed[24,27]. Further, a variation was introduced for adjusting the value of "normalising variable", which increased the range and the reliability of the method. These methods are summarized and discussed in this chapter.

For a realization of a linear passive electrical network to be feasible, it is necessary that all the element values obtained be non-negative. Methods of constraining the network element values to the positive domain are described and one such technique, the use of logarithmic transformation, which is employed in this research, is compared with other relevant methods.

3.2. The Coefficient Matching Technique.

3.2.1. Introduction.

The coefficient matching technique is the technique employed in this research which was first described by Calahan[14,15]. His main supposition was that the required network performance given in the form of a polynomial T(p) say, in p the complex frequency. In Section 2.2, the analysis of the network by means of the characterisation of the network in nodal admittance matrix form to yield the network performance in the form of polynomials in p was given. In this section the coefficient matching technique used is discussed.

For any two-element kind network with nl nodes, plus the reference node, node 0 , while there are nl(nl+1) maximum possible elements, the three y-parameters, if regarded as rational functions of the network polynomials, will only provide the designer with up to (4nl-2) independent pieces of information[21] (3(nl-1) zeros, (nl-2) poles and 3 normalizing factors). In general, we will have more elements (variables) than independent coefficients (equations); in other words an underdefined system, a case where it is not easy to predict how an optimization algorithm will react. The situation becomes more difficult as nl , the number of nodes, increases or three-element kind networks are considered (number of possible elements is 3nl(nl+1)/2).

In Calahan's original formulation of coefficient matching, only the coefficients of a single specified rational function were considered. He circumvented the constraint of the underdefined problem by fixing the values of sufficient number of elements to ensure a finite number of solutions. Later, a more general approach was suggested by Cutteridge[26,31]. He pointed out that simultaneous matching of the coefficients of the three short-circuit admittance functions could be attempted. Furthermore, by fixing the values of some of the maximum possible number of elements at zero and varying the rest of the elements, n elements say, such that

$n \leq m$ where m is the number of independent coefficients, the underdefined problem transferred to an overdefined $(n < m)$ or defined problem $(n = m)$. The selection of such a set of n elements and the alteration of the elements of this subset are to be discussed in Chapter 4.

The general features of the coefficient matching technique can be written as follows:

Let $x_j (j=1,2,\ldots,n)$ be the current values of the network elements.

Let $a_i (i=1,2,\ldots,m)$ be the given values of the coefficients of the polynomials to be matched.

Let $c_i (i=1,2,\ldots,m)$ be the corresponding values of the coefficients a_i actually achieved for given element values x_j.

Let the desired function be given by[9,12]

$$T_d(p) = \frac{\sum_{j=\ell+1}^{m} a_j p^{j-\ell-1}}{\sum_{i=1}^{\ell} a_i p^{i-1}} . \qquad (3.1)$$

In Section 2.2 it was shown that the coefficients of the network polynomials are multilinear functions of the network elements, hence for a starting network of the proper configuration and complexity, the calculated network function has the same form

$$T_c(p) = \frac{\sum_{j=\ell+1}^{m} c_j(\underline{x}) p^{j-\ell-1}}{\sum_{i=1}^{\ell} c_i(\underline{x}) p^{i-1}} . \qquad (3.2)$$

In general, the coefficients $c_i(\underline{x})$ will not be equal to a_i. Hence m error functions can be constructed by matching the required coefficients a_i to within a constant multiplier with those obtained by calculating c_i for the current element values x_j. These m error functions will be zero valued when an exact matching is achieved i.e.

when

$$c_i = Ka_i \tag{3.3}$$

where K is the constant multiplier needed since, in general, only proportionality (rather than equality) can be achieved between the a_i and c_i. The problem has now been transformed to solving m simultaneous equations in n unknowns, and can be solved as an optimization problem.

3.2.2. Calculation of Coefficients and their First Derivatives.

In order to achieve a solution by matching the values of the given and calculated set of coefficients, the values of the calculated coefficients are to be calculated after each iterative alteration in the element values. Hence, the efficiency of the coefficient matching technique depends on the speed of evaluation of the coefficient values and also, if required, their first derivatives with respect to the element values.

From the several computer algorithms developed during the last few years[22,23,28,29,52,64] for calculating coefficient values, the technique used in this research was that developed by Cutteridge and Di Mambro[22,23,28,29]. It is rapid and efficient. Because this technique has already been well documented[22,23], only the special features of that technique are considered in this section. The main theme is to evaluate the coefficients of the four principal polynomials Δ_{11}, Δ_{12}, Δ_{22} and Δ_{1122}, which are cofactors of the determinant Δ of the nodal admittance matrix (see Section 2.2), and also, when required, their first derivatives with respect to the network element values.

As a first step, the admittance determinant Δ for the specified network topology is created with the current element values. The elements of Δ will be quadratics in p divided by p for RLC networks.

Suppose the coefficients of one of the principal network polynomials, Δ_{11} say, are to be calculated. If $\Delta_{11}(p)$ of order r, is evaluated at $r+1$ values of $p(p=p_1,p_2,\ldots,p_{r+1})$ respectively then

$$\begin{bmatrix} 1 & p_1 & \cdots & \cdots & \cdots & p_1^r \\ 1 & p_2 & & & & p_2^r \\ \vdots & \vdots & & & & \vdots \\ 1 & p_{r+1} & \cdots & \cdots & \cdots & p_{r+1}^r \end{bmatrix} \begin{bmatrix} c_o \\ c_1 \\ \vdots \\ c_r \end{bmatrix} = \begin{bmatrix} \Delta_{11}(p_1) \\ \Delta_{11}(p_2) \\ \vdots \\ \Delta_{11}(p_{r+1}) \end{bmatrix} \qquad (3.4)$$

where $\Delta_{11}(p) = c_o p^o + c_1 p^1 + \ldots + c_r p^r$.

Since $p_1,p_2,\ldots,p_{r+1}$ are known, the values of $c_o,c_1,c_2,\ldots c_r$ can be obtained by multiplying both sides of equation (3.4) by the inverse of the Vandermonde matrix, namely the square matrix of order $((r+1) \times (r+1))$ on the left hand side of equation (3.4).

Because the derivative of a determinant with respect to one of its elements is equal to the corresponding cofactor and, the elements of Δ_{11} are functions of the network element values, then if element x_j is connected between nodes q and zero then

$$\partial(\Delta_{11}(p_i))/\partial x_j = \Delta_{11qq}(p_i) \qquad (3.5)$$

and if element x_j is connected between nodes q and t, then

$$\partial(\Delta_{11}(p_i))/\partial x_j = \Delta_{11qq}(p_i) + \Delta_{11tt}(p_i) - 2\Delta_{11qt}(p_i) \qquad (3.6)$$

Hence, from equations (3.5) and (3.6) and (3.4), if the values of $\partial(\Delta_{11}(p_i))/\partial x_j$ are known, where $\partial(\Delta_{11}(p_i))/\partial x_j$ is the derivative of Δ_{11} with respect to element x_j evaluated at $p=p_i$, then

$$\frac{\partial c_o}{\partial x_j}, \frac{\partial c_1}{\partial x}, \ldots\ldots, \frac{\partial c_r}{\partial x_j}$$

can be obtained by replacing c_i by $\frac{\partial c_i}{\partial x_j}$ and $\Delta_{11}(p_i)$ by $\partial(\Delta_{11}(p_i))/\partial x_j$ respectively in equation (3.4) and evaluating as for $c_o, c_1, \ldots, c_r$.

The evaluation of the coefficients and their derivatives can be performed more rapidly if the special properties of the principal polynomials are considered. Δ_{11}, Δ_{12}, Δ_{22} and Δ_{1122} can be evaluated simultaneously as they only differ by a row and a column, the rest of their elements being identical. The method was developed by Cutteridge and Di Mambro[28,29] and used Gaussian elimination.

If $\Delta_{11}(p_i)$ is the determinant of the matrix $M_{11}(p=p_i)$, then the inverse of M_{11}, M_{11}^{-1} equals the adjoint matrix of M_{11} divided by $\Delta_{11}(p_i)$, and $\mathrm{Adj}(M_{11}(p=p_i))$ is the transpose of the matrix of cofactors of $M_{11}(p_i)$. Hence, by evaluating $c_o, c_1, \ldots, c_r$ and their first derivatives $\frac{\partial c_o}{\partial x_j}, \frac{\partial c_i}{\partial x_j}, \ldots\ldots, \frac{\partial c_r}{\partial x_j}$ all the required information can be obtained. For more details see Di Mambro[22,23].

3.3. Error Functions.

Once the current coefficients are evaluated, as discussed in the previous section, the formulation of an error function which embodies the design criteria must be considered. This function will be a measure of the error between actual and desired network response, comparing the corresponding coefficients, one by one. The optimum network (a solution) is the one for which the error function is minimized to zero value.

When using coefficient matching technique, given the four principal polynomials Δ_{11}, Δ_{12}, Δ_{22} and Δ_{1122}, it is necessary to find a network which yields a set of polynomials such that the coefficients of these polynomials exactly match the corresponding coefficients of the given network polynomials.

Let m be the total number of coefficients in the network polynomials to be matched.

Let n be the number of elements needed to construct a starting network of proper configuration and complexity. The element values of this network are to be adjusted iteratively until the specified network polynomials are exactly matched.

Let $a_i (i=1,2,\ldots,m)$ be the given values of the coefficients of the polynomials to be matched.

Let $c_i (i=1,2,\ldots,m)$ be the corresponding values of the coefficients a_i actually achieved for given element values, x_j .

Let K be the multiplicative constant.

If $f_i (i=1,2,\ldots,m)$ is an individual error function measuring the difference between the current value of the coefficient c_i and the corresponding value of the coefficient a_i then, at a solution,

$$f_i = 0 \quad \text{if} \quad c_i = Ka_i \ (i=1,2,\ldots,m). \tag{3.7}$$

It is clear that c_i and f_i are functions of the element values x_j, i.e.

$$\left.\begin{aligned} c_i &= \phi(x_1,x_2,\ldots,x_n) \\ f_i &= \psi(K,a_i,x_1,x_2,\ldots,x_n) \end{aligned}\right\} \ (i=1,2,\ldots,m) \tag{3.8}$$

From the above, given the required network polynomials coefficients (a_i) and the coefficients generated by the suggested network (c_i) , a set of simultaneous equations (f_i) can be obtained by comparison between the corresponding coefficients. This problem has a solution if a set of non-negative element values $\underline{x}$ is reached such that the error function vector $\underline{f} = \underline{0}$.

The sequence of values of a given $f_i (i=1,\ldots,m)$ obtained during the iterative process used to solve $\underline{f} = \underline{0}$, need not necessarily be monotonically decreasing. However a single overall objective function

F can be constructed such that the value F will decrease monotonically and this function can be used to monitor the general progress towards a solution.

Throughout the work described in this thesis, minimization of a least-squares type of overall error function is used as it is particularly well suited to network optimization[19]. The least-squares overall function can be written as

$$F = \sum_{i=1}^{m} f_i^2 \tag{3.9}$$

which is a special case of the more general least p'th performance function[9]

$$F = \sum_{\text{all } i} |f_i W_i|^p \tag{3.10}$$

where W is a weighting function and $|\ |$ represents the absolute value. Temes and Zai[9,53] claim that values of p from 4 to 10 cause equation (3.10) to be an increasingly good engineering approximation to a minimax criterion, which is the limiting case of equation (3.10) when $p \to \infty$. Minimax approximations were not considered in this work because:

1) A solution will be achieved by solving for $\underline{f} = \underline{0}$, i.e. the notion of non-linear equation solving is used rather than that of function minimization.

2) Derivative information when using minimax errors is discontinuous, hence, the use of gradient optimization methods would not be possible.

The importance of this alternative is not considered in this research.

If $F = \underline{f}.\underline{f} = \sum_{i=1}^{m} f_i^2$ and, at a solution $\underline{f} = \underline{0}$, then $F = 0$. There are several ways[24,54] in which the individual error function f_i can be formulated.

1) Absolute error function:

$$f_i = c_i - Ka_i \tag{3.11}$$

It is unacceptable for c_i, K or $a_i >> 0$ and it has the trivial solution $f_i = 0$ if $c_i = 0$ and $K = 0$.

2) Relative error function:

$$f_i = \frac{c_i}{Ka_i} - 1 \tag{3.12}$$

It is unacceptable for K or $a_i \to 0$ and it is insensitive to large changes in c_i when $c_i \leqslant Ka_i$ (see Krzeczkowski[24]).

3) Double sided relative error:

$$f_i = \frac{c_i}{Ka_i} - \frac{Ka_i}{c_i} \tag{3.13}$$

It is a form which was suggested by Cutteridge[30,54] and which proved to be very suitable for coefficient matching technique when c_i is a non-negative multi-linear function of the non-negative elements of the network as $-\infty \leqslant f_i \leqslant +\infty$ for $0 \leqslant c_i \leqslant +\infty$. f_i will vanish if, and only if, $c_i = Ka_i$. There is no possibility of any trivial solution because $f_i \to \pm\infty$ when c_i or $K \to 0$.

The formula of equation (3.13) was that chosen for this research, on the basis of compromise. Although it increases the degree of non-linearity of any space and could introduce a greater number of local minima, it represents the error function more efficiently and will be less affected by extreme values of x_i (which will affect the values of c_i).

A study of the most efficient way to implement the multiplicative constant K in the error formula, when using coefficient matching technique, was carried out at Leicester[24,30]. Since the final value of K at a

solution cannot be predetermined, the problem is to choose the value of K so that it can speed up the rate of convergence and to increase the range in which x_j can be considered. Three different formulations of K have been considered:

1) K is varied in the same manner as the network element values $\underline{x}$. In this case there are $(n+1)$ variables.

2) K is represented as a multi-linear function of the network element values $\underline{x}$, i.e. considered as an extra coefficient c_{m+1}.

3) K is replaced by one of the coefficients of the given set of polynomials, a^* say, and the rest of the coefficients normalized with respect to it. In this latter case equations (3.11), (3.12) and (3.13) are

$$f_i = \frac{c_i}{c_i^*} - \frac{a_i}{a^*} \tag{3.14}$$

$$f_i = \left(\frac{c_i/c^*}{a_i/a_i^*}\right) - 1 \tag{3.15}$$

$$f_i = \left(\frac{c_i/c^*}{a_i/a^*}\right) - \left(\frac{a_i/a^*}{c_i/c^*}\right) \tag{3.16}$$

In case 3 above (equation (3.16)), although the number of variables has been reduced by one, the complexity of the functions has been increased from a set of multi-linear functions c_i, to a set of ratios of multi-linear functions c_i/c^*, in the network element values x_j. For cases 1 and 2 above, it has been illustrated[24] that the rate of convergence of the optimization algorithm will improve if K is calculated in an analytical manner.

Writing

$$F = \sum_{i=1}^{m} f_i^2 = \sum_{i=1}^{m} \left(\frac{c_i}{Ka_i} - \frac{Ka_i}{c_i}\right)^2$$

then

$$\frac{\partial F}{\partial K} = 2 \sum_{i=1}^{m} f_i \frac{\partial f_i}{\partial K} = 2 \sum_{i=1}^{m} f_i \left(- \frac{c_i}{K^2 a_i} - \frac{a_i}{c_i} \right)$$

$$= -2 \sum_{i=1}^{m} \frac{f_i}{K} \left(\frac{c_i}{Ka_i} + \frac{Ka_i}{c_i} \right)$$

$$= -2 \sum_{i=1}^{m} \frac{1}{K} \left(\frac{c_i}{Ka_i} - \frac{Ka_i}{c_i} \right) \left(\frac{c_i}{Ka_i} + \frac{Ka_i}{c_i} \right)$$

$$= -2 \sum_{i=1}^{m} \frac{1}{K} \left[\left(\frac{c_i}{Ka_i} \right)^2 - \left(\frac{Ka_i}{c_i} \right)^2 \right] \qquad (3.17)$$

For an electrical network with passive elements, the value of K must be positive and real. Whence the only feasible value for K to realize equation (3.17) is

$$K = \left[\sum_{i=1}^{m} \left(\frac{c_i}{a_i} \right)^2 \Bigg/ \sum_{i=1}^{m} \left(\frac{a_i}{c_i} \right)^2 \right]^{\frac{1}{4}} \qquad (3.18)$$

Substituting the value of K obtained from equation (3.18) into F, then

$$F_{min.} = 2 \left[\sum_{i=1}^{m} \left(\frac{c_i}{a_i} \right)^2 \Bigg/ \sum_{i=1}^{m} \left(\frac{a_i}{c_i} \right)^2 \right]^{\frac{1}{2}} - 2m \, . \qquad (3.19)$$

The strategy for using the information about K so obtained, which proved very successful[24], is to use the fomula for K whenever $\underline{f}$ or $F(\underline{x},K)$ are evaluated, updating K to the new optimum value.

3.4. The Optimization Algorithm used in the Coefficient Matching Technique.

3.4.1. Introduction.

In the previous section it was shown how the synthesis problem of finding the network element values to realize a given set of principal network polynomials was transformed to the equivalent analytical problem of finding the values of $\underline{x}(x_j \geqslant 0)$ which solve the equation $\underline{f} = \underline{0}$ and

hence minimize the overall error function F where $F = \sum_{i=1}^{m} f_i^2$.
Numerous constrained and unconstrained optimization techniques have been developed and compared to find an algorithm which is most efficient (e.g. see references 40, 41, 42). Unfortunately, the current state of the art in optimization techniques is still not capable of dealing with complicated and highly specialised problems such as the solution of a multi-modal, multi-dimension overall error function F representing the performance of an RLC network. Further, these techniques can only guarantee to find a local minimum, not necessarily a global minimum, if one exists.

In the meantime, it was shown[24] that any algorithm used in coefficient matching is problem dependent and the form of the individual error function and the overall error function to be used is very important. This fact was proved to be valid within the experience gained by the author with several well known optimization algorithms. Hence, in the rest of this section, the main features of a two-part Conjugate Gradient and Gauss Newton algorithm suggested by Cutteridge[31], which is employed in the present work, is introduced. A further justification for employing this algorithm is given in Chapter 6 in the form of a case study comparing the results obtained with those obtained by other algorithms.

The main strategy in the two-part algorithm is as follows. The Gauss Newton method produces rapid convergence towards a solution of a set of non-linear simultaneous equations provided a sufficiently near starting point is available. Further, a linear search is constructed along the vector of corrections to obtain a minimum value of the overall error function F . In general, the starting network element values are not near to the solution values required. To overcome this difficulty, the Gauss Newton algorithm is preceded by a preliminary section in which a single function, the overall error function F , is minimized by the Conjugate Gradients method. Far from a solution, the Conjugate Gradients method is basically relatively stable in contrast to the Gauss Newton algorithm.

Restating the notation used:

$\underline{x}$ is the vector of current element values $x_j, j=1,2,\ldots,n$.

$\underline{f}$ is the vector of individual error functions $f_i, i=1,2,\ldots,m$. $f_i=0$ if $c_i=Ka_i$.

F is the overall error function $F = \sum_{i=1}^{m} f_i^2$, F=0 at a solution ($f_i=0$).

Let J(mxn) be the Jacobian, the matrix of the first derivatives of the function f_i with respect to the network element values x_j such that the ij^{th} element is $J_{ij} = \frac{\partial f_i}{\partial x_j}$. J^{-1} and J^T are the inverse and the transpose of the matrix J respectively.

Let $\underline{g} = 2\, J^T . \underline{f} = \frac{\partial F(\underline{x})}{\partial x_j}$, j=1,2,...,n be the Gradient vector.

3.4.2. The Conjugate Gradients Algorithm.

The Conjugate Gradients algorithm used in this research is a Fortran IV translation of the Algol version published by Fletcher and Reeves[55].

Let $\underline{x}_o$ be a given starting point in the space variables.

Let t denote the number of the current iteration starting with t=1 .

The iteration requires the gradient

$$\underline{g}_t = \underline{g}(x_t). \tag{3.20}$$

the search direction $\underline{d}_t$ is defined by

$$\underline{d}_t = -\underline{g}_t \ , \ t=1,n+1,2n+1,\ldots \tag{3.21}$$

$$\underline{d}_t = -\underline{g}_t + \beta_t \underline{d}_{t-1}, \ t=2,3,\ldots,n;n+2,n+3,\ldots,2n \tag{3.22}$$

where β_t is a scalar quantity

$$\beta_t = \frac{\underline{g}_t \cdot \underline{g}_t}{\underline{g}_{t-1} \cdot \underline{g}_{t-1}} \ . \tag{3.23}$$

The value of $\underline{x}_{t+1}$ is obtained by searching for the least value of $F(\underline{x})$ from $\underline{x}_t$ along the direction $\underline{d}_t$.

Thus

$$\underline{x}_{t+1} = \underline{x}_t + \alpha_t \underline{d}_t \ , \tag{3.24}$$

where α_t is the value of α obtained by performing a linear search for the value that minimizes the function of one variable $\phi(\alpha)$ where

$$\phi_t(\alpha) = F(\underline{x}_t + \alpha \underline{d}_t) \tag{3.25}$$

The Conjugate Gradient algorithm guarantees convergence to a solution (F=0) in n iterations if F is a quadratic (convex) function. This is not the case in coefficient matching technique as F is, in general, a multi-modal function (i.e. possessing more than one minimum) as shown in Fig.[3.1]. This figure represents the shape of an overall error function F as a function of one element value (i.e. in one dimension).

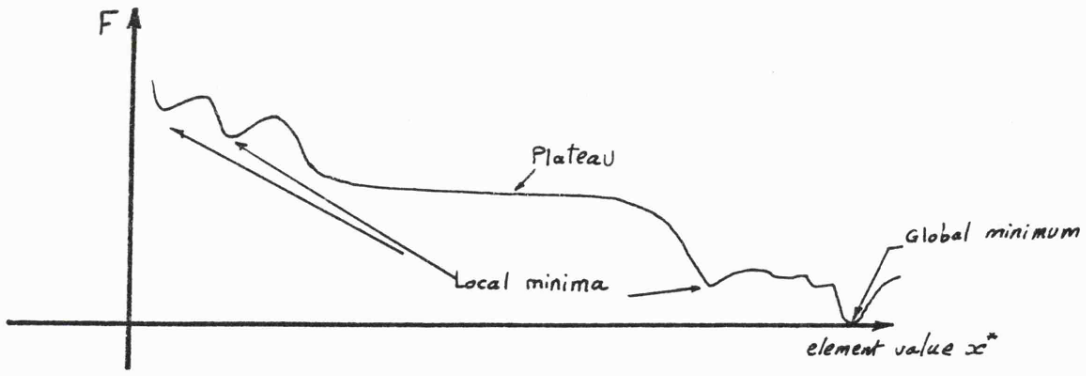


Figure [3.1] A plot of the overall error function value F against the value of one element x^*

The result of the Conjugate Gradient algorithm depends critically on the first search direction being the steepest descent direction. Therefore, when applied to general functions, it is usual to reset the search direction to the steepest descent direction every n iterations. In general, if $\underline{x}$ is any arbitrarily chosen vector, the initial rate of convergence of F is rapid, but after a certain number of iterations a plateau is reached.

The rate of reduction in the value of F becomes negligible. In this case, the gradient vector $\underline{g}$ is zero or very near to zero.
Then either,

1) $\underline{g}=\underline{0}$ and $F=0$ i.e. a solution is reached or ,

2) $\underline{g}=\underline{0}$ and $F\neq 0$ i.e. a local minimum is reached.

Thus, the Conjugate Gradients algorithm is useful to move from a poor initial guess as it reduces the value of F rapidly. However, it should then be replaced by another algorithm when the rate of decrease stops or becomes very small. Such an algorithm is the Gauss Newton algorithm.

3.4.3. The Gauss Newton Algorithm.

The Gauss Newton algorithm for solving an overdefined set of equations is based on a Taylor series expansion[12]. The condition for a minimum for a function f of a single variable x_j is

$$\frac{\partial f}{\partial x_j} = 0 \quad \text{and} \quad \frac{\partial^2 f}{\partial x_j^2} > 0 \tag{3.26}$$

Consider the case of f when it is a function of two variables, x_1 and x_2 say. If near a minimum, f is expanded in a Taylor series about the minimum $\underline{x}=\underline{\hat{x}}$, then the first few terms are

$$f(\underline{\hat{x}}+\underline{\delta x}) = f(\underline{\hat{x}}) + (x_1-\hat{x}_1)\frac{\partial f(\underline{\hat{x}})}{\partial x_1} + (x_2-\hat{x}_2)\frac{\partial f(\underline{\hat{x}})}{\partial x_2} + \frac{1}{2}\left[(x_1-\hat{x}_1)^2\frac{\partial^2 f(\underline{x})}{\partial x_1^2}\right.$$

$$\left. + 2(x_2-\hat{x}_2)(x_1-\hat{x}_1)\frac{\partial^2 f(\underline{\hat{x}})}{\partial x_1 \partial x_2} + (x_2-\hat{x}_2)^2\frac{\partial^2 f(\underline{\hat{x}})}{\partial x_2^2}\right] . \tag{3.27}$$

If $\underline{\delta x}^T = [(x_1-\hat{x}_1)(x_2-\hat{x}_2)]$, then

$$f(\underline{\hat{x}}+\underline{\delta x}) = f(\underline{\hat{x}}) + \left[\frac{\partial f}{\partial x_1} \quad \frac{\partial f}{\partial x_2}\right]\underline{\delta x} + \frac{1}{2}\underline{\delta x}^T \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_2^2} \end{bmatrix} \underline{\delta f} \tag{3.28}$$

where $\underline{\delta x}$ and $\underline{\delta f}$ are $\ll 1$.

In a more general way, if f is a function of n variable $\underline{x}$ then,

$$f(\hat{\underline{x}}+\delta x) = f(\hat{\underline{x}}) + (\nabla f(\hat{\underline{x}}))^T \underline{\delta x} + \frac{1}{2} \underline{\delta x}^T \left[\frac{\partial^2 f}{\partial x_i \partial x_j}\right] \underline{\delta x} \qquad (3.29)$$

where $\nabla f(\hat{\underline{x}})$ is the gradient of $f(\hat{\underline{x}})$ and is a column vector. Since $\hat{\underline{x}}$ is a minimum of $f(\underline{x})$, then any small change $\underline{\delta x}$ away from $\hat{\underline{x}}$ must increase $f(\hat{\underline{x}})$. As $f(\underline{x}) \overset{\lim}{\to} o$ and $\underline{\delta x}$ and $\underline{\delta f} <<< 1$ then,

$$-f(\hat{\underline{x}}) = (\nabla f(\underline{x}))^T \underline{\delta x} \; . \qquad (3.30)$$

If $\underline{f}$ is a vector of n functions, and each is a function of n variables $\underline{x}$, then

$$J(nxn) \; . \; \underline{\delta x}(nx1) = - \underline{f}(nx1) \cdot \qquad (3.31)$$

The Newton Raphson algorithm is

$$\underline{x}_{t+1} = \underline{x}_t + \lambda_t \underline{\delta x}_t \qquad (3.32)$$

where t denotes the number of the current iteration starting with $t=1$. λ_t is the value of λ obtained by performing a linear search for the value that minimizes the function of one variable $\phi(\lambda)$ where

$$\phi_t(\lambda) = f(x_t + \lambda \underline{\delta x}_t) \qquad (3.33)$$

On the $(t+1)^{'}$th iteration we have

$$\underline{\delta x}_{t+1} = - J_t^{-1} . \underline{f}_t \; . \qquad (3.34)$$

J^{-1}, the inverse of J will only exist when J is a non-singular square matrix.

If the system of equations is overdefined, so that there are m equations in n unknowns $(m > n)$, the Taylor series expansion gives

$$J(mxn) \; . \; \underline{\delta x}(nx1) = - \underline{f}(mx1). \qquad (3.35)$$

The least-squares solution of these equations can be written explicitly as

$$\underline{\delta x}(nx1) = - (J^T.J)^{-1}.J^T.\underline{f} \ . \qquad (3.36)$$

This is the well-known Gauss Newton method.

It is bad practice to determine the solution of equation (3.36) by forming the matrix $(J^T.J)$ since this increases any ill-conditioning of the problem. Instead,the equations (3.35) are best solved using Householder's transformations[56]. An appropriate routine from the NAG[57] library was used for this purpose.

The Gauss Newton algorithm usually gives rapid convergence to a solution $F=0$ (equivalently $\underline{f}=\underline{0}$) if a good estimate of the solution values are available. However when using coefficient matching technique this is not the case in general, and the initial estimate for the element vector $\underline{x}$ is far removed from the design goal. Potentially, equation (3.29) is no longer an acceptable approximation for the Taylor expansion. Under these conditions, $\underline{\delta x}$ could be very large, which could steer $\underline{x}$ to a region where J is singular. From equation (3.34), if $\underline{\delta x}=\underline{0}$, then either

(i) $\underline{f}=\underline{0}$ i.e. a solution, or ,

(ii) $J^T.\underline{f}=0$ and $\underline{f}\neq\underline{0}$ i.e. a local minimum.

The chances of the Gauss Newton algorithm reaching a solution will increase if it is preceded by a preliminary section to move from a poor initial guess to values of variables that are sufficiently near to those required by the Gauss Newton algorithm.

3.4.4. The Joint Conjugate Gradients/Gauss Newton Algorithm

As shown in Section 3.4.3., the Gauss Newton algorithm produces rapid convergence towards a solution of a set of non-linear simultaneous equations provided a sufficiently near starting point is available so that

the approximation of the Taylor series expansion of equation (3.29) will be valid. Similarly, in Section 3.4.2 it was shown that the Conjugate Gradients algorithm gives initially rapid convergence from arbitrary staring values, but will eventually reach a plateau from which convergence to a solution is unlikely. Cutteridge[31] suggested that a joint-algorithm, using Conjugate Gradients, remote from a solution and switching to Gauss Newton near to a solution would combine the better features of each algorithm and give convergence to a solution from starting values from which either algorithm used exclusively would fail.

It was found, by solving many different examples, that to pre-determine an empirical or semi-empirical criterion for the optimal point at which to switch from one algorithm to the other was impossible. The reason for this is that the performance of both the two algorithms depends on the form of the error function and the problem to be solved (see Section 3.2 and Krzeczkowski[24]). The method which was suggested by Cutteridge[31] proved to be very successful. Briefly, this method is to try for convergence in the Gauss Newton section initially and after every Conjugate Gradients iteration, returning to the latter algorithm for one iteration if the Gauss Newton algorithm fails to reach a solution. This process is repeated until convergence in Gauss Newton is finally obtained. Figure [3.3] gives a general flow diagram of this strategy.

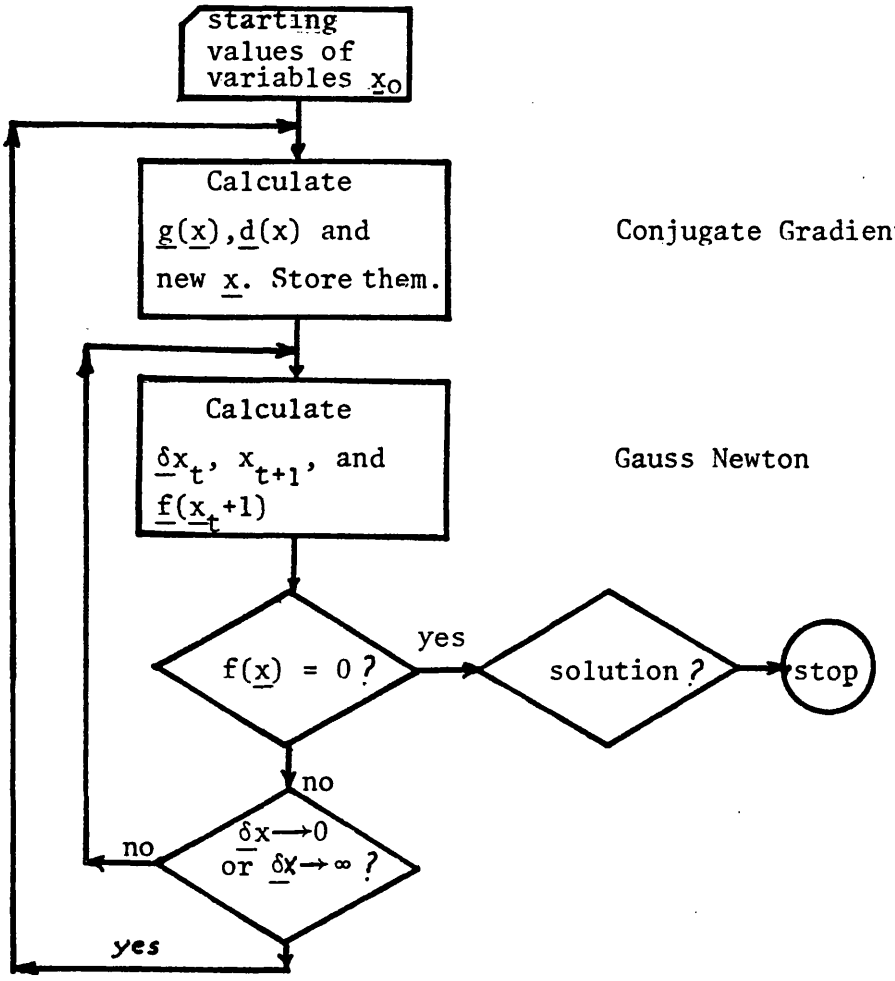


Figure [3.3] A flow diagram representation of the two-part strategy

The criteria for adjudging when to abandon the Gauss Newton algorithm and to return to the Conjugate Gradients algorithm is discussed in the next chapter.

Subsequent to the algorithm described above, Di Mambro[23], using the coefficient matching technique, developed an optimization algorithm based on Levenberg's algorithm. For a network with nl nodes, the algorithm developed by Di Mambro needs approximately $5nl^6$ operations per iteration. Using the two-part algorithm to obtain similar results (see Chapter 6) only nl^2m^2 operations per iteration are needed (approximately), where m is the number of coefficients to be matched. The reason for that is the two-part algorithm does not require the same amount of linear search[23].

3.5. Transformation of Variables.

Most optimization algorithms, including those used in the present work, converge best if the error function surface contours are circular or nearly so[15]. This occurs when all variables increments δx_j have roughly the same effect in reducing the error function f_i, i.e.

$$\frac{\partial f_i}{\partial x_{j1}} \approx \frac{\partial f_i}{\partial x_{j2}} . \quad j1 \neq j2 . \qquad (3.37)$$

Moreover, the negative gradient points almost directly towards the minimum[15]. Because this is not the general case in the synthesis of linear, passive electrical networks, a useful technique is to work with relative rather than absolute increments of the variable x_j. Further, in network synthesis, only real non-negative elements are to be considered in order to achieve a feasible realization.

The use of logarithmic variables (the base of the logarithm is irrelevant since it merely introduces a multiplicative constant[21]), which was first suggested by Temes and Calahan[15], seems to be a very convenient transformation. The reason for that is it fulfills the aforementioned two requirements. If the optimization is carried out in terms of the logarithmic variable, i.e.,

$$z_j = \log x_j \qquad (3.38)$$

then

1) The gradient vector $\underline{g}$ and the matrix of the first derivatives, J, becomes

$$\frac{\partial f_i}{\partial z_j} = \frac{\partial f_i}{\partial x_j} \cdot \frac{\partial x_j}{\partial \log x_j} = x_j \frac{\partial f_i}{\partial x_j} \qquad (3.39)$$

i.e. it provides the scaling factor which improves the convergence characteristics[15].

2) For $0 \leqslant x_j < 1$, $-\infty < \log x_j \leqslant 0$ and $1 \leqslant x_j \leqslant +\infty$, $0 \leqslant \log x_j \leqslant +\infty$.

The equation (3.32) in the Gauss Newton algorithm becomes

$$\underline{x}_{t+1} = \underline{x}_t \cdot \exp\left(\lambda \frac{\underline{\delta}\, x_t}{\underline{x}_t}\right) \quad . \tag{3.40}$$

Thus, the range of $\log x_j$ for $0 \leqslant x_j \leqslant 1$ has been made equal to the range of $\log x_j$ when $1 \leqslant x_j \leqslant +\infty$.

Because the starting element values $x_j \geqslant 0$, there is no possibility that the element values become negative as when $x_j \xrightarrow{\lim} 0$, $\log x_j \xrightarrow{\lim} -\infty$. Further, an initial element value of $x_j = 1$ seems adequate as it is equidistant from $\pm\infty$. Thus there is no perturbation in favour of any particular extreme (i.e. $\pm\infty$) . Electrically, when working with variables that are dimensionally admittances, $\log x_j \xrightarrow{\lim} -\infty$ corresponds to an open circuit and $\log x_j \xrightarrow{\lim^-} +\infty$ corresponds to a short-circuit. In the first case the element value will be zero. In the second case, the element will be shorted, i.e. the two nodes connected to this element will be connected together.

It was suggested[23] that numerical difficulties might result as x_j tends to zero and $\log x_j$ tends to minus infinity. In practice, these apparent difficulties were made use of by considering them as an indication that the element, which shows this tendency, be open-circuited (or conversely short-circuited). As the element will still have a finite value when it is removed, a discontinuity in the value of the overall error function F will follow. This is to be discussed in more detail in the next two chapters.

An alternative method for variable transformation, using squared variables, has also been tried and with some success[23,58]. Because the logarithmic transformation, the method chosen by the author and others[15,21, 22,24] , has proved to be perfectly satisfactory, no detailed investigation was made of possible alternatives.

In general, constrained optimization should be avoided if unconstrained methods are made possible by a simple transformation[15,21], although working with natural variables is the best as this preserves the multilinear aspects of the network coefficients. The only disadvantage of the latter technique is the possible arrival at a negative, unrealizable solution, which merely wastes time.

3.6. Summary

In this chapter the author reviewed the coefficient matching technique and the optimization algorithm used, which form the backbone of the automated design technique adopted.

Because of the difficulty and complexity of the network synthesis problem, a compromise between pragmatic and theoretical research attitudes is necessary for dealing with the issues of selecting different techniques and algorithms. No particular method or technique can as yet be recommended as the optimal method for all possible problems. The author's attitude was influenced by the simplicity of the methods described in this chapter and by practical evidence that the methods were rapid, reliable and efficient in linear network synthesis using computers automatically. The optimization algorithm was adopted from those currently available, making use of the special characteristics of the linear networks. The technique used is successful in finding a solution, if one exists, for any initial starting values.

The various sections in this chapter can be summarised as follows:

1) The form of the error function representation plays a decisive role in the rate of convergence to a solution of the optimization algorithm. The best method for the error function representation is

$$f_i = \frac{c_i}{Ka_i} - \frac{Ka_i}{c_i}$$

2) The rate and the range of convergence is improved to a great extent if the value of K, the multiplicative constant, is evaluated analytically. This value is updated such that the equation $\frac{\partial F}{\partial K} = 0$ is always satisfied, using

$$K = \left[\sum_{i=1}^{m} \left(\frac{c_i}{a_i}\right)^2 \Big/ \sum_{i=1}^{m} \left(\frac{a_i}{c_i}\right)^2 \right]^{\frac{1}{4}} .$$

3) The logarithmic transformation of variables is used to constrain the values of the network elements to the positive domain. Further, a logarithmic transformation scales the corrections which improves the performance of the optimization algorithm.

4) The coefficient matching technique employing the Conjugate Gradients/Gauss Newton optimization algorithm is efficient, rapid and reliable in finding the solution for a network synthesis problem. The implementation of the aforementioned recommendations increase the efficiency of the technique further.

CHAPTER 4

THE CONCEPT OF AUTOMATED NETWORK DESIGN

4.1 Introduction

As a result of numerous test runs carried out by the author with two-element kind and three-element kind networks containing up to ten nodes and with up to three variable common factors, it was found that the optimization algorithm described in the previous chapter would consistently obtain a feasible set of network element values satisfying the desired performance, from any arbitrary set of starting values, if the correct initial topology for the network was selected. In this case the solution is achieved readily without any difficulty and the automated design of the network presents no problem. However, it is unlikely that the designer will choose the correct topology for his starting network; especially if he does not know a solution or it is not possible to find one by using alternative methods (e.g. classical methods).

If a suitable solution topology is not known to the designer, there are two alternatives should the algorithm fail to find a solution with an arbitrary topology:

1) To repeatedly select a new topology until one yields a solution, which is clearly inefficient, or ,

2) To develop a technique to modify the starting network.

This technique would make use of the information obtained by the designer from the method of failure of the optimization algorithm. This information could be employed to develop the criteria for modifying the selected network in an evolutionary manner. Hence, in case the initially selected network topology proves unsatisfactory and can never yield a feasible solution, this chapter introduces ideas for changing the topology.

These ideas are based on the information obtained during the failure in the optimization algorithm. Potentially, this will lead to a suitable network (both topology and element values) which realizes the required set of functions exactly. Thus an efficient optimization algorithm is only the first building block in developing a more general and complex automated synthesis technique.

The concept of topological alteration was first introduced by Rohrer[16] in 1967. Later, Director and Rohrer[18] indicated that virtual element sensitivities of least squares performance functions would be useful as a criterion for element addition to a network when necessary.

Using the coefficient matching technique for 3-terminal RC network synthesis, criteria for removing or adding elements, with the constraint that the order of the required polynomials is unchanged during the topological modification, were developed[22-24] at the University of Leicester. The present work is a direct extension to the latter.

In this thesis only 3-terminal lumped, linear, passive networks, with no mutual inductance, are considered. These networks may consist of the combination of any two types of elements (e.g. RL, LC, RC) or all the three types of elements (RLC). More sophisticated techniques for topological modification are considered in this chapter which overcome the limitations of the previous work[23,24].

Using the coefficient matching technique described in the previous chapter, the following were achieved:

1) New criteria for removing or adding elements.

2) Criteria for adding or removing nodes with the possibility that the order of the required network polynomials may change during the topological modification.

3) A method for modifying the values of up to three common factors at the same time.

Sometimes, in RLC networks, element addition or element removal may change the order of the calculated polynomials. Further, node addition or node removal always corresponds to a change in the order of the required polynomials. In general, modifying the common factors helps, in obtaining solutions as shown in Section 2.5.

Before describing the details of the criteria developed for network modifications, it is convenient to introduce some terms, in the following three sections, which are used very frequently.

4.2. Virtual Elements

In Section 3.2 it was shown that the network synthesis problem is, in general, an underdefined problem and that there are more possible elements that might be considered than the number of the independent coefficients to be matched.

In practice, to consider all the possible elements in order to achieve a solution using an optimization technique is very slow, if at all possible. Furthermore, the Gauss Newton algorithm described in Section 3.4 would break down as the Jacobian is singular for an underdefined problem. For these reasons, only a subset of the total set of variables can be considered at a time in order to speed up the rate of convergence and to avoid ill-conditioning. This subset of variables corresponds to those network elements which were used to construct the starting network. These elements are initially non-zero. The rest of the possible elements are to be "frozen" at zero values and are known as the virtual elements. Further, any element or elements from the non-zero subset which have been driven close to zero (open-circuited) or infinity (short-circuited) by the optimization algorithm will be added to the virtual elements.

Similarly, those virtual elements which have the greatest tendency to go positive or which cause the largest reduction in the overall error function may be added to the non-zero subset.

4.3. Starting Networks

The subset of network elements selected as non-zero valued elements from the total possible set of elements construct the starting network. The selection of the starting network is confined by the structure of the required network polynomials. The spread of powers of p, the complex frequency, is helpful in deciding the minimum number of nodes for the starting network. The network configuration may then be estimated on the basis of a topological analysis (Section 2.3) and the information obtained from the various equivalent forms of the admittance functions (Section 2.4). Previous design experience is also of great help in selecting a starting network.

The starting network should be of the proper configuration and complexity in order to yield the required degree of polynomials. Any network realizing a given set of admittance functions must have a minimum number, nl say, of nodes. A realization with $nl+r$ nodes will, in general, introduce r common factors of unknown value (see Section 2.5). Thus even starting from a feasible topology, the optimization algorithm must obtain solution values for the network element and the common factors. In the author's experience, the introduction of even one common factor of unknown value degrades the performance of the optimization algorithm more than the introduction of several new elements. Thus in the present work, an optimum design is considered to be the one with a minimum number of nodes rather than a minimum number of elements.

For a starting network composed of the minimum number of nodes required, if the selected network fails to achieve a solution, more elements may be added by increasing the value of some virtual elements from zero.

An alternative method for selecting the starting network is to select a network with more elements, and perhaps more nodes, than those required. In this case a solution may be achieved by removing any elements which tend to be open circuited or short circuited. The latter is less efficient because of the reasons mentioned above. Moreover, in general, the optimization algorithm is less efficient as the number of elements under consideration is increased.

The minimum number of nodes can be defined as follows:

1) For RC or RL networks the minimum number of nodes, nl , is equal to the order of the denominator polynomial in the short-circuit admittance function plus three. Thus if the order of the denominator polynomial in the short-circuit admittance function is ml122 say, then

$$nl = ml122 + 3 .$$

2) Using the same notation of 1), then the minimum total number of nodes for an RLC or LC network is

$$nl = 3 + \frac{ml122}{2} , \quad \text{if ml122 is EVEN ,}$$

or

$$nl = 3 + \frac{ml122 + 1}{2} , \quad \text{if ml122 is ODD .}$$

The minimum number for each type of element in a 2-element kind network is given by the minimum number of nodes minus two. Each node must be connected to at least one of each type of element if these elements are to be effective in generating the required set of network polynomials. Furthermore, the network elements of a particular type should not form any loops for the same reason. For RLC networks similar conditions are applied.

In Section 3.5 it was shown that an initial element value of $x_j = 1$ seems appropriate when using the logarithmic transformation. For

$0 \leqslant x_j \leqslant 1$, $-\infty \leqslant \log x_j \leqslant 0$ and for $1 \leqslant x_j \leqslant +\infty$, $1 \leqslant \log x_j \leqslant +\infty$ whence $x_j = 1$ gives an initial starting value whose transformed value is equidistant from $\pm\infty$. Thus, there is no bias on x_j in favour of the short-circuit condition ($\log x_j = +\infty$) or the open-circuit condition ($\log x_j = -\infty$) .

4.4. The Strategy of Automated Network Design

In Chapter 3 it was shown that the problem of finding the network which can yield a particular set of admittance functions could be transformed to an equivalent optimization problem 'Find the values of the vector $\underline{x}(x_j \geqslant o)$ which minimizes the overall error function F , where $F = \sum_{i=1}^{m} f_i^2$ '. Further in Section 3.2 it was shown that this problem is an underdefined problem since, in general, there are more variables (elements) than independent equations (coefficients). To overcome this difficulty the problem is transformed to an overdefined or exactly defined problem. This can be achieved by considering, at any one time, a subset of the set of all possible variables.

Let n be the total number of possible variables (elements).

Let $\underline{x}$ be the vector of element values ($x_j \geqslant 0$, $j=1,2,\ldots,n$).

Let m be the number of independent equations (coefficients).

Let n_e be the number of variables to be considered at a time ($n_e \leqslant m, n_e \leqslant n$).

Let $\underline{x}_e$ be the vector of the corresponding element values.

The subset of elements $\underline{x}_e$ corresponds to those network elements which construct the starting network, i.e. the subset of non-zero valued elements. The rest of the elements $n_v (n_v = n-n_e)$ are the virtual elements which are frozen with zero values (see Section 4.2) i.e. the corresponding vector $\underline{x}_v = \underline{0}$.

In this context, it is better to consider the synthesis problem as the problem of minimizing a single function (the overall error function F) with respect to a number of degrees of freedom (e.g. number of nodes, number of elements, type of elements, ...). From the geometrical viewpoint, this function can be represented by a space S of n dimensions containing an unknown number of minima. The selection of any subset of n_e elements with starting element values $\underline{x}_e^o$ corresponds to a starting point p_e^o on a subspace s_1, of S, where the search for a global minimum will be confined. After t iterations, $\underline{x}_e^o$ will be adjusted to $\underline{x}_e^t$ and p_e^o to p_e^t within the same subspace s_1. If at p_e^t a global minimum is reached, i.e. the overall error function $F=0$, then a solution is achieved where n_e represents the solution topology and $\underline{x}_e^t$ represents the vector of the solution element values. This condition may be achieved if the initially selected topology is a solution topology. However, if the search in the subspace s_1 led only to a local minimum, i.e. the optimization algorithm fails to achieve a solution; the result obtained is useful in:

1) Strongly suggesting that the subset n_e does not have a solution.
2) Providing network element values which give a much lower magnitude for the overall error function than the starting values.

In order to explore the possibility of a solution in the rest of S, the search is to be continued in another subspace s_2 say, which corresponds to a modified subset of elements n_e'. In other words the initially starting network may be modified as follows:

1) By removing one or more elements. These elements are to be added to the virtual elements (see Section 4.5).

2) By adding one or more elements. These elements are to be selected from the virtual elements (see Section 4.6).

The process of changing from one subspace to another within S is to be repeated until a global minimum is found. However, in some cases, all the subspaces of S may be examined and no solution be found. Then the search has to be continued in a new space S'. The transformation of the search from one space S to another space S' may be achieved by:

1) Removing one node or more from the current set of nodes (see Section 4.5).

2) Adding one node or more to the current set of nodes (see Section 4.6).

Potentially, any change in the number of nodes will alter the order of the required set of polynomials. In general, after any topological modification, a check should be made to ensure that the initial requirements are still fulfilled.

To summarize, the selection of a suitable starting network which provides a response close to that required is of a great advantage. Such a network would need a minimum amount of topological alteration, if any, to achieve a solution. Hence any information which can be used in selecting such a network is of a great help. The author succeeded in obtaining the required information by making use of the various equivalent forms of the admittance functions, the topological analysis and the concept of the degree of connectivity discussed in Chapter 2. The block diagram in Fig.[4.1] shows the main features of the evolutionary approach in automated design using the coefficient matching technique.

It should be noted that the information obtained by the designer from the failure in the search in one subspace is very useful and relevant when conducting another search in another subspace, within the frame of the same space (i.e. element addition and element removal). However, it is very difficult to say the same thing when relating the information obtained in one space to another space (node addition or node removal).

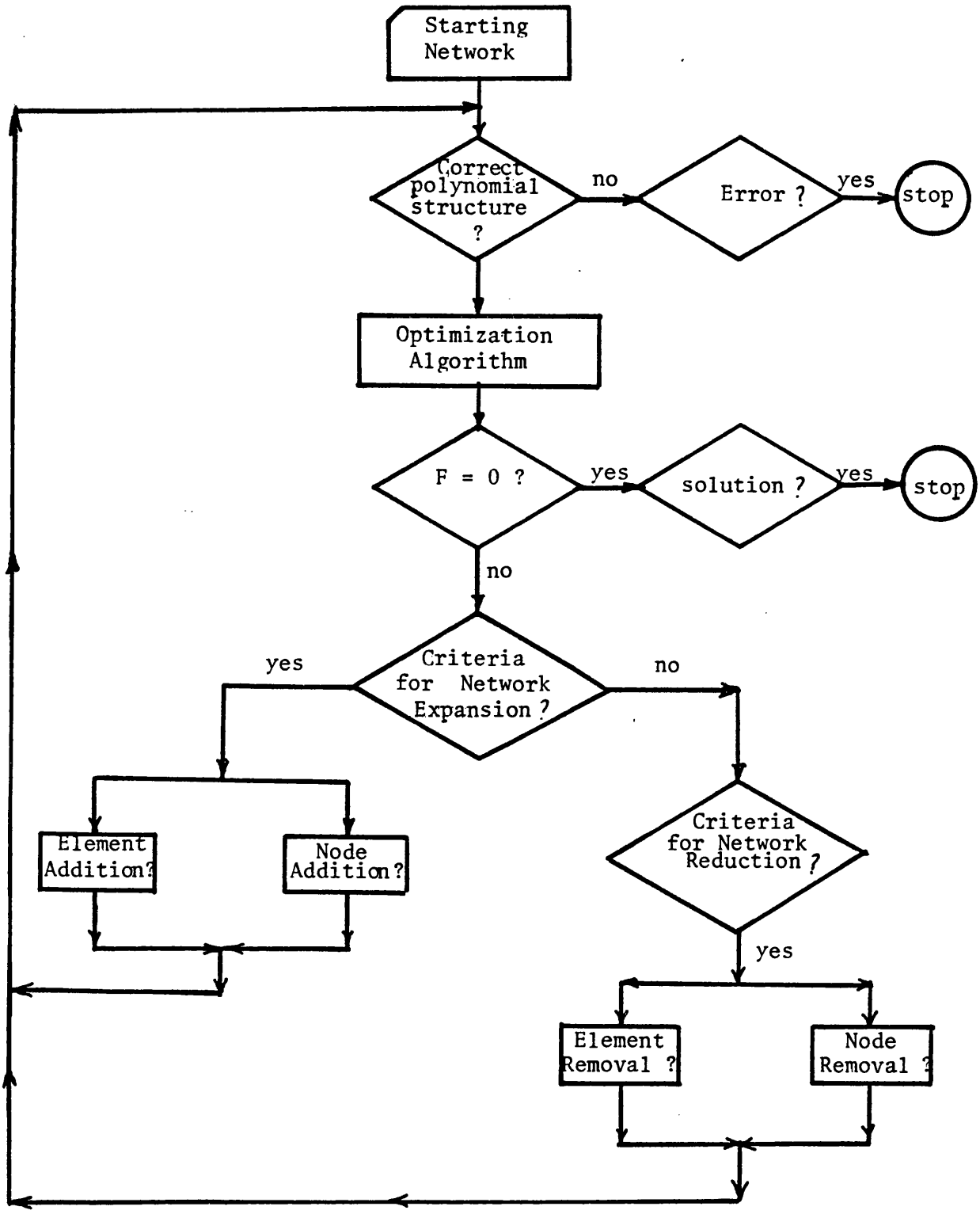


Figure [4.1] Block Diagram of Design Strategy

The structure of the design procedure shown in Fig.[4.1] may be summarized as follows:

1) Select a starting network using the criteria described in Section 4.3.

2) Check that this network fulfils the requirement discussed in Section 4.3.

3) Use the optimization algorithm to alter the element values iteratively, trying to reach a solution.

If the value of the overall error function F is reached below a certain minimum value ε, depending on the accuracy of the computer used ($\varepsilon \leqslant 10^{-18}$ say), an accurate solution has been reached. If the value of F is reached below a certain minimum value ε' say ($\varepsilon' \leqslant 10^{-4}$) and no further improvement could be achieved then, in this case, an engineering solution has been reached.

4) If $F > \varepsilon$ then the possibility of making a minor topological alteration should first be considered, i.e. adding or removing elements, or both, depending on the criteria available at the time.

5) Modify the topology accordingly and repeat steps 2), 3) and maybe 4), if necessary.

6) If all the minor alterations are exhausted and no exact solution has been reached then, if $F \leqslant \varepsilon'$ the current solution achieved could be accepted as an engineering solution or, add or remove a node according to the criteria available at the time.

7) Modify the topology and the order of the given set of polynomials accordingly and repeat steps 2) to 5) .

8) If an exact solution has not yet been achieved, repeat steps 6) and 7) then return to 2) until the minimum value of F is obtained.

4.5. Network Evolution by Reduction.

If a starting network is selected such that:

a) it yields polynomials of the correct order,

b) there are equal numbers of network elements and independent coefficients,

c) and it has the correct configuration and complexity,

then the optimization algorithm is certain to converge to a feasible solution from any set of initial element values. However, if the starting network contains more elements, and perhaps nodes than required, some of the excess elements, and maybe nodes, must be removed in order to achieve a solution.

4.5.1. Element Removal.

If the logarithmic transformation is used to constrain the element values to the positive domain (see Section 3.5), it is theoretically impossible for any element value to become identically equal to zero. However, in practice, the effect of an element on the coefficients will be zero when its value with respect to the other elements in the network is less than the computer accuracy. Thus an element with such a low value can be removed from the network. To reach such a low value for an element when using the two-part optimization algorithm described in Section

3.4 and employing the logarithmic transformation, a large number of iterations are usually required. During this period, the value of that element will decrease monotonically. Hence this could be considered as another criterion for element removal. On this basis, Cutteridge[59] suggested a criterion for element removal which the author and others[24] investigated and found to be very successful. If the element value is decreasing in the Gauss Newton algorithm, it must be associated with a negative Gauss Newton correction.

Hence

$$x_j^t = x_j^{t-1} . \exp(\lambda \delta x_j^{t-1})$$

$$\text{if} \quad x_j^t < x_j^{t-1} \text{ , then } \exp(\lambda \delta x_j^{t-1}) < 1 .$$

$$\text{i.e.} \quad e^{(\lambda \delta x_j^{t-1})} < 1 \quad \text{hence} \quad \delta x_j < 0 .$$

Cutteridge's criterion for element removal was as follows:

'If the correction associated with a particular element is negative over a certain number of iterations, and if the absolute value of that correction is increasing monotonically, then the element should be removed.'

$$\text{i.e.} \quad \text{If} \quad -\delta x_j^{t-4} > -\delta x_j^{t-3} > -\delta x_j^{t-2} > -\delta x_j^{t-1} > -\delta x_j^{t} > 0$$

then the x_j^{th} network element should be removed.

In practice, when this criterion was tried with several different examples, it was found that, occasionally, it would indicate the removal of an element unnecessarily. Hence, whilst using this criterion, the author found that if its removal is indicated after three trials in Gauss Newton section (not necessarily in succession), from different starting values for the network elements, it would lead to a better criterion for element removal. The different starting values for the network elements can be obtained by performing one iteration in the Conjugate Gradient section each time, starting from the values obtained from the previous Conjugate Gradient iteration.

As a result of employing this criterion for element removal, an element may be removed while its value is still of comparable magnitude with the other element values in the network, and thus while it is still contributing significantly to the current value of the overall error function F. Hence, the current value of F is badly perturbed. Generally, the value of the error function F will rise after the removal of an element when using the above criterion. However, if its removal was strategically necessary, then the value of F will fall rapidly after a few Gauss Newton iterations to below the value of F immediately before removal i.e. its removal was justified. On the other hand, if the element value is increasing in the Gauss Newton algorithm, it must be associated with a positive Gauss Newton correction. Hence

$$x_j^t = x_j^{t-1} . \exp(\lambda \delta x_j^{t-1})$$

$$\text{if} \quad x_j^t > x_j^{t-1} \text{ , the } \exp(\lambda \delta x_j^{t-1}) > 1$$

$$\text{i.e.} \quad e^{(\lambda \delta x_j^{t-1})} > 1 \text{ , hence } \delta x_j > 0 .$$

If the correction associated with a particular element is increasing and the order of magnitude of this correction is relatively high, then equation (3.29) is no longer an acceptable approximation to the Taylor expansion for this element and the element should be short-circuited. If this indication reoccurs in the Gauss Newton algorithm section after three trials from different starting values for the network elements, this element is short-circuited. Further, it was found that, especially with sophisticated networks, the correction associated with a particular element may oscillate between very high positive and very high negative values on every successive passages through the Gauss Newton section. The optimization algorithm has become unstable. Hence an element associated with such correction values is also to be removed. The last

two criteria for element removal was developed by the author.

The methods of element removal described here may remove more than one element simultaneously if each of these elements fulfills the criteria for element removal.

4.5.2. Node Removal

Some starting networks, although they generate the required set of network polynomials, may contain nodes in excess of the minimum number required. In Section 2.5 it was shown that excess nodes can generate common factors in the network admittance functions, some of which may be redundant. These redundant common factors correspond to the nodes which can be removed without preventing the network from generating the required set of network polynomials. Further, excess nodes bear the penalty of corresponding excess elements.

Wright[21] and Spence[60], working in the frequency domain, developed different methods for node removal. Di Mambro[23], employing the coefficient matching technique developed a further method for node removal. This method was applied to 2-terminal RLC networks where the realization has been generated by the Bott Duffin synthesis technique.

The author has developed his own techniques for node removal. These techniques are applicable to 3-terminal networks composed of any two types of elements (e.g. RC, LC, RL) or all three types of elements (RLC) . They proved to be simple and efficient (see Section 6.3). These techniques are applied under the following circumstances, see Fig.[4.2] :

1) The method of element removal developed by the author (Section 4.5.1) may remove more than one element simultaneously. As a result of successive element removal, one of the following two cases will occur:

(i) A node might be left unconnected to any element, i.e. an isolated node, Fig.[4.2a].

(ii) One of the internal nodes may be connected to one other node only by one element or more in parallel, Fig.[4.2b].

In both cases, such a node is not effective in generating the required set of network polynomials and it can be removed.

2) If the optimization indicates that two nodes, i and j say, should be short-circuited (the branch admittance becomes very large) then all elements connected between these two nodes are removed and the two nodes replaced by one node, Fig[4.2c].

3) If a node ℓ is connected to three other nodes by three elements of the same type forming a 'Star' connection, this node may be removed by a simple 'Star-Delta transformation', Fig.[4.2d].

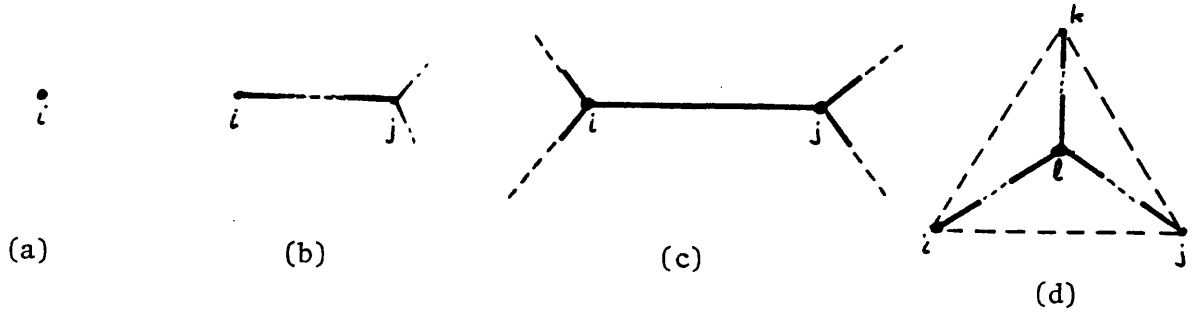


Figure [4.2] Node Removal

The procedure developed employing the aforementioned techniques for node removal consists of the following steps, see Fig.[4.3].

1) Starting from an initial network which yields the required spread of powers and with element values all unity, apply optimization to this network employing the two-part optimization algorithm described in Section 3.4.

2) If a solution is achieved ($F \leqslant 10^{-21}$) terminate the program.

3) Check if the criteria for element removal are fulfilled. If

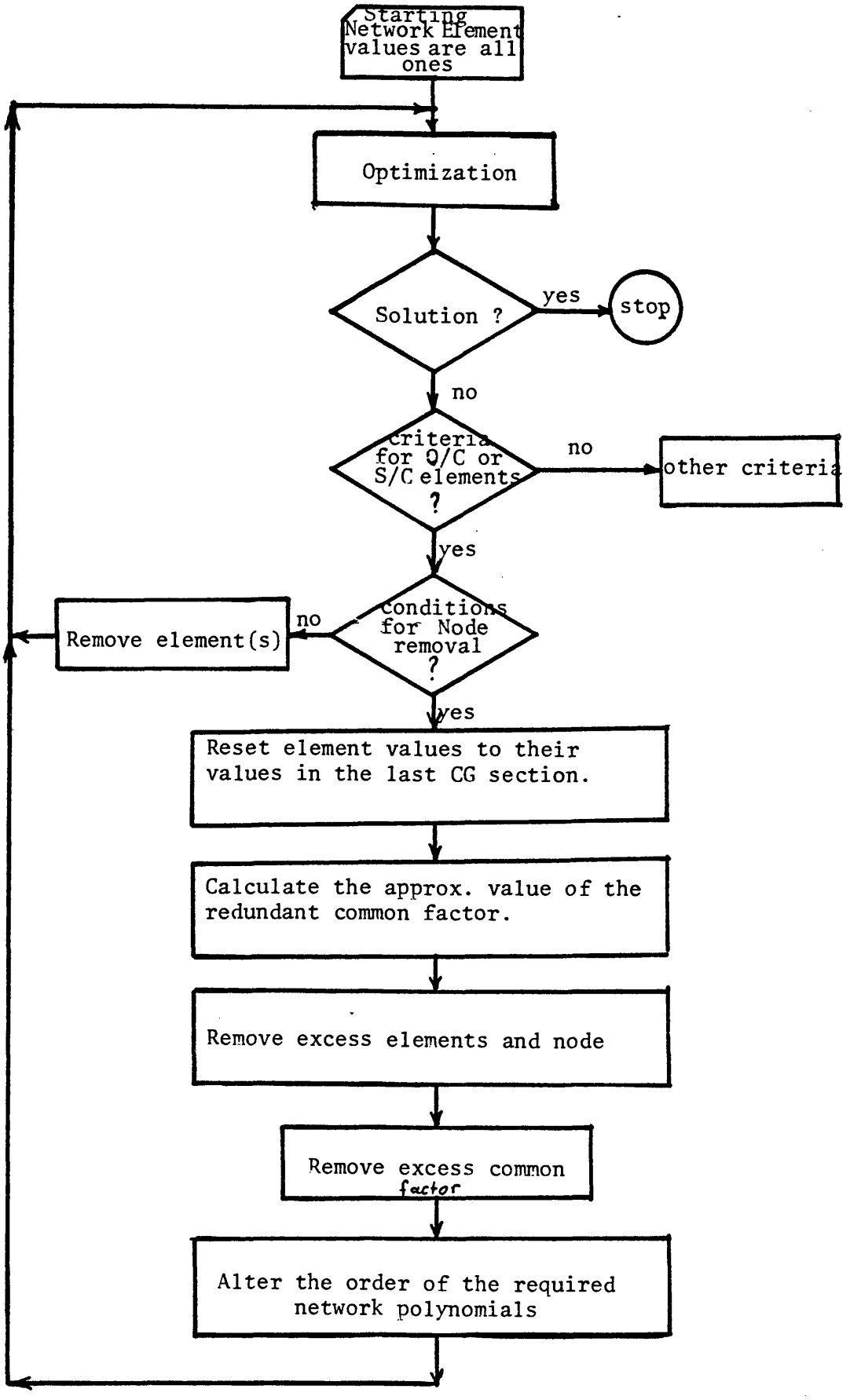


Figure [4.3] Block diagram for node removal technique

yes, then one or more elements are driven to nearly zero (open circuit) and/or to a very large value (short circuit).

4) If not, then check for other criteria (element addition).

5) Before removing any element, check if any condition for node removal will be fulfilled if elements are removed.

6) If the element or elements removal will not lead to node removal, then remove the appropriate elements. Return to step 2) .

7) If the element or elements removal will lead to node removal, then (i) reset the values of all the current elements, including those to be removed, to their values in the last Conjugate Gradient iteration.

(ii) Calculate the approximate value of the redundant common factor, which corresponds to the node to be removed as follows:

Let C be the value of all the current capacitors connected to the node to be removed.

Let G be the value of all the current conductances connected to the node to be removed.

Let Γ be the value of all the current inverse inductances connected to the node to be removed.

Then, the approximate value of the redundant common factor is equal to

$$\left(p^2 + \frac{G}{C} p + \frac{\Gamma}{C}\right) . \qquad (4.1)$$

If there are no capacitors connected to the node to be removed, then the approximate value of the redundant common factor is equal to

$$\left(p + \frac{\Gamma}{G}\right) . \qquad (4.2)$$

8) The approximate value of the redundant common factor is to be compared with all the values of the current excess common factors which are of the same order. The excess common factor which is the nearest to that calculated is to be removed.

9) Remove the element, node and common factor. Alter the required set of polynomials accordingly.

10) Return to step 2) .

This process is to be repeated until a solution is achieved.

4.5.3. Comparison with other methods of Node Removal

Working in the frequency domain Villalaz and Spence[60] described a very effective method for reducing complicated equivalent circuits. This method, although somewhat crude, is simple and works well. Also in the frequency domain, Wright[21] developed another method for node removal. This method mainly relies on the experience of the designer when working in an interactive mode.

Using the coefficient matching technique, the method developed by the author has the following advantages over that developed by Di Mambro[23].

1) It is applicable to 3-terminal networks composed of any two types of elements or all three types of elements.

2) It does not need high accuracy or special sophisticated algorithms.

3) In general, the roots of a polynomial are very sensitive to the values of its coefficients. Hence, unless the exact value of the redundant common factor is removed, the solution achieved may not be an exact one.

4.6. Network Evolution by Growth

To alter the topology of a starting network by removing elements and possibly nodes is only effective if the starting network contains a root network capable of yielding a solution. Unfortunately, this is not always the case. Hence for a more general design program, there must exist the capability of adding elements as well as nodes. These modifications are to be made, preferably on the basis of the properties of the network under consideration. A program capable of element and node addition, together with the removal of redundant elements and nodes when necessary, would clearly give scope for major modifications to the topology, though in rather a piecemeal fashion.

4.6.1. Element Addition

While the criteria for element and node removal were relatively simple to devise, the problems involved with element and node addition are greater. For element addition the following must be considered:

(i) To adjudge that the network currently being optimized will not yield a solution and that element addition is necessary.

(ii) To determine the type, position and value of element.

This is a very difficult task even for a simple RC network as there are many possibilities to be considered. For the addition of a resistor to an $n1$ node network, there are $n1(n1-1)/2$ possible positions, less the number of resistors already present.

The idea of adding elements to a network was first introduced by Rohrer[16]. Later, work of Director and Rohrer[18] indicated that virtual element sensitivities of least squares performance functions would be useful in element addition.

Using the coefficient matching technique and employing the joint Conjugate Gradients/Gauss Newton optimization algorithm (see Chapter 3), the optimization algorithm will fail to make any further reduction in the value of the overall error function F when a local minimum has been reached.

i.e. $$J^T.\underline{f} = 0 \qquad \text{and} \qquad \underline{f} \neq \underline{0} \,. \tag{4.3}$$

From experience obtained with the algorithm, it is unlikely to converge to a local minimum when a true solution exists. Thus this is a suitable criterion for element addition which was first introduced by Krzeczkowski[24]. Further, it was found[24] that, with some test problems the rate of convergence of the error function F is very slow and it could be improved by element addition. Krzeczkowski[24] introduced the following criterion:

> ' If more than fifty consecutive Gauss Newton iterations are performed without element removal being indicated and, if the value of F is still greater than 10^{-3}, then a new element is to be added. '

These two criteria have also been employed by the author and found to be successful.

For some test problems the value of the error function F in the Gauss Newton algorithm will oscillate between two points of equal values. If this criterion is to be repeated after commencing from three different Conjugate Gradient iterations, then a new element is to be added. Further, if the joint Conjugate Gradient/Gauss Newton algorithm becomes unstable because of successive failure in the linear search in the Gauss Newton section after commencing from three different Conjugate Gradient iterations, a new element is to be added.

The last two criteria were developed by the author.

When the criteria for element addition are satisfied, there is still the problem of the number, type, position and value of the element to be added. The efficiency of the element addition technique is very dependent on the optimal choice being made consistently and rapidly. Director and Rohrer[18] published a technique for element addition in specific places based upon the steepest descent algorithm. At the point of addition, the designer selects the element to add from the virtual elements with the most negative gradients. Di Mambro[23,58] developed an alternative algorithm which was successful. Both methods require the derivatives of the network coefficients with respect to the virtual elements at the point of addition. Cutteridge[61] has suggested a technique for element addition based upon the calculation of the optimum values of virtual elements. This technique is simple and allows for quantitative comparison of the various possibilities. This technique was developed and employed by the author and others[24].

In Chapter 3 it was pointed out that the formulation of the individual error functions, f_i, and the overall error function F employed in the present work is

$$f_i = \frac{c_i}{Ka_i} - \frac{Ka_i}{c_i} \qquad \text{and} \qquad F = \sum_{i=1}^{m} f_i^2 \ . \tag{4.4}$$

Further it was shown that K should be chosen such that $\frac{\partial F}{\partial K} = 0$. Then

$$K = \left[\sum_{i=1}^{m} \left(\frac{c_i}{a_i}\right)^2 \Big/ \sum_{i=1}^{m} \left(\frac{a_i}{c_i}\right)^2 \right]^{\frac{1}{4}} ,$$

and

$$F = 2\left[\sum_{i=1}^{m} \left(\frac{c_i}{a_i}\right)^2 \Big/ \sum_{i=1}^{m} \left(\frac{a_i}{c_i}\right)^2 \right]^{\frac{1}{2}} - 2m \ . \tag{4.5}$$

Suppose a virtual element, value μ, is introduced in the network. As the polynomial coefficients c_i^* are multilinear functions of the network element values, then

$$c_i^* = c_i + \mu c_i' \,, \qquad (4.6)$$

where c_i is the value of c_i^* when $\mu = 0$ and c_i' is the value of $\left(\frac{\partial c_i^*}{\partial \mu}\right)$ when $\mu = 0$.

Now, replacing c_i by c_i^* in equation (4.5) will clearly alter the value of F . If c_i and c_i' are kept constant, an optimum value of μ can be found by solving $F_{min} = \frac{\partial F}{\partial \mu} = 0$. Unfortunately it is not possible to find a closed form expression for an optimum value of μ in this case. However, a numerical linear search procedure can be conducted over positive values of μ on the function $\Phi(\mu)$ where

$$\Phi(\mu) = \left[\sum_{i=1}^{m} \left(\frac{c_i + \mu c_i'}{a_i} \right)^2 \cdot \sum_{i=1}^{m} \left(\frac{a_i}{c_i + \mu c_i'} \right)^2 \right] . \qquad (4.7)$$

Cutteridge[61] suggested that when analysing and calculating μ and F_{min} for all possible virtual elements, the element which gives the best reduction in the value of F is the element to be added. Later Krzeczkowski[24], using the same technique, suggested that the optimal element to add is the element whose value μ is the greatest. In both cases, an element whose value was negative should be disregarded.

In practice, by solving many different examples the author found the following:

1) If the starting topology is very close to that known to have a solution (i.e. with only one element missing), then the element to add is that which both reduces F the most and whose value μ is the greatest.

2) If the starting topology needs the addition of more than one element in order to achieve a solution, then the optimal element to add is either the element which reduces F the most or else the element whose value μ is the greatest. However, in general, this element μ_i say, has the second largest ratio of (μ_i/F_i) (where μ_i is the element value and F_i is the corresponding value of the overall error function F).

3) Usually, the optimum value of the new element obtained from analysis is relatively low (see Tables (1) and (2)). The main reason for this is that the current elements already existing in the network partly compensate for its absence. If the value of the new element is taken equal to its optimum value obtained from the analysis, considerable number of iterations would be wasted to increase its value in order to be effective in the calculation. On the other hand, if the new element value is boosted to a larger value (10 to 100 times the optimum value say), this might cause difficulty in the optimization algorithm as it has to compensate for the introduction of a new element with large value. Further, if the newly added element is to be removed later, this means that the optimization algorithm would take a larger number of iterations to reduce its value. Hence, as a compromise, the author found out that instead of inserting the new element with the optimum value obtained by analysis, it was inserted with a value equal to the geometrical mean of the element values existing in the network, the rate of convergence of the overall error function F could be greatly improved.

Accordingly, the technique developed and adopted by the author for element addition is as follows:

1) Analyse and calculate μ , F_{min} and (μ/F_{min}) for all possible virtual elements.

2) If the virtual element which has the largest ratio (μ/F_{min}) reduces F the most, then add this element.

3) If not, then add the virtual element which has the second largest ratio (μ/F_{min}) .

4) Instead of inserting the new element with the optimum value obtained by analysis, insert it with a value equal to the geometrical mean of the element values already existing in the network.

4.6.2. Node Addition

The different technique for topological modification described so far may converge to a local minimum with a value of F in the range 10^{-4} to 10^{-10} (see Chapter 7), and then no further reduction in F is possible. There is no way of breaking out from such regions of difficulty without adding a new node. In other words, if the starting network does not provide the degree of connectivity required to overcome the constraints imposed by the required set of polynomials, then an extra node (or more) must be added (see Sections 2.5 and 6.5). Adding or removing nodes necessitates the addition or removal of common factors.

In the case that the addition of a new node is necessary, the problem is much more difficult when using the computer design program. The main difficulties associated with node addition are caused by the number of extra variables which are introduced. For node addition the following must be considered:

1) To adjudge that the network currently being optimized will not yield a solution even with element addition.

2) To determine the position of the new node and the number, type and values of the elements associated with it. A new node must be the intersection of at least two network elements of different types, each connected to other nodes, if this node is to be electrically meaningful.

3) To determine the value of the corresponding common factor to be introduced.

Di Mambro[23] has described several strategies for node addition based upon the splitting of some of the network elements. Later Krzeczkowski[24] suggested another two strategies, node addition by Delta-Star transformation and node addition by duplication. The aforementioned strategies[23,24]

were based on the possibility of formulating a closed form expression to calculate the optimum value of the new common factor. This form depends on the type of the individual error function employed. Each method of formulating the individual error function will give a unique expression, if one exists, for the optimum value of the new common factor. The difficulty occurs when the optimization algorithm uses a type of error function which has no closed form expression for the optimum value of the common factor, which was the case in the aforementioned works. However, they assumed the feasibility of using an alternative form of the individual error functions, which does yield a closed form expression, purely at the node addition stage. Thus one error function was used to embody the design criteria namely

$$f_i = \frac{c_i}{Ka_i} - \frac{Ka_i}{c_i} \quad , \tag{4.8}$$

and another error function to calculate the optimum value of the common factor namely

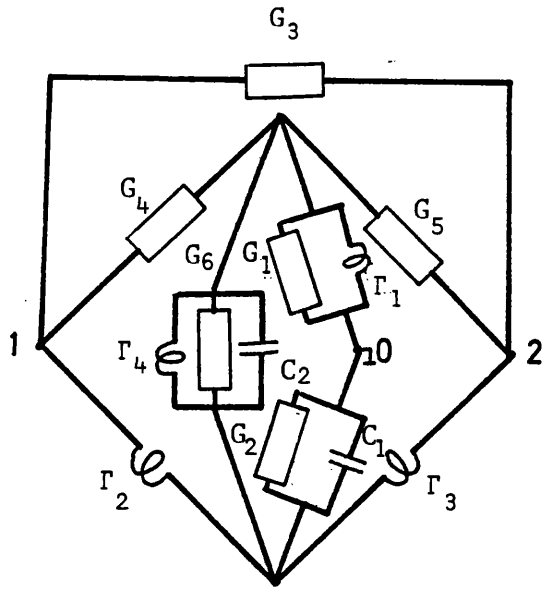
$$f_i = Kc_i - a_i \ . \tag{4.9}$$

The author found that, in practice, this assumption of using an alternative form of the individual error function, which does yield a closed form expression, purely at topological alteration, is not feasible even for element addition.

Example

The network shown in Fig.[4.4] was selected to synthesize equations (4.10) .

$$\left.\begin{aligned} Y_{11} = Y_{22} &= \frac{(p+20)(p+1.65)(1197p^3+56613.14p^2+28368.584p+191.184)}{(p+20)(p+1.65)(800000p+408000p+3840)} \\ -Y_{12} &= \frac{(p+20)(p+1.65)(3p^3-1.14p^2+197.176p+77.616)}{(p+20)(p+1.65)(800000p+408000p+3840)} \end{aligned}\right\} \tag{4.10}$$



$C_1 = 1$, $C_2 = 8.8$
$G_1 = 6.4$, $G_2 = 13.82$
$G_3 = 3.5 \times 10^{-6}$, $G_4 = G_5 = 0.0014$
$G_6 = 12.13$, $\Gamma_1 = 5.66$,
$\Gamma_2 = \Gamma_3 = 0.07$, $\Gamma_4 = 0.105$

Figure [4.4] Initial Network to Realize Equation (4.10)

For the element values shown in Fig.[4.4], the optimization algorithm reached a local minimum ($J.\underline{f} = 0$, $f_i \neq 0$) . At this point, the optimum values of all possible virtual elements and the corresponding values of F_{min} were calculated. Table (1) shows the results obtained for individual error functions given in equation (4.8). Table (2) shows the results obtained when equation (4.9) was employed to calculate the value of virtual elements in a program using equation (4.8).

By comparing Tables (1) and (2) we can notice the following:

1) The two different error functions give a difference of approximately thirty fold in the value of the overall error function F (printed as FSSQ in Tables (1) and (2)).

2) For several elements (e.g. inductance between nodes 0 and 4), while one error function gives a negative value, the other error function gives a positive value. Since the strategy for element addition ignores negative valued virtual elements, thus the two results obtained in Tables (1) and (2) are inconsistent.

If the technique developed for element removal, node removal and

ANALYSIS OF VIRTUAL ELEMENT

FROM NODE	0	TO NODE	1	INDUCTANCE	VALUE	=-2.11554246456E-04	FSSQ	=	2.48363049777E-02	-8.51794365731E-03
FROM NODE	0	TO NODE	2	INDUCTANCE	VALUE	=-2.11554245532E-04	FSSQ	=	2.48363049791E-02	-8.51794361965E-03
FROM NODE	0	TO NODE	4	INDUCTANCE	VALUE	=-3.69632368047E-06	FSSQ	=	2.49905862399E-02	-1.47908642277E-04
FROM NODE	1	TO NODE	2	INDUCTANCE	VALUE	=-6.78239156642E-07	FSSQ	=	1.74371421399E-02	-3.88962337521E-05
FROM NODE	1	TO NODE	3	INDUCTANCE	VALUE	=-2.30403354449E-05	FSSQ	=	2.49885867104E-02	-9.22034355601E-04
FROM NODE	2	TO NODE	3	INDUCTANCE	VALUE	=-2.30403336137E-05	FSSQ	=	2.49885867108E-02	-9.22034282303E-04
FROM NODE	0	TO NODE	1	RESISTOR	VALUE	= 1.82025279200E-05	FSSQ	=	2.43390003448E-02	7.47874919352E-04
FROM NODE	0	TO NODE	2	RESISTOR	VALUE	= 1.82025276937E-05	FSSQ	=	2.43390003611E-02	7.47874909554E-04
FROM NODE	1	TO NODE	4	RESISTOR	VALUE	= 1.03373592131E-06	FSSQ	=	2.49883414331E-02	4.13687288560E-05
FROM NODE	2	TO NODE	4	RESISTOR	VALUE	= 1.03373568984E-06	FSSQ	=	2.49883414338E-02	4.13687195919E-05
FROM NODE	0	TO NODE	1	CAPACITOR	VALUE	= 1.17357e03001E-06	FSSQ	=	2.38577770569E-02	4.91905019989E-05
FROM NODE	0	TO NODE	2	CAPACITOR	VALUE	= 1.17357602003E-06	FSSQ	=	2.38577770765E-02	4.91905015405E-05
FROM NODE	0	TO NODE	3	CAPACITOR	VALUE	= 7.27153136257E-04	FSSQ	=	2.49821522273E-02	2.91069051874E-02
FROM NODE	1	TO NODE	2	CAPACITOR	VALUE	= 3.84979998669E-09	FSSQ	=	2.30521935982E-02	1.67003628973E-07
FROM NODE	1	TO NODE	3	CAPACITOR	VALUE	= 1.15355568387E-06	FSSQ	=	2.39013504615E-02	4.82632010992E-05
FROM NODE	1	TO NODE	4	CAPACITOR	VALUE	= 1.11849750816E-06	FSSQ	=	2.39655542737E-02	4.66710469280E-05
FROM NODE	2	TO NODE	3	CAPACITOR	VALUE	= 1.15355567375E-06	FSSQ	=	2.39013504804E-02	4.82632006378E-05
FROM NODE	2	TO NODE	4	CAPACITOR	VALUE	= 1.11849749825E-06	FSSQ	=	2.39655542921E-02	4.66710464785E-05

Table (1)

FROM NODE	0	TO NODE	1	INDUCTANCE	VALUE	=-1.61641538955E+01	FSSQ	=	6.32071539720E-03	-2.55732980838E+03
FROM NODE	0	TO NODE	2	INDUCTANCE	VALUE	=-1.61641539005E+01	FSSQ	=	6.32071539721E-03	-2.55732980917E+03
FROM NODE	0	TO NODE	4	INDUCTANCE	VALUE	= 7.11479309169E+00	FSSQ	=	6.34870851058E-03	1.12066778304E+03
FROM NODE	1	TO NODE	2	INDUCTANCE	VALUE	=-6.15481373936E-02	FSSQ	=	4.37916700774E-03	-1.40547591094E+01
FROM NODE	1	TO NODE	3	INDUCTANCE	VALUE	=-1.55328708270E+00	FSSQ	=	6.34887884229E-03	-2.44655335420E+02
FROM NODE	2	TO NODE	3	INDUCTANCE	VALUE	=-1.55328708729E+00	FSSQ	=	6.34887884197E-03	-2.44655336155E+02
FROM NODE	0	TO NODE	1	RESISTOR	VALUE	= 1.65255299298E+00	FSSQ	=	6.18003710155E-03	2.67401791579E+02
FROM NODE	0	TO NODE	2	RESISTOR	VALUE	= 1.65255299190E+00	FSSQ	=	6.18003710187E-03	2.67401791391E+02
FROM NODE	1	TO NODE	4	RESISTOR	VALUE	=-6.11678927900E-04	FSSQ	=	6.34916515735E-03	-9.63400561712E-02
FROM NODE	2	TO NODE	4	RESISTOR	VALUE	=-6.11679962256E-04	FSSQ	=	6.34916515751E-03	-9.63402190810E-02

FROM NODE	0	TO NODE	1	CAPACITOR	VALUE	= 1.05666297006E-01	FSSQ	=	6.05720241207E-03	1.74447360048E+01
FROM NODE	0	TO NODE	2	CAPACITOR	VALUE	= 1.05666296953E-01	FSSQ	=	6.05720241236E-03	1.74447359951E+01
FROM NODE	0	TO NODE	3	CAPACITOR	VALUE	= 2.57396090365E+01	FSSQ	=	6.34883045601E-03	4.05422844646E+03
FROM NODE	1	TO NODE	2	CAPACITOR	VALUE	= 3.43382453231E-04	FSSQ	=	5.85793684352E-03	5.86183262817E-02
FROM NODE	1	TO NODE	3	CAPACITOR	VALUE	= 1.03694982376E-01	FSSQ	=	6.06959751532E-03	1.70843259564E+01
FROM NODE	1	TO NODE	4	CAPACITOR	VALUE	= 1.00461180415E-01	FSSQ	=	6.08688900041E-03	1.65045198636E+01
FROM NODE	2	TO NODE	3	CAPACITOR	VALUE	= 1.03694982322E-01	FSSQ	=	6.06959751580E-03	1.70843259462E+01
FROM NODE	2	TO NODE	4	CAPACITOR	VALUE	= 1.00461180361E-01	FSSQ	=	6.08688900061E-03	1.65045198543E+01

Table (2)

element addition fail to make any further improvement in the topology under consideration, a new node may be added if anyone of the following conditions is fulfilled:

1) The program removes an element which has just been added in the previous stage of topological modification and it is repeated for all virtual elements with positive optimum values.

2) All the possible virtual elements have a negative optimum value.

3) The addition of all the possible virtual elements, one at a time, will fail to make any further reduction in the value of the error function F.

4) The addition of a new element will contravene the constraints imposed by the required network polynomials on the required network (e.g. compactness, symmetry, etc.).

The strategy suggested by Krzeczkowski[24] for node addition by delta-star transformation was developed and employed by the author. In case any of the aforementioned conditions is fulfilled, a new node is added as follows, see Fig.[4.5].

1) Check if any of the conditions for node addition is fulfilled.

2) In the network under consideration, search for all the three elements of the same type in the network, connected in the form of a delta as shown in Fig.[4.6a].

3) If there are more than one group of three elements connected in the form of delta, select the delta connection whose nodes are connected to more elements with other nodes.

4) Using a standard transformation[1], the elements of the delta network can be replaced by the star network of Fig.[4.6b]. The

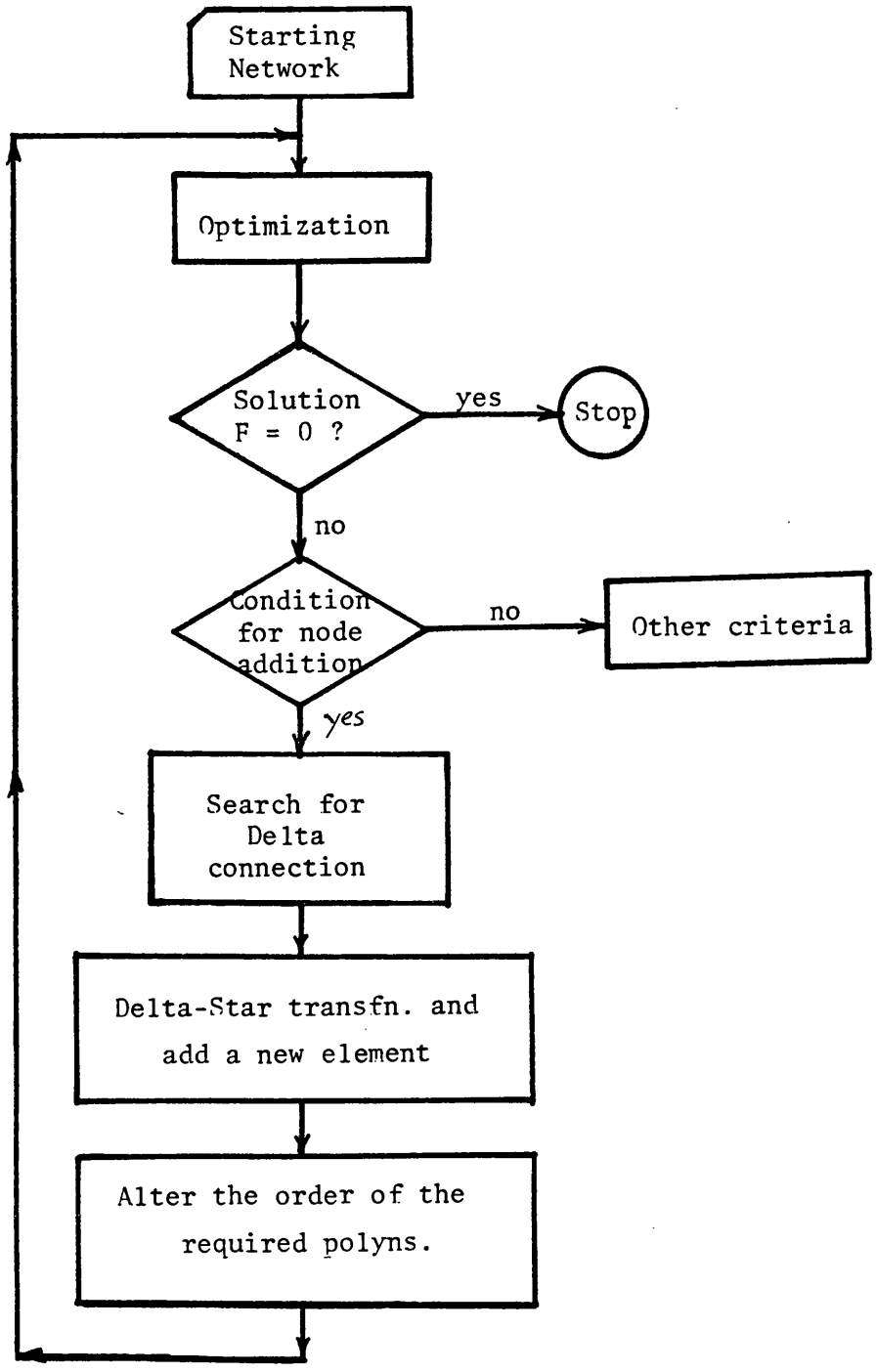
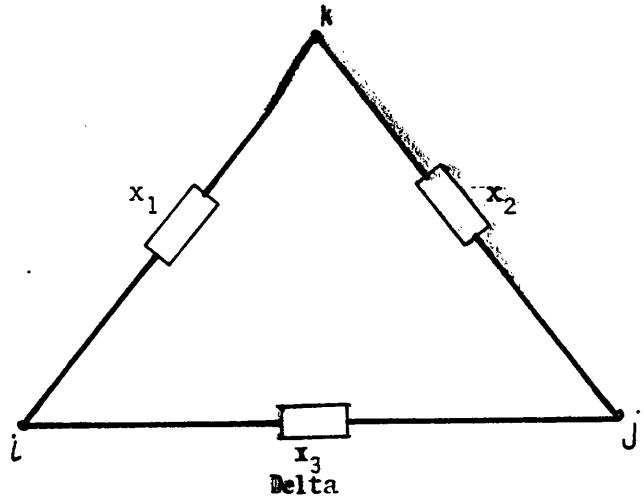
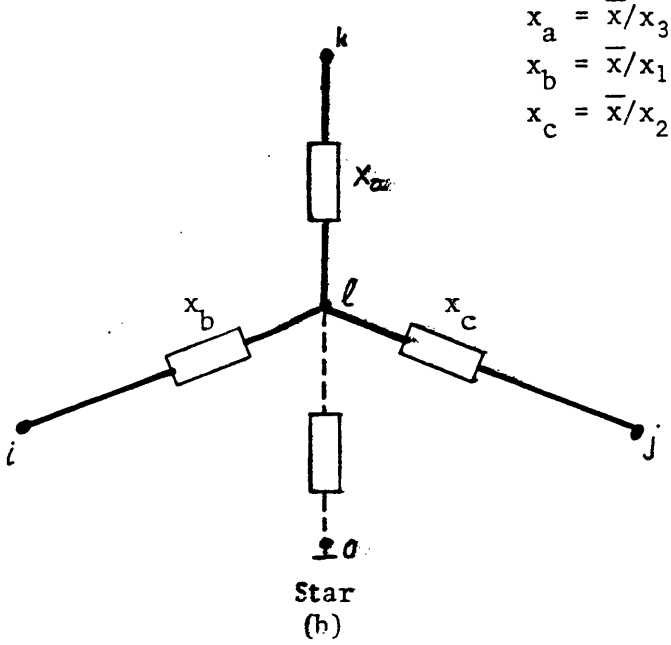


Figure [4.5] Block diagram for Node Addition



(a)

$$\bar{x} = x_1x_2+x_2x_3+x_3x_1$$
$$x_a = \bar{x}/x_3$$
$$x_b = \bar{x}/x_1$$
$$x_c = \bar{x}/x_2$$



(b)

Figure [4.6] The Delta-Star Transformation

equivalent values are functions of the old values as shown.

5) Add a new element of a type different from that of the delta network to the new node; the value of the new element is set equal to $((x_a+x_b+x_c)/3)$ (see Fig. [4.6]).

Thus, by the delta-star transformation and the addition of a new element, a new common factor of the form (p+1) (or (p^2+p)) is added to all four principal polynomials.

6) Modify the order of the required set of polynomials accordingly. Return to step 2) .

7) Vary the values of the common factor and the network elements, alter the topology if necessary, until a solution is finally achieved.

An example for the synthesis of RLC network employing node addition is given in Chapter 7.

4.7. Common Factor Adjustment

4.7.1. Introduction

In Section 2.5 the significance of common factors in the synthesis of electrical network using the coefficient matching technique was pointed out. In Section 4.5 it was shown that the removal of an extra node necessitates the removal of a redundant common factor. In Section 4.6 it was stated that the addition of an extra node necessitates the introduction of an extra common factor. Further, in some examples of RLC networks, the addition of an element may require the introduction of a new common factor. In general, if a solution is not known for a given set of short-circuit admittance functions then, neither theory nor previous experience will help the designer in determining beforehand the number of common factors and the associated number of extra nodes, necessary to

achieve a realization. Further, there is no way of predicting the final values to be taken by the common factors when a solution is achieved. Thus in a general network design program capable of altering the number of nodes of a given network, such as the program employed in the present work, facilities must exist for removing, adding and changing the values of common factors. In Section 4.5 the author described the strategy adopted to evaluate and remove a common factor when removing an excess node. In Section 4.6 the strategy adopted to introduce an extra common factor and its starting value when adding an extra node was described. In this Section, the author describes the strategies developed for modifying the values of common factors.

Krzeczkowski[24] introduced several strategies for adjusting the value of one common factor. These strategies were developed for 3-terminal RC networks with a fixed number of nodes. The strategies developed by the author are for 3-terminal two-element kind and three-element kind networks. Hence the adjustment of up to three common factors is considered. Further, the common factor considered in reference 24 was of the form $(p+\alpha)$. In the present work three different forms of common factors are considered.

4.7.2. Common Factor Adjustment using the Optimization Algorithm

As shown in Section 2.5, the introduction of a common factor of the form $(p+\alpha)$ to the set of four principal polynomials of the form given in equation (2.19) will produce the following:

1) An increase in the number of the non-zero coefficients by four.

2) An alteration to the values of the coefficients.

In this case, the Conjugate Gradient/Gauss Newton optimization algorithm will adjust the network element values $x_j, j=1,\ldots,n$ so that the overall error function F is minimized.

Then

$$F = \sum_{i=1}^{m+4} f_i^2 \quad , \quad f_i = \frac{c_i}{Kb_i} - \frac{Kb_i}{c_i} \quad ,$$

and

$$K = \left[\sum_{i=1}^{m+4} \left(\frac{c_i}{b_i}\right)^2 \Big/ \sum_{i=1}^{m+4} \left(\frac{b_i}{c_i}\right)^2 \right]^{\frac{1}{4}} \quad , \tag{4.11}$$

where $b_i = a_{i-1} + \alpha a_i$.

The derivatives of the individual error function f_i with respect to α are directly available.

Then

$$\frac{\partial f_i}{\partial \alpha} = - \frac{c_i a_i}{K b_i^2} - \frac{K a_i}{c_i} \quad . \tag{4.12}$$

In general, the optimization algorithm adopted in the present work, which was discussed in Section 3.4, requires the values of the error functions f_i and their first derivatives with respect to all the variables in order to find a global minimum. If a common factor $(p+\alpha)$ is to be included of unknown value α it can be considered as an extra variable. The values of the network elements and α at a solution can be obtained as follows:

1) Start with an estimate for the values of $\underline{x}$ and α .

2) From the required coefficients $a_i, i=1,\ldots,m$ calculate $b_i, i=1,\ldots,m+4$.

3) Calculate $c_i, i=1,\ldots,m+4$.

4) Calculate f_i . Evaluate K and F . If $F \neq 0$ then,

5) Calculate $\partial f_i/\partial x_j$ and $\partial f_i/\partial \alpha$ where $i=1,\ldots,m+4$ and $j=1,\ldots,n$.

6) Perform a Conjugate Gradient or Gauss Newton iteration to obtain the correction for $\underline{x}$ and α .

7) Update $\underline{x}$ and α . Return to step 2) .

If other forms of the common factor are required such as $(p^2+\alpha p+\beta)$ (quadratic common factor) or $(p^2+2\rho\cos\theta+\rho^2)$ (conjugate complex pair) , then in this case $b_i = a_{i-2} + \alpha a_{i-1} + \beta a_i$. Further if several common factors of unknown values are required, then they can be adjusted in the optimization routine in the same way as α .

From the experience gained by solving several examples, the aforementioned strategy was found to be efficient only if the starting network was composed of the number of nodes required to yield a solution. However, if the addition of an extra node is necessary, this method might lead to instability in the optimization algorithm. The reason for this is that the optimization algorithm is very sensitive to the order of the required polynomials and the values of their coefficients. During optimization, the values of the network elements and the common factors are adjusted according to the tactical information available at the time. This information will often lead to a local minimum within the error space of the current number of nodes. The common factors will start having large corrections which might lead to extreme values for the common factors. In the meantime, the reduction in the value of the error function F is very small. However, this difficulty can be overcome by employing the strategy described in the next section.

4.7.3. Adjustment of Common Factor by Linear Search and Optimization

In order to overcome the difficulties described in the previous section a new strategy for adjusting the value of the common factors is developed. While modifying the values of the network elements,

$x_j, j=1,\ldots,n$, K and k common factors say, if a stage is reached such that :

1) The value of the error function $F \geq 10^{-4}$.

2) The reduction in the value of the error function is very small (e.g. 0.1%).

3) The maximum absolute correction corresponding to the values of the network elements is $\leq$ 0.1% .

4) Some of the common factors are associated with a large positive correction (10% say) while the rest are associated with a large negative correction.

Then the following strategy is to be used instead of that described before, until the value of the error function F is reduced below 10^{-6} .

1) Reset the values of the current elements and common factors to their values in the last Conjugate Gradient iteration.

2) Calculate $c_i, i=1,\ldots,m+4k$.

3) Calculate $b_i, i=1,\ldots,m+4k$.

4) Calculate f_i . Evaluate K and F . If $F > 10^{-6}$ continue.

5) Modify only the value of the largest common factor by 10% of its current value while keeping the values of the rest of the common factors fixed.

6) Apply a linear search to the values of all the common factors to find an optimum value for F .

7) Keep the values obtained from step 6) for the k common factors fixed, evaluate $\partial f_i/\partial x_j, i=1,\ldots,m+4k$ and $j=1,\ldots,n$.

8) Perform five Gauss Newton iterations. If the rate of convergence of F improved by more than 0.5% , then continue in Gauss Newton algorithm.

9) If not, return to step 5) . Repeat steps 6) to 8) . Each time halve the amount of correction (e.g. 5%, 2.5% ...) . If no improvement then,

10) Repeat steps 5) to 8) by selecting the second largest common factor. Return to step 2) .

11) If $F < 10^{-6}$ then use the strategy described in the previous section until a final solution is achieved.

4.8. Summary

In this chapter the theme has been the development of techniques to modify the starting network. This has been successfully achieved by developing the following criteria:

1) Criteria for changing the topology by element removal.

2) Criteria for changing the topology by element addition.

3) Criteria for changing the topology by node removal, and altering the order of the required network polynomials by removing the corresponding excess common factor.

4) Criteria for changing the topology by node addition, and altering the order of the required network polynomials by introducing extra common factors.

5) Increasing the range of feasible solutions by common factors variation.

Although the present techniques overcome the limitations of the previous works[23,24], the efficiency and power of the present techniques could be improved by further investigations and particularly in the following areas:

1) Topological modifications by adding and removing elements simultaneously.

2) Topological modifications by adding and removing nodes simultaneously.

3) Developing more efficient strategies for adjusting the values of the common factors.

4) The present technique can be greatly enhanced by further investigation in the concept of the degree of connectivity.

CHAPTER 5

COMPUTER PROGRAMS FOR AUTOMATED NETWORK DESIGN

5.1 Introduction

In the previous two chapters the concept of automated network design using the coefficient matching technique has been described. Programs using this concept are capable of:

1) Optimizing an overall error function F to obtain, for a particular topology, the network element values which yield the required set of network polynomials.

2) Modifying the current network topology by element removal.

3) Modifying the current network topology by element addition.

4) Modifying the current network topology by node removal.

5) Modifying the current network topology by node addition.

6) Altering the order of the required network polynomials by removing excess common factors.

7) Altering the order of the required network polynomials by introducing extra common factors.

8) Enlarging the field of possible realizations by adjusting the values of up to three common factors simultaneously.

The process of developing an automated network design program, using the coefficient matching technique, has been under consideration at Leicester University since 1969. During this period, several programs for the synthesis of lumped, linear 3-terminal RC networks were

developed[22,23,24,54]. The language employed in these programs was ALGOL 60 and most of the work was performed on an ICL(ELLIOTT) 4130 computer.

For reasons of machine availability, higher accuracy and speed, a FORTRAN IV program was developed by the author. During the period of this research, the early development work was performed on the ICL(ELLIOTT) 4130 computer. Later an ICL 1906A computer was used. Finally much of the computation in the present work was performed on the CDC Cyber 72 and CDC 7600 computers. In general, the American computers do not support efficient Algol compilers. Further, FORTRAN IV is a more universal language and Fortran programs are often faster and more accurate.

Because of basic differences between the Fortran and Algol languages, and to avoid any duplication in work, the first version of the Fortran program was developed from the flow diagram representation of the Algol version which was available at that time.

In this chapter, the author describes how the techniques developed in Chapters 3 and 4 were implemented in one Fortran IV program for the synthesis of lumped, linear 3-terminal two element-kind and three element-kind networks. Further a comparison with the previously developed Algol program is given.

5.2. Basic Differences between the Algol and Fortran Languages

The facilities provided by any programming language will impose certain constraints on the way it is used. The following are the basic differences between the Algol and Fortran languages.

1) Arrays:

a) In Algol programs:

i) The array subscripts may be negative, zero or positive.

ii) The length of arrays and hence of dependent

routines can be changed during the running of the program i.e. Algol provides the facility of dynamic arrays.

iii) A particular array is limited to the block where it has been declared.

b) In Fortran programs :

i) The array subscripts must be positive.

ii) The length of arrays is defined on the basis of the largest dimension which may be required. Thus if less than this largest dimension is used, the excess store required is wasted.

iii) A particular array is normally available in only one segment of the program.

2) DO Loops and FOR Statements:

a) In Algol programs the initial parameters of FOR statement may be negative, zero or positive.

b) In Fortran programs the initial parameters of DO loops must be positive.

3) Subprograms :

a) In Algol programs:

i) The subprograms are known as Procedures.

ii) The facility of using recursive procedures exists.

iii) The parameters in a procedure may be called by name or by value.

iv) Blocks can be nested within blocks to any depth and declarations of procedures can be made at the head of any block. A procedure can of course be called only within the block in which it is declared.

v) Procedures can assign values to variables which are declared outside that procedure, at some higher level, and which have not been explicitly declared as procedure parameters.

b) In Fortran programs:

i) The subprograms are known as Subroutines.

ii) A subroutine parameter can only be called by name.

iii) Fortran has no block structure as has Algol. Thus, subroutines are declared linearly and cannot be nested one within another. Calls on subroutines can be made from either the main program or any other subroutine except that a subroutine cannot call itself nor can two subroutines call one another mutually.

iv) In Fortran, it is conventional to declare the main program as the first block and then there is no restriction on the order of the subsequent subroutines. However, with some Fortran compilers, the subroutines and the main program can be declared in any order.

v) A subroutine can assign values to variables outside its boundaries only through its parameters and 'common blocks'.

5.3. The Main Features of the Previously Developed Algol Program

5.3.1. Programming

The Algol program[24,54] made great use of the facility of declaring procedures within the inner members of nested blocks. Further all the arrays were dynamic arrays. Both facilities are not allowable in Fortran and meant that the preparation of an equivalent Fortran version of the Algol program was a far from straightforward matter.

5.3.2. Type of Networks

The program was used for the synthesis of lumped, linear 3-terminal RC networks with a fixed number of nodes.

5.3.3. The Error Functions

The individual error functions f_i were obtained from

$$f_i = \frac{c_i}{Ka_i} - 1 \quad , \quad i=1,\ldots,m \tag{5.1}$$

and

$$K = \sum_{i=1}^{m} \left(\frac{c_i}{a_i}\right)^2 \Big/ \sum_{i=1}^{m} \left(\frac{c_i}{a_i}\right)$$

The overall error function was $F = \sum_{i=1}^{m} f_i^2$. A realization was accepted when the value for the overall error function $F \leqslant 10^{-16}$ was obtained.

5.3.4. The Optimization Algorithm

The optimization algorithm employed was the joint Conjugate Gradient/ Gauss Newton algorithm i.e. the same algorithm as that described in Section 3.4 of this thesis.

5.3.5. Topological Modifications

Techniques for removing or adding elements were developed. One constraint was that the order of the required network polynomials be unchanged during the topological modification. In brief

i) An element was removed if its value with respect to the other elements was too small or it had been associated with an increasingly negative correction.

ii) An element was added whenever $||(J^TJ)^{-1}J^Tf||^2 < 10^{-7}$ and $F > 10^{-6}$ or if the optimization algorithm reached a plateau. All possible virtual elements were analysed. The element chosen was the one with the largest positive value given by the equation

$$\mu = \frac{\sum_{i=1}^{m} \frac{c_i}{a_i} \sum_{i=1}^{m} \frac{c_i c_i'}{a_i a_i} - \sum_{i=1}^{m} \frac{c_i'}{a_i} \sum_{i=1}^{m} \left(\frac{c_i}{a_i}\right)^2}{\sum_{i=1}^{m} \frac{c_i'}{a_i} \sum_{i=1}^{m} \frac{c_i c_i'}{a_i a_i} - \sum_{i=1}^{m} \frac{c_i}{a_i} \sum_{i=1}^{m} \left(\frac{c_i'}{a_i}\right)^2} \quad , \qquad (5.2)$$

where c_i' and μ are as described in Section 4.5.

5.3.6. Common Factors

Only one common factor of the form $(p+\alpha)$ could be considered at any one time. The common factor could be fixed or variable. The value of the common factor was adjusted, when necessary, by the global optimization algorithm. In this case the common factor was considered as if it was an extra element.

5.3.7. How the Program was Used

In order to use this program, the initial topology was suggested by the designer. Once the designer had specified:

i) the required set of polynomials to be realized,

ii) an initial topology of proper configuration and complexity yielding polynomials of the correct order,

iii) an initial value for the network elements,

iv) an initial value for the common factor, when required, and

v) whether the common factor was fixed or variable,

then no further designer interaction with the program was possible. The topology was evolved until either

i) a feasible solution was obtained

or ii) a time limitation was exceeded.

In the latter case, the synthesis technique was considered to be incapable of yielding a solution.

5.4. The Limitations of the Algol Program

Within the scope of lumped, linear 3-terminal RC networks the Algol program proved to be moderately successful[24,25,54,59]. However, the program had the following limitations:

1) The program had no facilities for the addition or removal of nodes. Thus before using the program, the designer had to choose an initial topology which yielded network polynomials of the correct order and to decide whether a common factor was needed or not.

2) The program could only synthesize RC networks.

3) The program could only adjust one common factor of the form $(p+\alpha)$ by global optimization.

4) The program employed the individual error function of the form given by equation (5.1). This form of error function representation is not very efficient[24].

5) Because there were no facilities for node addition, the method of topological modification could converge to a local minimum with a value of F in the range of 10^{-4} to 10^{-10} and no further reduction in F was obtained.

6) Because there were no facilities for node removal, the method of topological modification could remove elements until one of the polynomials was no longer of the correct order required, and then the program would fail due to numerical difficulties.

7) The topological modification strategy could only remove one element at a time.

8) The optimization procedure used was a joint Conjugate Gradient/ Gauss Newton algorithm. The Conjugate Gradient algorithm evaluated the corrections to the network element values on the basis of their gradients. Thus it would still make progress even if there were more variables in the network than independent coefficients. The Gauss Newton algorithm obtained the corrections by solving equations (3.36) using Householder's transformations[56]. With this method it is not possible to solve an underdefined system of linear equations and consequently it is not possible to use the Gauss Newton algorithm where there exists more variable elements than independent coefficients.

5.5 The Main Features of the FORTRAN Program Developed

Because of the basic difference between Algol and Fortran languages (see Section 5.2) and the complexity of the automated network design program, the first version of the Fortran program was a transformation rather than a

translation from the Algol program.

5.5.1. Programming

The Fortran program was developed specifically for use on a variety of computers. Thus, the program was written using standard Fortran IV. Any special features available with a particular compiler were avoided. The main segment of the program provided a link between over twenty subroutines. Each subroutine was to perform a particular task. All arrays were defined on the basis of the largest possible number of elements which could be required to synthesize the network under consideration (up to 10 nodes, 21 elements and three variable common factors). Thus if less than this largest number was used, the excess store requested was wasted. The different algorithms were constructed so that they evaluated the required values with maximum efficiency and accuracy. Because of this and the basic difference in accuracy between Fortran and Algol compilers, better accuracy is achieved in the Fortran program. Further, the overall speed of the Fortran program was at least 10% faster than the Algol program.

5.5.2. Type of Networks

This program was constructed for the synthesis of lumped linear 3-terminal networks. The networks could consist of the combination of any two types of elements (e.g. RL, LC, RC) or all the three types of elements (RLC) .

5.5.3. The Error Functions

The individual error functions f_i were obtained from equation (3.13), where

$$f_i = \frac{c_i}{Ka_i} - \frac{Ka_i}{c_i} \quad , \quad i=1,\ldots,m \ .$$

K was chosen according to the formula given by equation (3.18), where

$$K = \left[\sum_{i=1}^{m} \left(\frac{c_i}{a_i}\right)^2 \Big/ \sum_{i=1}^{m} \left(\frac{a_i}{c_i}\right)^2 \right]^{\frac{1}{4}} .$$

The overall error function $F = \sum_{i=1}^{m} f_i^2$. A realization was accepted if a value of $F \leqslant 10^{-21}$ was obtained (for a machine of 64 bits word length e.g. the CDC Cyber 72).

5.5.4. The Optimization Algorithm

The optimization algorithm employed was the joint Conjugate Gradient/ Gauss Newton algorithm (see Section 3.4). It is very similar to the algorithm used in the Algol version. The linear search techniques were altered in order to obtain higher accuracy. The two-part optimization algorithm would transfer to the Conjugate Gradient algorithm from the Gauss Newton section if:

(i) the step length obtained in the linear search was negative,

(ii) one of the elements was to be open-circuited (value too small),

(iii) one of the elements was to be short-circuited (value too large),

(iv) the Jacobian was singular or near singular,

(v) the overall error function F started to oscillate between two values.

5.5.5. Topological Modifications

The criteria for removing or adding elements and nodes which were described in the previous chapter were employed. The order of the required network polynomials could be changed during the topological modifications when necessary. Further,

i) The criteria for element removal were as described in Section 4.5.1.

ii) The criteria for node removal were as described in Section 4.5.2.

iii) An element was added whenever $||(J^TJ)^{-1}J^Tf||^2 < 10^{-7}$ and $F > 10^{-6}$ or if the optimization algorithm reached a plateau. All possible virtual elements were analysed as described in Section 4.6.1, the element chosen being the one with the largest positive ratio (optimum injection value/F_{min}).

iv) The criteria for node addition and the value of the new common factor so included were obtained as described in Section 4.6.2.

5.5.6. Common Factors

The program could consider the following forms for the common factor

i) $(p+\alpha)$, $\alpha \geq 0$ i.e. linear factor

ii) $(p^2+\alpha p+\beta)$ i.e. quadratic factor

iii) $(p^2+2\rho\cos\theta+\rho^2)$ i.e. complex conjugate pair of factors

iv) Any combination of the above forms.

The program could adjust the values of up to three common factors

simultaneously. The value of the common factor could be adjusted, if necessary, either by including it among the other independent variables in the optimization processes or by a linear search as described in Section 4.7. Further, for some examples with a large number of variables the following facilities existed:

i) To adjust the values of the common factors while keeping the network element values fixed.

ii) To adjust the network element values while the common factor values were fixed.

iii) To adjust the values of all the network element and common factors.

These facilities were found to be very useful in some difficult examples such as that given in Chapter 7.

5.5.7. How to Use the Program

In order to use this program, the designer has to specify the following:

i) The required set of polynomials to be realized.

ii) An initial topology of proper configuration and complexity yielding polynomials of the correct order.

iii) Initial values for the network elements.

iv) Initial values and types for any common factors, that are required.

v) Whether the common factors are fixed or variable.

No further designer interaction with the program is possible. The topology is evolved and the order of the required set of polynomials may be altered by the program until a final topology and element values yielding the required set of network polynomials is achieved, or until a time limit is

exceeded. The latter possibility is considered as a failure of the synthesis routines.

5.6. Additional Features Introduced

The Fortran program introduced the following additional features.

5.6.1. Programming

It is written in a more universal language which is implemented on all the computers currently available.

5.6.2. Type of Networks

The program can synthesize any lumped linear 3-terminal network consisting of any types of elements.

5.6.3. The Error Functions

The program employs the most efficient form available to evaluate the individual error function f_i .

5.6.4. The Optimization Algorithm

The optimization algorithm used combines the Conjugate Gradient and Gauss Newton algorithms. The Gauss Newton algorithm obtains the corrections for the network element values by inverting the matrix of the first derivatives J^TJ . Sometimes, for certain element values J^TJ is very nearly singular. The author found that the singularity in J^TJ can be avoided if a different set of starting values for the network elements is used. This can be achieved by performing one iteration in the Conjugate Gradient section. The Conjugate Gradient algorithm can make progress even if the number of variables in the network is more than independent coefficients.

5.6.5. Topological Modifications

The program extends the scope of the type and complexity of the network which can be synthesized as follows.

i) It can synthesize any two-element kind 3-terminal networks as well as three-element kind 3-terminal networks.

ii) It employs more sophisticated criteria for element removal.

iii) It can remove nodes.

iv) It employs more efficient methods for adding new elements.

v) It can add nodes.

5.6.6. Common Factors

The Fortran program enlarges the field of possible realizations by considering different forms for the common factors. Further it employs several techniques to adjust the values of up to three common factors simultaneously.

5.7. Limitations of the FORTRAN Program

Although the Fortran program developed by the author succeeded in overcoming most of the limitations of the previously developed Algol program, it still has the following limitations:

1) The program has no facilities for multiple topological alterations i.e. it cannot remove an element and add another element simultaneously. Further, while it is possible to remove more than one element at the same time, only one element can be added at a time.

2) Theoretically, the optimization algorithm employed can handle very large topologies. In practice, the accuracy of the calculations depends on the order of the required set of polynomials and the number of network elements. The accuracy decreases as the order of the required polynomials increases or the number of network elements increases.

3) The efficiency of the optimization algorithm is sensitive to the values of the common factors, especially when far from a solution. A carefully selected starting network and common factor values can be of a great help to the functioning of the program.

4) Although the optimization algorithm employed (see Section 3.4) proved to be reliable and efficient, the Gauss Newton algorithm obtains the correction by solving equations (3.36) using the Housholder's transformations[56]. With this method it is not possible to solve an underdefined system of linear equations and consequently, it is not possible to use the Gauss Newton algorithm where there exist more variable elements than independent coefficients.

5) It was observed for some starting networks , that the initial element values being all equal may cause the matrix of the first derivative J^TJ to be singular.

5.8. Summary

In this chapter the author described the Fortran program which he had developed. This program employs the techniques which were described in the previous two chapters. The main theme of the present chapter can be summarized as follows:

1) To explain the reasons for developing a Fortran program.

2) To summarize the main features of the previously developed Algol program and its limitations.

3) To briefly introduce the main features of the Fortran program developed and the additional features introduced.

4) To explain the limitation of the Fortran program.

In general none of the methods or criteria employed in the Fortran program is 'absolute' in the sense that, none is incapable of further improvement, or the collection of methods incorporated will always prove successful for all lumped, linear, 3-terminal network design problems. The efficiency and scope of the program developed can doubtless be improved by further investigation, for example, into the concept of the degree of connectivity (see Section 2.5) and by the introduction of multiple topological modification. However, the existing program promises much for the future.

CHAPTER 6

CASE STUDIES I : ATTEMPTED SYNTHESIS OF SOME TWO-ELEMENT KIND 3-TERMINAL NETWORKS

6.1. Introduction

In Chapters 4 and 5 of this thesis, the author described different techniques for altering the network topology and showed how they were implemented in an automated manner in a Fortran program using the coefficient matching technique. The final form of these techniques was determined by a systematic study involving numerical experimentation. Such techniques must be assessed by their efficiency in solving practical examples. The wider the range of practical examples with which these techniques can cope, the more useful they are. The synthesis of some networks, such as non-series-parallel networks having no series-parallel equivalent, cannot be achieved by classical techniques[8]. The author established that this class of network has a very special transfer phase-frequency characteristic which is given in Appendix B . The realizations for such networks are a further justification for the development of automated design techniques.

Computers have had a dominant influence in the present work. Most of the results described in this thesis were obtained by using the Leicester University CDC Cyber 72 computer. The early work was carried out using an ICL(ELLIOTT) 4130 computer. For a variety of reasons, an ICL 1906A computer and a CDC 7600 computer were also used.

During the course of this research a considerable number of design examples were produced. The examples presented in this chapter have been selected to illustrate the current possibilities in the synthesis of two-element kind networks using coefficient matching techniques.

6.2. Case Study 1 : Program Efficiency

In order to adjudge the speed, flexibility and accuracy of the Fortran program developed, some of the examples which had been solved before[24] using the coefficient matching technique were considered initially.

6.2.1. Network Realization with Fixed Common Factor

The example shown in Fig.[6.1] to realize the set of functions given by equations (6.1) is of particular interest for the following reasons:

1) The final network which can yield an exact realization is a non-series-parallel network possessing no series-parallel equivalent. Thus it could not be synthesized using classical methods.

2) The author established that this set of functions has a special transfer phase-frequency characteristic which was not previously known and which is described in detail in Appendix B .

3) The final realization obtained from the starting network shown in Fig.[6.1a] was not possible to achieve using a fixed common factor by the Algol program used in previous work[24].

4) As a result of using a fixed common factor, the final realization (originally given by Fialkow[8]) was obtained in the relatively short time of two minutes, compared with 45 minutes for the realization obtained by using a variable common factor[24] (both times refer to run times on the ICL(ELLIOTT) 4130 computer).

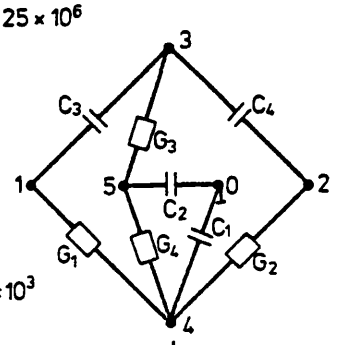
The set of equations realized by the network of Fig.[6.1f] are

$$\left.\begin{aligned} Y_{11} = Y_{22} &= \frac{\Delta_{11}}{\Delta_{1122}} = \frac{(p+1)(1197p^3+56613.14p^2+28368.584p+191.184)}{(p+1)(800000p^2+408000p+3840)} \\ -Y_{12} &= \frac{\Delta_{12}}{\Delta_{1122}} = \frac{(p+1)(3p^3-1.14p^2+197.176p+77.616)}{(p+1)(800000p^2+408000p+3840)} \end{aligned}\right\} \quad (6.1)$$

Initial element values :-

$C_i = G_i = 10^0$ ($i = 1, \ldots, 4$)

Initial error function = $2{\cdot}25 \times 10^6$



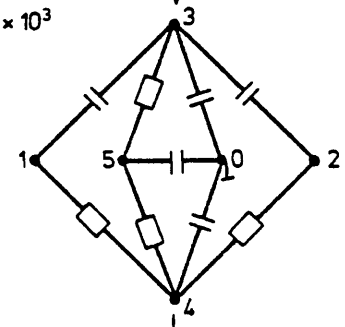
INITIAL STRUCTURE (a)

Final error function = $1{\cdot}167 \times 10^3$

ADD CAPACITOR
value $1{\cdot}7 \times 10^{-1}$ between nodes 0 & 3

Initial error function = $9{\cdot}8 \times 10^3$

Final error function = 4·07

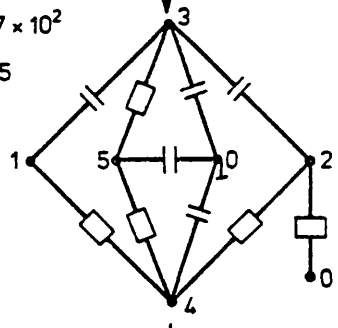


(b)

ADD RESISTOR
value 2·02 between
nodes 0 & 2

Initial error function = $4{\cdot}77 \times 10^2$

Final error function = 2·985



(c)

ADD RESISTOR
value 4·00 between nodes 0 & 3

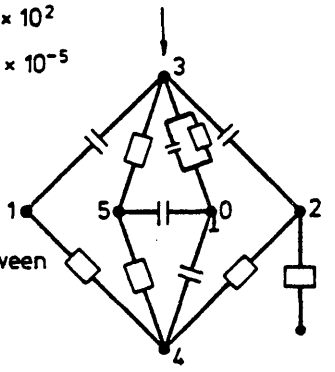
FIG [6·1] NETWORK REALIZATION WITH FIXED COMMON FACTOR

Initial error function = $4 \cdot 81 \times 10^{2}$

Final error function = $4 \cdot 86 \times 10^{-5}$

ADD RESISTOR

value $4 \cdot 4 \times 10^{-3}$ between nodes 0 & 4



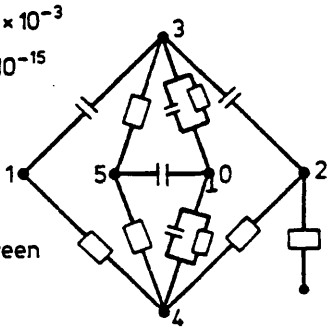
(d)

Initial error function = $5 \cdot 25 \times 10^{-3}$

Final error function = $9 \cdot 1 \times 10^{-15}$

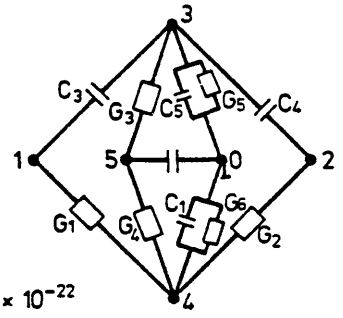
REMOVE RESISTOR

value $2 \cdot 6 \times 10^{-10}$ between nodes 0 & 2



(e)

FINAL STRUCTURE



(f)

Final error function = $1 \cdot 84 \times 10^{-22}$

Final element values :-

$C_1 = 20$, $C_2 = 5 \cdot 3016$, $C_3 = C_4 = 0 \cdot 0015$, $C_5 = 0 \cdot 597$

$G_1 = G_2 = 0 \cdot 07$, $G_3 = 0 \cdot 3948$, $G_4 = 2 \cdot 256$, $G_5 = 0 \cdot 1392$ & $G_6 = 0 \cdot 004$

FIG. [6·1] - CONTINUED

This set of functions, when the common factor $(p+1)$ is included, consists of the rational functions, a full quartic divided by a full cubic in each case. Thus the minimum number of nodes for a realization must be five plus the reference node 0. For realizing non-zero coefficients for p^4 and p^0 in Δ_{12} and Δ_{11}, a minimum of four capacitors and four resistors is required. However, since the coefficients of p^4 in Δ_{11} and Δ_{12} are not equal (hence the residues at the pole at infinity in Y_{11} and Y_{22} are not equal to that in $-Y_{12}$) then there must exist a purely capacitive path between nodes 0-1 and 0-2 (see Chapter 2). Similarly, since the coefficients of p^0 in Δ_{11} and Δ_{12} are not equal, then there must exist a purely resistive path between nodes 0-1 and 0-2. Thus any realization of (6.1) must have at least one capacitor and one resistor more than the minimal number required to realize Δ_{12}.

The initial topology selected (Fig.[6.1a]) contained only four capacitors and four resistors since this was sufficient to yield polynomials of the order required. Figs.[6.1b] to [6.1f] show the five topological modifications which were made by the program before a solution was obtained. From detailed analysis of the computer printout produced, there are several items of interest:

1) The starting network with all the element values equal to unity yielded one extra finite pole than those required.

2) The time taken between topological modifications was very uneven. During a total run time of two minutes, one minute was spent obtaining a local minimum with the initial topology. The other minute was spent in optimizing the value of the network elements and performing another three topological modifications until a solution was reached. The reason was that, for the initial starting network, the matrix of the first derivatives

was near-singular and the Gauss Newton algorithm made correspondingly slow progress. However, once the first local minimum was reached, the program added suitable elements and the final solution was achieved very rapidly.

3) Between the addition of elements in stages, the value of the overall error function, F , decreased monotonically. Since the strategy adopted (see Section 4.6) required that the value of the new element added be equal to the geometrical mean of the current existing elements in the network before recommencing optimization, the value of F necessarily increased after each element addition. However after only two Gauss Newton iterations the value of F was reduced below its value before the element addition. This justified the efficiency of the technique developed for element addition.

4) The second element to be added was a resistor between nodes 0 and 2 , and it was subsequently removed two stages later. This may be considered to cast doubt on the efficiency of the element addition technique. However the fault can also be considered to be more with the problem than the design method by virtue of the existence of a number of exact realizations for the required network functions with different topologies[24,27]. For example, if the program added a resistor between nodes 0 and 1 instead of a resistor between nodes 0 and 3 , an alternative realization would be achieved as given in the next example.

6.2.2. A Further Non-Series-Parallel Realization

The example shown in Fig.[6.2], to realize the set of functions given by equations (6.1), is of particular interest because:

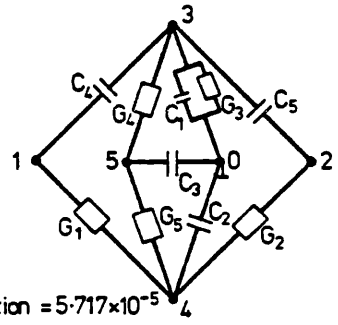
1) The final realization obtained had never previously been published.

Initial element values :-

$C_i = G_i = 10^0 \ (i = 1,\ldots,5)$

Initial error function $= 1{\cdot}499 \times 10^6$

INITIAL STRUCTURE



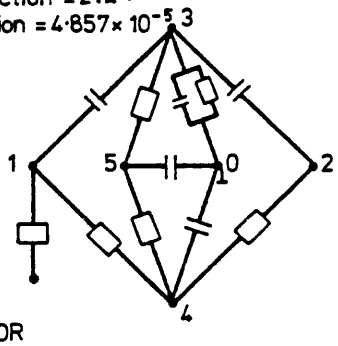
(a)

Final error function $= 5{\cdot}717 \times 10^{-5}$

ADD RESISTOR
value = 0·19 between
nodes 0 & 1

Initial error function $= 2.247 \times 10^2$

Final error function $= 4{\cdot}857 \times 10^{-5}$

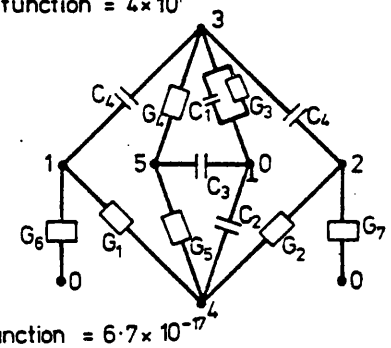


(b)

ADD RESISTOR
value = 0·2286 between
nodes 0 & 2

Initial error function $= 4 \times 10^{1}$

FINAL STRUCTURE



(c)

Final error function $= 6{\cdot}7 \times 10^{-17}$

Final element valves :- $C_1 = 5{\cdot}97$, $C_2 = 19{\cdot}668$

$C_3 = 5{\cdot}2267$, $C_4 = C_5 = 0{\cdot}0015$

$G_1 = G_2 = 0{\cdot}069416\dot{}$, $G_3 = 0{\cdot}142$

$G_4 = 0\ 392$, $G_5 = 2{\cdot}221$, $G_6 = G_7 = 0{\cdot}000583\dot{}$

FIG. [6·2] STRUCTURAL CHANGES LEADING TO A NEW REALISATION OF FUNCTIONS OF EQNS.(6·1)

2) It illustrates the effect of the starting network on the final solution. In general, the values of the current elements in any topology compensate, to a limited extent, for the absence of elements necessary for an exact realization. The way in which such a compensation is fulfilled depends on the current topology and element values. Accordingly, this will influence the optimization algorithm and the local minimum reached.

3) The initial starting network yielded the required number of poles, right sign of residues and the required spread of powers of p. Further, the solution was obtained in less than one minute.

4) The two realizations shown in Fig. [6.1] and [6.2] together with other realizations published elsewhere[24,54,59] illustrate that the problem of network synthesis is, in general, a multimodal problem, i.e. it is a problem with several equivalent solutions. Further, there must, in general, be an even greater number of quasi-equivalent realizations (very good approximate solutions) to any problem. However the technique developed, although employing an optimization algorithm that assumed unimodality, was successful in achieving realizations from initial networks far from a solution in respect to both element values and structure.

6.3. Case Study 2 : Realizations by Network Reduction

The following set of short-circuit admittance functions (originally given by Cutteridge[59]) are realized by the networks shown in Figs.[6.3] and [6.4].

$$\left.\begin{aligned} Y_{11} &= \frac{12(4p^3+15p^2+12p)(p+1)(p+2)(p+3)}{(p+1)^2(p+2)^2(p+3)^2} \\ -Y_{12} &= \frac{p^2(p+1)(p+2)(p+3)}{(p+1)^2(p+2)^2(p+3)^2} \\ Y_{22} &= \frac{(4p^3+15p^2+12p)(p+1)(p+2)(p+3)}{12(p+1)^2(p+2)^2(p+3)^2} \end{aligned}\right\} \qquad (6.2)$$

Initial element values:-

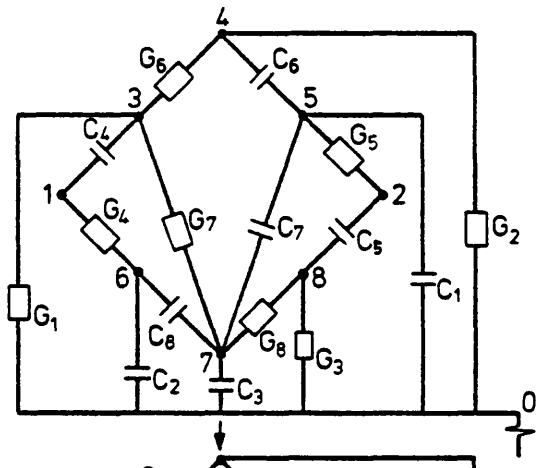
$C_i = G_i = 10^3 \; (i = 1, \ldots, 8)$

Initial error function = $8{\cdot}08 \times 10^5$

INITIAL STRUCTURE

Final error function = $2{\cdot}32 \times 10^2$

REMOVE $C_3 = 1{\cdot}18 \times 10^{-2}$

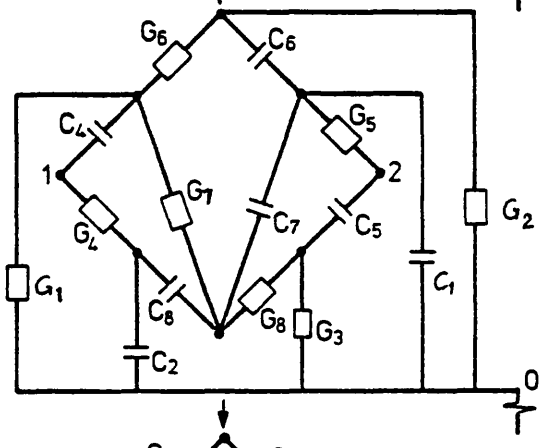


(a)

Initial error function = $2{\cdot}32 \times 10^2$

Final error function = 12·12

REMOVE $G_2 = 1{\cdot}64 \times 10^{-1}$

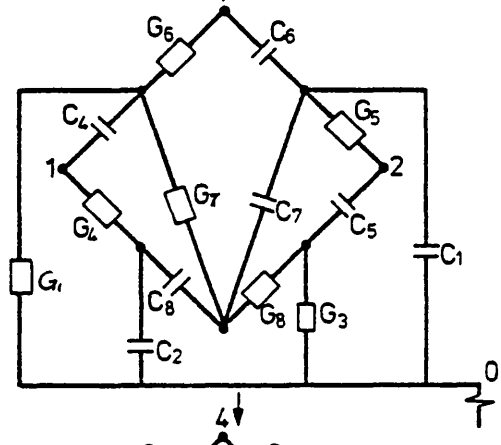


(b)

Initial error function = 12·9

Final error function = 1·498

REMOVE $C_1 = 1{\cdot}75 \times 10^{-4}$



(c)

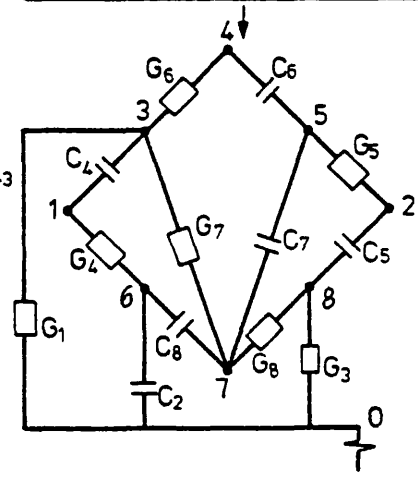
Initial error function = 1·496

Final error function = $8{\cdot}085 \times 10^{-3}$

REMOVE $C_6 = 3{\cdot}167 \times 10^{-4}$

$G_6 = 9{\cdot}2 \times 10^{-4}$

REMOVE NODE 4
& Common factor = (P+3)



(d)

FIG [6·3] REALIZATION BY NETWORK REDUCTION - EXAMPLE 1

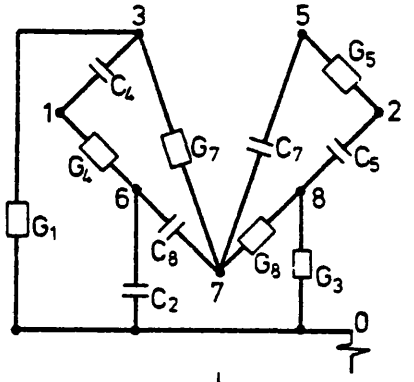
Initial error function = $1 \cdot 675 \times 10^{-2}$

Final error function = $1 \cdot 4 \times 10^{-3}$

REMOVE $C_7 = 1 \cdot 18 \times 10^{-3}$

$G_5 = 1 \cdot 18 \times 10^{-3}$

REMOVE NODES 5
& Common factor = (P+1)



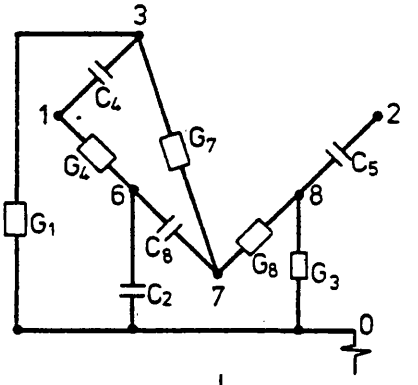
(e)

Initial error function = $3 \cdot 84 \times 10^{-4}$

FINAL STRUCTURE

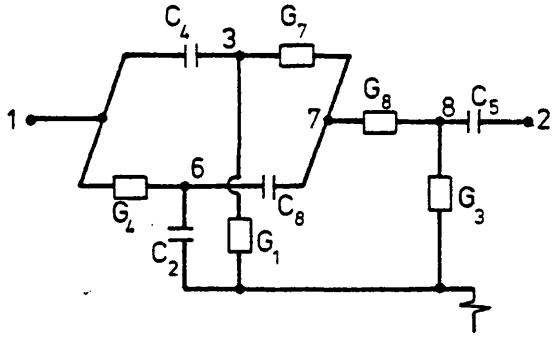
Final error function = $8 \cdot 57 \times 10^{-17}$

Final element values :-
$C_2 = 12 \cdot 998$, $C_4 = 10 \cdot 058$, $C_5 = 1 \cdot 667$
$C_8 = 0 \cdot 944$, $G_1 = 18 \cdot 754$, $G_3 = 0 \cdot 0625$,
$G_4 = 27 \cdot 88$, $G_7 = 1 \cdot 361$, $G_8 = 0 \cdot 2708$



(f)

FINAL STRUCTURE
REDRAWN



(g)

FIG. [6.3] - CONTINUED

This example, shown in Figs.[6.3a] to [6.3f] , was selected for the following reasons:

1) It demonstrates that the program is capable of removing nodes and elements using the techniques described in Section 4.5.

2) Cutteridge[59] published two other realizations. He achieved his realizations by using a different technique. Thus the most important conclusions to be drawn from the results described here are:

a) There are many apparently equivalent realizations (at least four) for such a simple set of functions.

b) It is reasonable to conjecture that there must be an even greater number of quasi-equivalent solutions.

c) Such a multiplicity of exact and quasi solutions are undesirable when using an optimization algorithm that assumes unimodality, especially if the network modification strategy employed is also viewed 'unimodally'.

Figs.[6.3b] to [6.3f] show the five topological modifications which were made by the program, before a solution was attained. From detailed analysis of the computer printout produced, there are several items of interest.

1) The time taken between topological modification was once again very uneven. Thus of a total run time of eight minutes, half was spent removing the first two elements, in almost equal time. The second half was spent removing five elements, two nodes and optimizing the value of the final network elements until a solution was achieved. The reason was that, at the start, there were more elements than independent coefficients. Thus the matrix of first derivatives was near-singular. Hence the

optimization algorithm performed, in the first stage of topological modification, more conjugate gradient iterations than were performed in all the following stages. However the optimization algorithm finally overcame this difficulty.

2) The value of the overall error function F decreased monontonically throughout the design process, except when the resistor between nodes 0 and 4 was removed. This was because the element removed was associated with a large negative correction and its value was relatively large on removal. However, the removal of this resistor was justified because after removal, the value of F decreased in ten Gauss Newton corrections from 12.94 to 3.12 , while the value of F immediately before removal was 12.12.

Fig.[6.4] shows another realization for the set of functions given in equation (6.2). The initial starting topology shown in Fig.[6.4a] is the same as that in Fig.[6.3a]. However, in this example, the initial element values are all equal to unity. This example illustrates the effect of the initial starting element values on the performance of the technique developed. Figs.[6.4b] to [6.4f] show the five topological modifications which were made before a solution was attained. From the computer output produced there are several items of interest.

1) The time taken between topological modifications was as uneven as in the previous example. However, in this case, one third of the total time of ten minutes was spent removing the first node and a second third optimizing the values of the final network elements until a solution was reached. This was because the starting element values caused the optimization algorithm to converge to the nearest local minimum ($F = 3.54 \times 10^{-13}$). The program removed three elements and converged to this quasi-solution in less than two minutes.

Initial element values :-

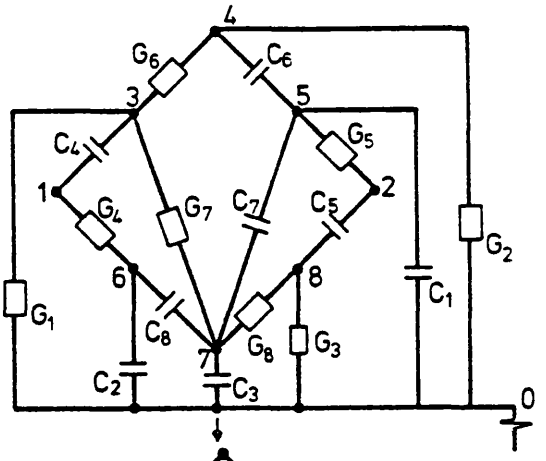
$C_i = G_i = 10^0$ ($i = 1..., 8$)

Initial error function = $6{\cdot}62 \times 10^3$

INITIAL STRUCTURE

Final error function = 61·07

REMOVE G_2 = 0·564

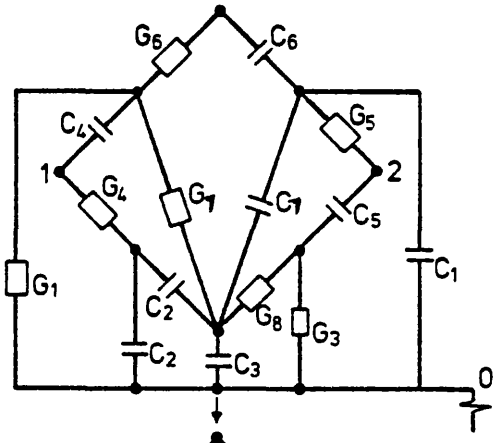


(a)

Initial error function = 42·07

Final error function = 16·46

REMOVE $C_7 = 4{\cdot}6 \times 10^{-25}$

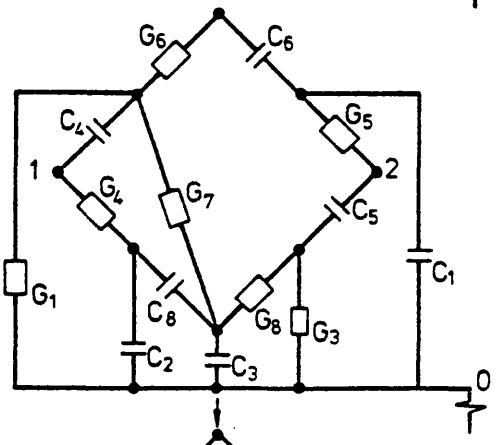


(b)

Initial error function = 8·85

Final error function = 1·08

REMOVE $G_3 = 7{\cdot}025 \times 10^{-3}$



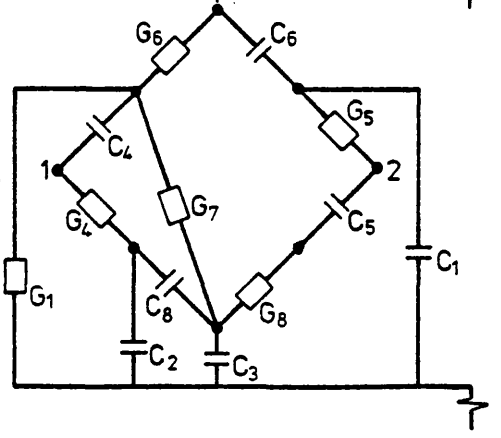
(c)

Initial error function = 0·965

Final error function = $3{\cdot}54 \times 10^{-13}$
(quasi - solution)

REMOVE $G_6 = 4{\cdot}7 \times 10^{-9}$,
$C_6 = 4{\cdot}4 \times 10^{-9}$

REMOVE NODE 4 &
Common factor = (P + 1)



(d)

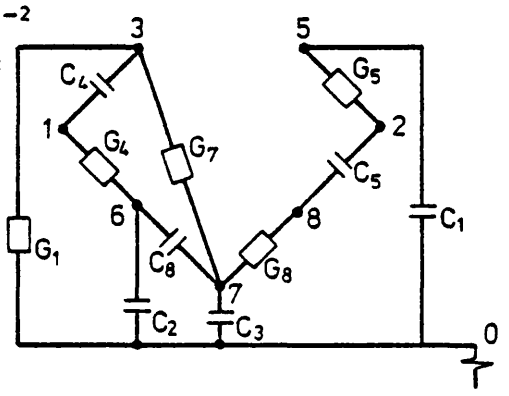
FIG. [6·4] REALIZATION BY NETWORK REDUCTION – EXAMPLE 2

Initial error function = $4 \cdot 58 \times 10^{-2}$

Final error function = $2 \cdot 74 \times 10^{-3}$

REMOVE $G_5 = 1 \cdot 05 \times 10^{-7}$,
$C_1 = 3 \cdot 68 \times 10^{-8}$ &

REMOVE NODE 5 &
Common factor = (P+3)



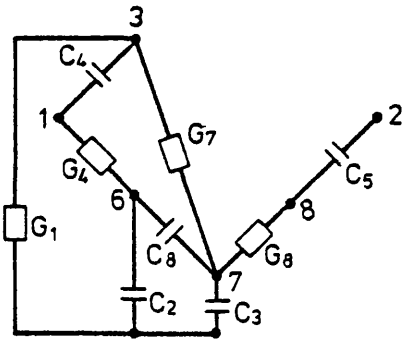
(e)

Initial error function = $1 \cdot 77 \times 10^{-2}$

FINAL STRUCTURE

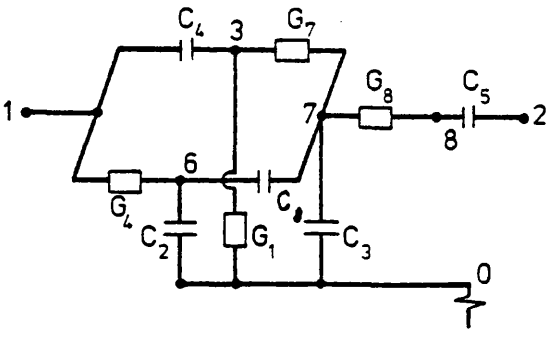
Final error function = $1 \cdot 75 \times 10^{-17}$

Final element values:-
$C_2 = 10$, $C_3 = 0 \cdot 5$, $C_4 = 13 \cdot 09$,
$C_5 = 1 \cdot 667$ $C_8 = 0 \cdot 909$, $G_1 = 23 \cdot 9965$
$G_4 = 21 \cdot 82$, $G_7 = 2 \cdot 181$, $G_8 = 0 \cdot 333$



(f)

FINAL STRUCTURE
REDRAWN



(g)

FIG. [6.4] - CONTINUED

2) Contrary to the previous example, after the removal of the first node, the value of the overall error function F jumped from 3.54×10^{-13} up to 4.585×10^{-2} . This was because the current network elements at the quasi-solution were reduced to relatively low values to compensate for the existence of the extra elements. All of the network elements connected to the internal nodes had values which were much less than their values at the final solution.

3) For the two examples, in both cases node 4 was removed before node 5 . However the redundant common factor, which was removed with node 4 , was (p+3) in the first example and (p+1) in the second example. This was because the sequence in which the elements were removed in each example was different. In Section 4.5, it was shown how the elements which are connected to the excess node at the time of removal decide the value of the redundant common factor.

6.4. Case Study 3 : New Set of Admittance Functions having no Exact 'Series-Parallel' Realization

The set of short-circuit admittance functions given by Fialkow[8] (equations (6.1)) has the following combination of particular characteristics:

1) $Y_{11} = Y_{22}$.

2) The residues at the finite poles are compact.

3) Non-compact residues at the poles at zero and infinity.

4) One of the coefficients in the numerator of the transfer admittance function $(-Y_{12})$ is negative.

Thus this set of functions can only be realized by a non-series-parallel network[8,24]. Later Krzeczkowski[24], with a theoretical approach,

succeeded in obtaining a realization having altered Fialkow's set of functions to a set with equal residues at the pole at zero. In the example in this section, the author investigates the possibility of formulating a set of functions similar to those given by Fialkow[8] but with compact residues at the poles at zero and infinity.

Before going into the details of this investigation attention is drawn to the following:

1) It was not possible to formulate such a set of functions that were known to be realizable using any known theoretical approach; instead, the synthesis of several such sets of short-circuit admittance functions was tried using the program developed. It was rather difficult to find an exact realization for the different selected sets of functions.

2) The search for networks which can realize the required set of functions was limited to networks having the minimum number of nodes (in this case five nodes plus the reference node, node 0). Thus, during the process of topological modification, if a local minimum is reached where the criteria for node addition are satisfied, the program will be stopped.

The best result was obtained from the example shown in Fig.[6.5]. The network shown in Fig.[6.5c] yielded a quasi-realization for the following set of short-circuit admittance functions:

$$\left.\begin{aligned} Y_{11} = Y_{22} &= \frac{(p+0.5)(100p^3+32344.84p^2+16271.3351p+77.381616)}{(p+0.5)(800000p^2+408000p+3840)} \\ -Y_{12} &= \frac{(p+0.5)(100p^3-0.5p^2+172.17383p+77.381616)}{(p+0.5)(800000p^2+408000p+3840)} \end{aligned}\right\} \quad (6.3)$$

The following comments are made with respect to this example:

1) Because of the negative coefficient in the numerator of the transfer admittance function, this set of functions cannot have an exact series-parallel realization. However a very good quasi-realization

Initial element values:-

$C_i = G_i = 10^{0}$

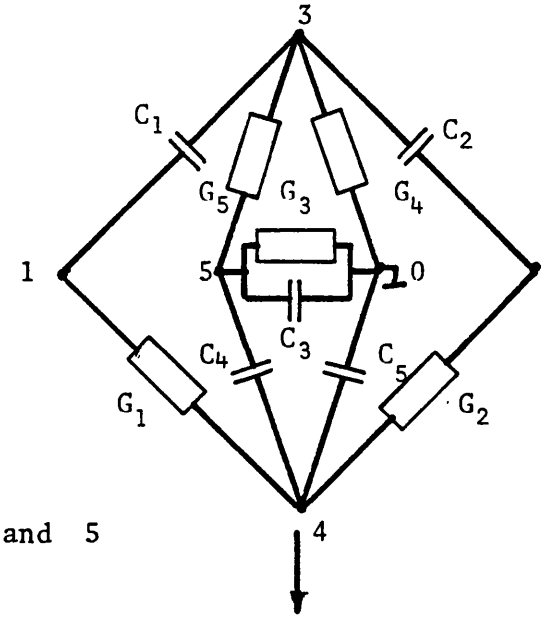
INITIAL STRUCTURE

Initial value of error function = 6.4×10^{4}

Final error function = 46.7

Remove Resistor

Value 0.5 between nodes 3 and 5



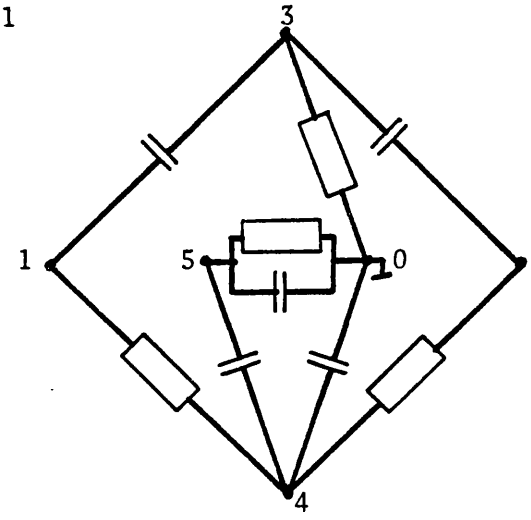
(a)

Initial error function = 46.01

Final error function = 1.46

Remove Capacitor

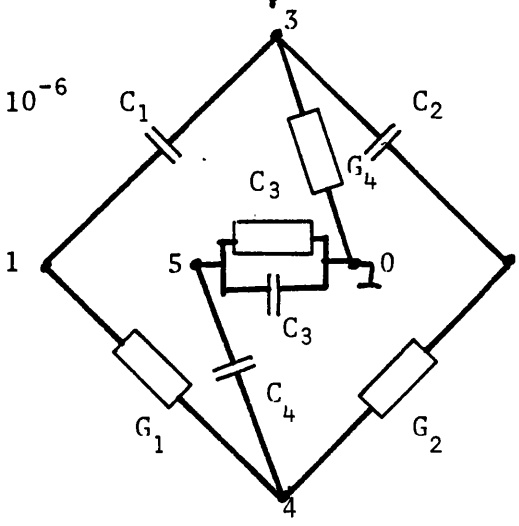
Value 1.9×10^{-4} between nodes 0 and 4



(b)

Initial error function = 1.44

Final error function = 1.56×10^{-6}



(c)

$C_1 = C_2 = 0.00025,$

$C_3 = 76.2 \ldots, C_4 = 8.383$

$G_1 = G_2 = 0.0403,$

$G_3 = 36.98 \ldots,$

$G_4 = 0.00026,$

$K = 3.994 \times 10^{-7}$

Figure [6.5] Series-Parallel Quasi-Realization for the Functions of Equations (6.3)

could be obtained ($F = 1.56 \times 10^{-6}$) that was series-parallel in form.

2) The example shown in Fig.[6.5] is one of several attempts to produce an exact realization of the set of functions given in equations (6.3). It was found that the value of K, the constant multiplier, was very sensitive to the corresponding topology, e.g. for a network composed of 5 nodes, plus reference node, and four elements, the value of K can vary from 10^{-6} to 10^{-14} for different element distributions.

3) In the example shown in Fig.[6.5], two topological modifications were made before the overall error function F ceased to reduce further. This was because all the possible virtual elements to choose from were of negative values. However, this does not prove that this set of functions cannot be realized by a network containing only six nodes. It can only be considered as an indication that such a realization is very difficult to achieve using network evolution. Further it is a strong evidence for the need to grow a new node.

6.5. Case Study 4 : The Degree of Connectivity

The initial starting network shown in Fig.[6.6] is selected to realize the following set of admittance functions,

$$\left.\begin{aligned} Y_{11} &= p + \frac{1}{6} + \frac{a_1 p}{p+1} + \frac{a_2 p}{p+2} + \frac{a_3 p}{p+3} \\ -Y_{12} &= p + \frac{1}{6} - \frac{a_1 p}{p+1} + \frac{a_2 p/2}{p+2} - \frac{a_3 p/3}{p+3} \\ Y_{22} &= p + \frac{1}{6} + \frac{a_1 p}{p+1} + \frac{a_2 p/4}{p+2} + \frac{a_3 p/4}{p+3} \end{aligned}\right\} \qquad (6.4)$$

with $a_1 = 1$, $a_2 = 15$ and $a_3 = 35$ (originally given by Lucal[34]).

The best realization for this set of functions published by Cutteridge[36] had fourteen elements and two common factors. Later, Hansen

and Wanet[36] presented an LC realization whose RC equivalent realization possesses two common factors, and uses one less network element than did Cutteridge. Existing linear network theory gives no guide to the number, if any, of excess common factors required to achieve a solution. Thus the realization of the set of functions given in equations (6.4) represents a very good case study to investigate into the concept of the degree of connectivity introduced in Section 2.5.

Another reason which made this example of particular interest is that the results shown in Fig.[6.6] are very similar to those obtained by Di Mambro[23] who employed a completely different optimization algorithm and topological modification strategies.

In solving this example, two different approaches were used, namely:

1) To start from any of the published realizations hoping that the program would remove elements and nodes until a new realization was obtained with a lesser number of nodes. This approach was unsuccessful.

2) To adopt the following approach:

i) Try to reach a solution starting from a network with the minimum number of nodes (i.e. no common factor) which could still yield the required spread of powers.

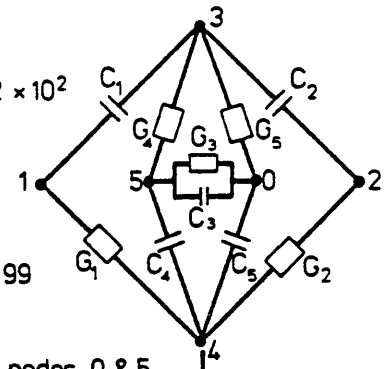
ii) If a solution was not possible with the minimum number of nodes, to investigate the reasons for failure.

iii) As described in Section 2.5, if the failure was caused by a large difference in the admittance level between input and output (i.e. the degree of connectivity), then try to find another example which could establish the concept of the degree of connectivity.

Initial element values :-

$C_i = G_i = 10^0 \; (i = 1,\ldots,5)$

Initial error function = $2{\cdot}62 \times 10^2$



(a)

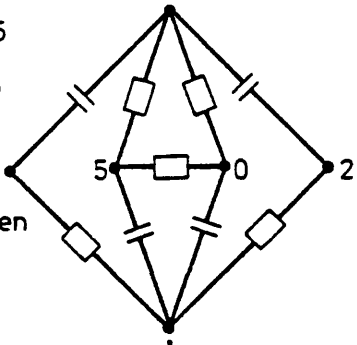
INITIAL STRUCTURE

Final error function = 85·99

REMOVE CAPACITOR
value 0·77 between nodes 0 & 5

Initial error function = 99·55

Final error function = 12·66

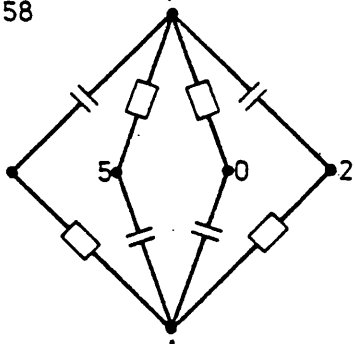


(b)

REMOVE RESISTOR
value $2{\cdot}3 \times 10^{-2}$ between
nodes 0 & 5

Initial error function = 12·58

Final error function = 3·74

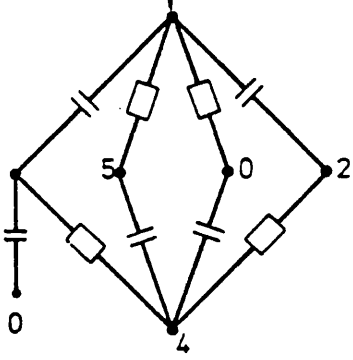


(c)

ADD CAPACITOR
value 0·617 between
nodes 0 & 1

Initial error function = 3·94

Final error function = 3·36



(d)

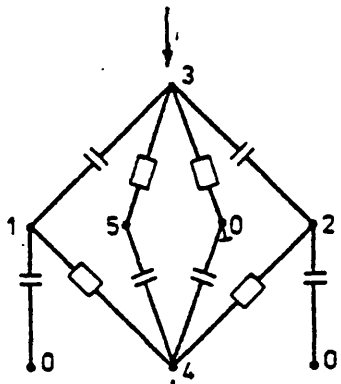
ADD CAPACITOR
value 0·752 between
nodes 0 & 2

FIG. [6·6] ATTEMPTED SYNTHESIS OF FUNCTIONS OF EQUATION (6·4)

Initial error function = 4·0

Final error function = 2·799

ADD RESISTOR
value $2{\cdot}13 \times 10^{-2}$
between nodes 0 & 2

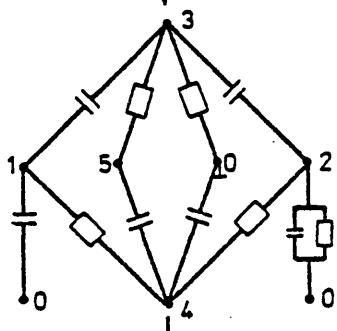


(e)

Initial error function = 2·795

Final error function = 2·781

ADD RESISTOR
value $2{\cdot}56 \times 10^{-2}$
between nodes 0 & 1

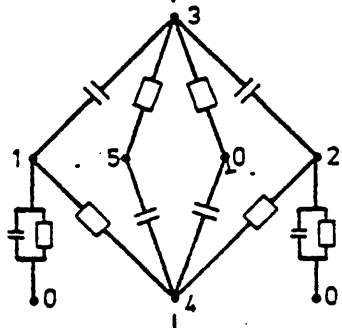


(f)

Initial error function = 2·77

Final error function = 2·75

ADD CAPACITOR
value $7{\cdot}97 \times 10^{-4}$
between nodes 0 & 3

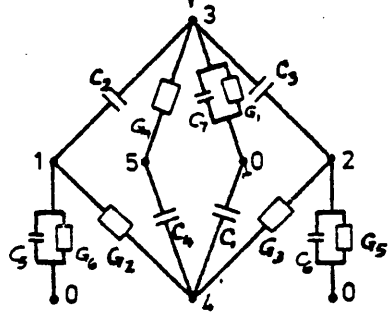


(g)

Initial error function = 2·75

Final error function = 2·75

No further improvement as the optimum values of the remaining virtual elements are all negative



(h)

FIG. [6·6] CONTINUED

$C_1 = 26$, $C_2 = 0.644$,
$C_3 = 4.381$, $C_4 = 0.386$,
$C_5 = 0.349$, $C_6 = 0.299$,
$C_7 = 0.0133$, $G_1 = 14.76$,
$G_2 = 37.84$, $G_3 = 0.1668$,
$G_4 = 0.4364$, $G_5 = 0.0136$,
$G_6 = 0.0189$

Figs.[6.6b] to [6.6h] show seven topological modifications which were made by the program before the overall error function F ceased to converge. From the computer printout produced, there were several items to consider.

1) The starting network yielded the following set of functions:

$$\left.\begin{aligned} Y_{11} = Y_{22} = \frac{\Delta_{11}}{\Delta_{1122}} &= \frac{3p^4+20p^3+34p^2+20p+3}{6p^3+22p^2+22p+6} \\ &= \frac{p}{2} + \frac{1}{2} + \frac{0.311p}{p+0.451} + \frac{0.689p}{p+2.215} \\ -Y_{12} = \frac{\Delta_{12}}{\Delta_{1122}} &= \frac{3p^4+8p^3+10p^2+8p+3}{6p^3+22p^2+22p+6} \\ &= \frac{p}{2} + \frac{1}{2} - \frac{0.311p}{p+0.451} - \frac{0.689p}{p+2.215} \end{aligned}\right\} \quad (6.5)$$

Equations (6.4) and (6.5) both have the same number of coefficients. However, equation (6.5) has one less finite pole. This is because there is a factor (p+1) common to all polynomials (see Section 2.4).

2) The optimization algorithm in the Gauss Newton section yielded very large corrections for the element values. This was attributed to the large difference in the admittance level between input and output. As a result, several elements were associated with large corrections. These corrections were oscillating between large positive and large negative values. Thus the optimization algorithm performed iterations only in the conjugate gradient section. After two conjugate gradient iterations, the current element values yielded the following set of functions:

$$Y_{11} = \frac{1.966p^4+63.729p^3+185.429p^2+136.8p+5.556}{9.229p^3+55.525p^2+83.026p+29.169}$$

$$= \frac{1}{9.22985}\left[1.96657p+1.758152 + \frac{1.742744}{(p+0.510295)} + \frac{1.09685}{(p+1.59)} + \frac{14.69211}{(p+3.89905)}\right]$$

$$-Y_{12} = \frac{1.966p^4+6.37p^3+5.979p^2+6.1694p+5.556}{9.229p^3+55.525p^2+83.026p+29.169}$$

$$= \frac{1}{9.22985}\left[1.96657p+1.758152 - \frac{1.742744}{(p+0.510295)} - \frac{0.548425}{(p+1.59)} - \frac{4.89737}{(p+3.89905)}\right]$$

$$Y_{22} = \frac{1.966p^4+14.632p^3+31.439p^2+24.227p+5.556}{9.229p^3+55.525p^2+83.026p+29.169}$$

$$= \frac{1}{9.22985}\left[1.96657p+1.758152 + \frac{1.742744}{(p+0.510295)} + \frac{0.27421}{(p+1.59)} + \frac{1.63246}{(p+3.89905)}\right] . \quad (6.6)$$

From the equations above we notice that by altering the network element values, the residues of the third finite pole increased from zero. However, the signs of the residues in the transfer function are still not as required. This could explain the instability in the Gauss Newton algorithm.

3) The value of the overall error function F decreased monotonically except:

i) When the capacitor between nodes 0 and 5 was removed. This was because its value was relatively high (see Section 4.5).

ii) When the capacitor between nodes 0 and 1 was added. This was because the addition of this capacitor contravenes the constraints imposed by the required set of functions. A capacitor between nodes 0 and 1 would make the residues of the pole at zero non-compact.

iii) Starting from Fig.[6.6e] the optimization algorithm reached a plateau. The addition of four elements only reduced the value of F from 2.799 to 2.7515 . No further element addition was possible as the optimum values of all the virtual elements were negative. This is a further justification for the technique developed for node addition. This technique was based on the idea that adding elements which would contravene the topological constraints imposed by the compactness of the poles at zero and infinity in the required set of functions, will not improve the rate of convergence sufficiently.

Comparing the results given by this example with those obtained by Di Mambro[23] we note that:

1) Di Mambro selected the topology shown in Fig.[6.6c] as a starting network.

2) The first element his program added was a capacitor between nodes 0 and 1 (as in Fig.[6.6d]).

3) Instead of adding a capacitor between nodes 0 and 2 and a resistor between nodes 0 and 1 , his program added a capacitor between nodes 0 and 3 and a resistor between nodes 3 and 4 .

4) In his program, the final value of F ceased at F = 2.77 and all the virtual element values were negative.

Hence from the above comparison we can conclude the following:

1) Using two different optimization algorithms, the same end point was reached. If the residuals of the equations are studied, then

i) The residual corresponding to the error in the coefficient of p^3 in Δ_{12} suggests that the element contributing in its 2-trees should either be removed or reduced in value.

ii) The residual corresponding to the error in the coefficient of p^4 in Δ_{12} suggests the reverse, i.e. the elements contributing in its 2-trees should be increased in number or in value.

Since many of these elements are in common to both sets of 2-trees, there is no solution within the frame of the minimum number of nodes.

2) The optimization algorithm employed in the present work is efficient and reliable even if it is compared with a more sophisticated algorithm[23]. Moreover the optimization algorithm employed in the present work is faster.

3) The topological modification techniques developed by the author are efficient.

The last part of this case study yet to be fulfilled is to find an example which has a similar set of functions and can be realized by the minimum number of nodes. This has been successfully achieved as follows:

The set of functions in the above examples (originally given by Lucal[34]) is

$$\left.\begin{aligned}
Y_{11} &= \frac{\Delta_{22}}{\Delta_{1122}} = \frac{36p^4+2058p^3+6552p^2+4638p+36}{36p^3+216p^2+396p+216} \\
&= p + \frac{1}{6} + \frac{p}{p+1} + \frac{15p}{p+2} + \frac{35p}{p+3} \\
-Y_{12} &= \frac{\Delta_{12}}{\Delta_{1122}} = \frac{36(p^4+p^3+2p^2+p+1)}{36p^3+216p^2+396p+216} \\
&= p + \frac{1}{6} - \frac{p}{p+1} + \frac{15p/2}{p+2} - \frac{35p/3}{p+3} \\
Y_{22} &= \frac{\Delta_{11}}{\Delta_{1122}} = \frac{36p^4+533p^3+1572p^2+1183p+36}{36p^3+216p^2+396p+216)} \\
&= p + \frac{1}{6} + \frac{p}{p+1} + \frac{15p/4}{p+2} + \frac{35p/9}{p+3}
\end{aligned}\right\} \qquad (6.7)$$

and the set of functions selected is

$$\left.\begin{aligned} Y_{11} &= \frac{\Delta_{22}}{\Delta_{1122}} = \frac{36p^4+1563p^3+4817p^2+3348p+36}{36p^3+216p^2+396p+216} \\ &= p + \frac{1}{6} + \frac{11p/36}{p+1} + \frac{85p/9}{p+2} + \frac{55p/2}{p+3} \\ -Y_{12} &= \frac{\Delta_{12}}{\Delta_{1122}} = \frac{36p^4+51p^3+67p^2+66p+36}{36p^3+216p^2+396p+216} \\ &= p + \frac{1}{6} - \frac{11p/36}{p+1} + \frac{85p/18}{p+2} - \frac{55p/6}{p+3} \\ Y_{22} &= \frac{\Delta_{11}}{\Delta_{1122}} = \frac{36p^4+428p^3+1157p^2+823p+36}{36p^3+216p^2+396p+216} \\ &= p + \frac{1}{6} + \frac{11p/36}{p+1} + \frac{85p/36}{p+2} + \frac{55p/18}{p+3} \end{aligned}\right\} \quad (6.8)$$

By comparing these two sets of short-circuit admittance functions it is seen that:

1) Both sets have the same poles.

2) The ratio between the relevant residues are the same.

3) The residues at the poles at zero are the same.

4) The residues at the pole at infinity are the same.

5) The value of the residues at the finite poles in equations (6.8) are less. Thus the values of the relevant coefficients in equations (6.8) are less than those of equations (6.7). Hence the difference between the admittance level at the input and output in equations (6.8) is less than that in equation (6.7). In other words, the constraints imposed by equations (6.8) on the required network can be satisfied by the degree of connectivity available within the minimum number of nodes.

The example shown in Fig.[6.7] realizes the set of functions given in equations (6.8). Further, for the same final topology given in Fig.[6.7b],

but for different element values, a group of examples can be formulated. The values of a_1, a_2 and a_3 in any of those examples should not exceed the relevant values given in equations (6.8).

Initial element values:-
all equal to unity

Initial error function =
3.37×10^{2}

Final error function =
5.59×10^{-5}

Initial error function =
5.54×10^{-5}

Final error function =
5.8×10^{-23}

Final element values:-
$C_1 = 3.4355$, $C_2 = 15.194$
$C_3 = 3.5$, $C_4 = 1.54$
$G_1 = 39.38$, $G_2 = 8.887$, $G_3 = 0.1667$, $G_4 = 2.275$, $G_5 = 1.93$, $G_6 = 0.04$.

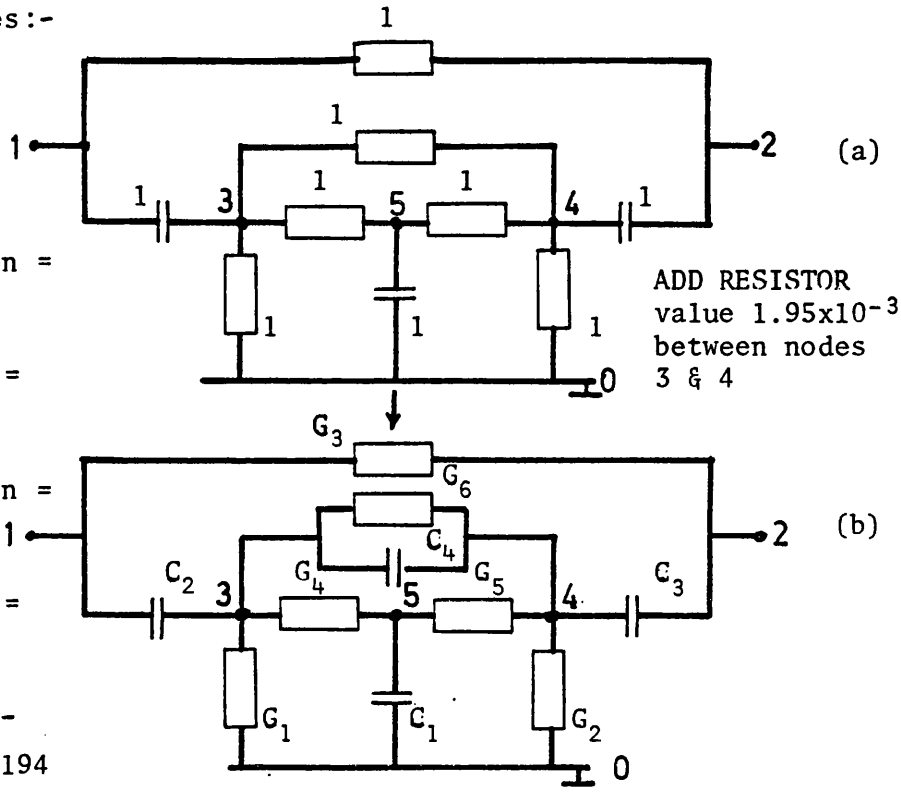


Figure [6.7] Realization of Equations (6.8)

As a result of this case study, another interesting conclusion which could be useful in practice was established as follows:

For a set of required functions, if a realization is required with a minimum number of nodes and such a realization cannot be achieved because the degree of connectivity within the minimum number of nodes is not enough, then by perturbing the values of the given set of coefficients, such a realization may be obtained within a certain tolerance.

6.6. Conclusions

After the description of each case study in this chapter, detailed conclusions were given. The general conclusion which can be obtained from the work presented in this chapter is that the coefficient matching technique is efficient and reliable. The optimization algorithm adopted and the various techniques developed for topological modifications are easily implemented and often successful.

CHAPTER 7

CASE STUDIES II : ATTEMPTED SYNTHESIS OF SOME THREE ELEMENT KIND 3-TERMINAL NETWORKS

7.1. Introduction

In the previous chapter, several case studies for the synthesis of two-element kind 3-terminal networks were described and discussed. In this chapter the author presents some case studies for the realization of 3-terminal RLC networks.

In general, in the automated synthesis program, the orders of the required set of polynomials and the number of coefficients to be matched must be correctly specified at all times during the design process, if the program is to realize a feasible final network. In RLC networks, the addition or removal of an element may alter the order of the set of functions obtained. Hence in RLC network synthesis, the order of the set of polynomials currently obtained should be checked after each topological alteration.

During the evolution of any RLC network it is possible, because of element removal, that the network may degenerate to a two-element kind network (RL, LC or RC). Further, for some RLC networks, the addition of one element may alter the orders of the obtained network polynomials. This alteration, in general, depends on the position and type of the new element. For example, the set of functions given in equations (7.1) can be realized by the RC network shown in Fig.[7.1].

$$\left.\begin{aligned} \Delta_{11} = \Delta_{22} &= \frac{1}{4}(1+5p+5p^2+p^3) \\ \Delta_{12} &= \frac{1}{4}(1+p+p^2+p^3) \\ \Delta_{1122} &= 1+2p+p^2 \end{aligned}\right\} \qquad (7.1)$$

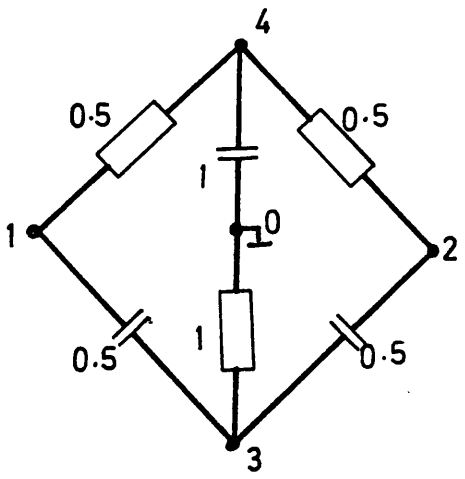


Figure [7.1] Network Realizing Equations (7.1)

The addition of one inductance between any two nodes will affect each of the four principal polynomials in a different way. Further, the way in which each polynomial is modified depends on the position in which this inductance is connected.

1) Suppose an inductance L is inserted between nodes 0 and 3 :

If an inductance of value unity is connected between nodes 0 and 3 , then the network of Fig.[7.2] is obtained. This network now realizes the four principal polynomials of equations (7.2)

$$\left.\begin{aligned} \Delta_{11} = \Delta_{22} &= \frac{1}{4p}\,(1+5p+7p^2+5p^3+p^4) \\ \Delta_{12} &= \frac{1}{4p}\,(1+p+p^2+p^3+p^4) \\ \Delta_{1122} &= \frac{1}{p}\,(1+2p+2p\ +p^3)\ . \end{aligned}\right\} \qquad (7.2)$$

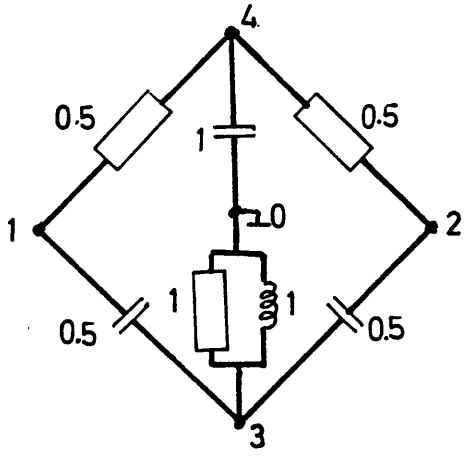


Figure [7.2] Network Realizing Equations (7.2)

2) Suppose an inductance L is inserted between nodes 0 and 4 :

If an inductance of value unity is connected between nodes 0 and 4 , then the network of Fig.[7.3] is obtained. This network now realizes the four principal polynomials of equations (7.3)

$$\left.\begin{aligned} \Delta_{11} = \Delta_{22} &= \frac{1}{4p}\,(2+5p+6p^2+5p^3+p^4) \\ \Delta_{12} &= \frac{1}{4p}\,(0+p+2p^2+p^3+p^4) \\ \Delta_{1122} &= \frac{1}{p}\,(1+2p+2p^3+p^3)\ . \end{aligned}\right\} \qquad (7.3)$$

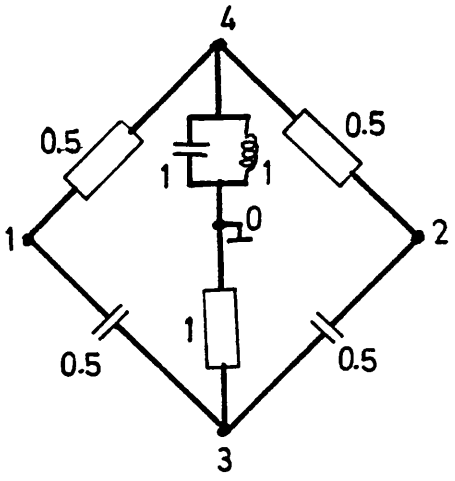


Figure [7.3] Network Realizing Equations (7.3)

Comparing equations (7.1), (7.2) and (7.3), the following is noticed:

1) The coefficients of Δ_{1122} in equations (7.2) and (7.3) are identical.

2) The number of the non-zero coefficients is increased and the range of powers of p is increased by one with respect to equations (7.1) (p^{-1} has been added to the polynomials).

3) Δ_{11} and Δ_{22} in equations (7.2) and (7.3) are polynomials of the same order but with different coefficient values.

4) The polynomials corresponding to Δ_{12} in equations (7.2) and (7.3) are different in order and in the coefficient values.

The increase in the range of powers of p in the polynomials shows the need for a continuous check on the order and the number of coefficients of the set of polynomials obtained during the evolution of the required network. Clearly, in the example given above, if the solitary inductance in the networks of Figs.[7.2] and [7.3] is removed, then the remainder is identical to the network of Fig.[7.1] in both cases. Further, the realized set of network polynomials will be that of equations(7.1). On the other hand, adding more inductance to the network shown in Fig.[7.1] may increase the range of powers of p in the set of polynomials obtained.

In practice, it is rather difficult to formulate a systematic technique to check the orders of the set of polynomials currently obtained during the evolution process, especially for sophisticated RLC networks. However, the author developed the following empirical check. The program calculates the current values of all coefficients for polynomials of the maximum order possible, as determined by the number of the current nodes. Hence during network evolution and after each topological alteration,

1) If the value of a coefficient that is required to be non-zero is calculated to be below a certain prescribed value (e.g. 10^{-6} say), this is considered as an indication that the program suggested a wrong topological alteration. In this case, the suggested topological alteration is to be abandoned and the search for another topological alteration is to be carried out.

2) If the value of one or more of the zero coefficients increases above a certain minimum value (e.g. 10^{-6}), then this coefficient(s) should be included with the set of coefficients to be matched and the orders of the required set of polynomials to be matched will be altered accordingly.

3) The coefficients of the four principal polynomials yielded by the current network are printed out after every change in topology.

7.2. Case Study 1 : A simple RLC Realization

The example shown in Fig.[7.4] is to realize the set of functions given by equations (7.4).

$$\left.\begin{aligned}
\Delta_{11} &= \frac{1}{p^2}\,(8+33p+29p^2+12p^3+4p^4) \\
\Delta_{12} &= \frac{1}{p^2}\,(2+13p+14p^2+4p^3+4p^4) \\
\Delta_{22} &= \frac{1}{p^2}\,(2+19p+34p^2+19p^3+12p^4) \\
\Delta_{1122} &= \frac{1}{p^2}\,(2+17p+13p^2+7p^3+2p^4)\ .
\end{aligned}\right\} \qquad (7.4)$$

The corresponding set of short-circuit admittance functions, if written in the pole-zero form, is

$$\left.\begin{aligned}
Y_{11} &= \frac{\Delta_{22}}{\Delta_{1122}} = \frac{6(p+0.13611)(p+0.19165)(p+0.655+j2.037)(p+0.665-j2.037)}{(p+0.1296)(p+2.207)(p+0.582+j.1776)(p+0.582-j1.776)} \\
-Y_{12} &= \frac{\Delta_{12}}{\Delta_{1122}} = \frac{2(p+0.19165)(p+0.78434)(p+0.012+j1.824)(p+0.012-j1.824)}{(p+0.1296)(p+2.207+(p+0.582+j1.776)(p+0.582-j1.776)} \\
Y_{22} &= \frac{\Delta_{11}}{\Delta_{1122}} = \frac{2(p+0.3233)(p+1.3466)(p+0.665+j2.037)(p+0.665-j2.037)}{(p+0.1296)(p+2.207)(p+0.582+j1.776)(p+0.582-j1.776)}
\end{aligned}\right\} \qquad (7.5)$$

From equations (7.5), this set of admittance functions has a conjugate complex poles and zeros in each of the admittance functions. Hence it cannot be realized by any two-element kind network.

From the starting network of Fig.[7.4a] the synthesis program converges to a local minimum in six Gauss Newton iterations. From 24 virtual elements the program selected the resistor between nodes 0 and 4 . This resistor corresponds to maximum optimum value and maximum reduction in the value of the overall error function. A solution was achieved after inserting the

Initial element values :-

$C_i = G_i = \frac{1}{L_i} = 10^{\circ}$ (i=1,2)

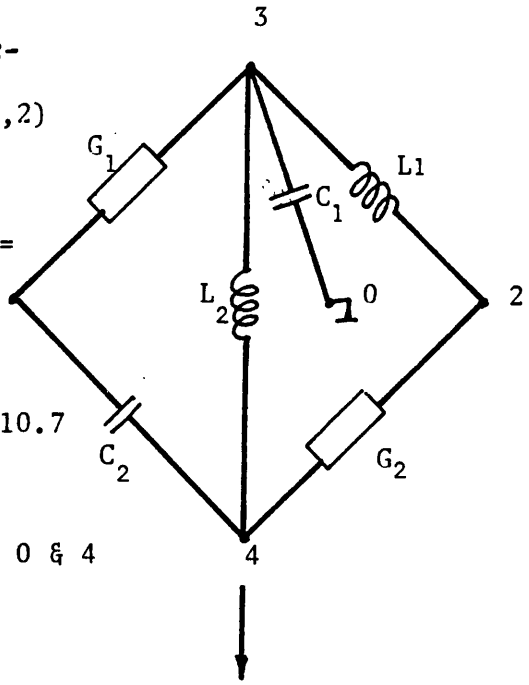
Initial error function = 20.66

INITIAL STRUCTURE

Final error function = 10.7

ADD RESISTOR

Value 2.8 between nodes 0 & 4



(a)

Initial error function = 4.34

FINAL STRUCTURE

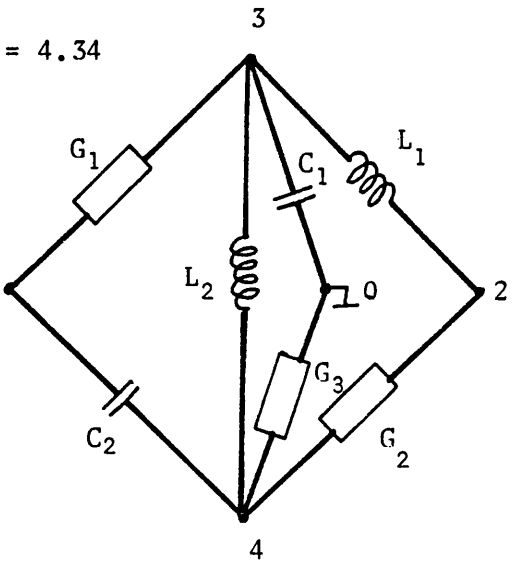
Final error function = 2.07×10^{-25}

Final element values :-

$C_1 = 1$, $C_2 = 2$,

$G_1 = 1$, $G_2 = 2$, $G_3 = 3$

$L_1 = 1$, $L_2 = 0.5$



(b)

Figure [7.4] Network Realizing Equations (7.4)

new element, in five Gauss Newton iterations which is shown in Fig.[7.4b].

7.3. Case Study 2 : RLC Realization by Node Addition

For every statement relating to a 3-terminal RC network, corresponding statements can be formulated for RL and LC networks without mutual inductances. This follows from the well known correspondence between networks of any two kinds of element, chosen from the three kinds: inductance, capacitance and resistance. The range of powers of p in the set of functions representing two-element kind networks will depend on the type of elements selected. The example given in Section 7.2 cannot be realized by any two-element kind network. However, any set of admittance functions which has a two-element kind realization, can also be realized by an RLC network. An RLC equivalent for a two-element kind network may be possible, but it cannot be obtained from a straightforward transformation. Further, an RLC network can yield polynomials satisfying the range of powers of p corresponding to any two-element kind network containing the same number of nodes. Moreover, an RLC network can yield twice the range of powers of p corresponding to any RL or RC network containing the same number of nodes.

Fialkow[8] published a set of short circuit admittance functions (equations (6.1)). The transfer admittance of this set of functions had a negative coefficient in the numerator. Fialkow proved that if this set of functions is realized by an RC network then,

1) The realization must be a non-series-parallel network with no series-parallel equivalent.

2) At least one common factor of the form $(p+\alpha)$ $\left(\frac{1.14}{3} < \alpha < \frac{197.176}{1.14}\right)$ is necessary in order to mask the existence of the negative coefficient.

3) With the addition of one common factor, Δ_{1122} becomes a full cubic polynomials in p, whence any two-element kind realization must have at least five nodes plus the reference node, node zero.

The RL equivalent to Fialkow's original set of functions (equations (6.1)), obtained by changing every capacitor present in his realization to a resistor of the same value and each resistor in his realization to an inverse inductance of the same value, is given in equations (7.6)

$$\left.\begin{aligned} Y_{11} = Y_{22} &= \frac{\theta(p)}{\theta(p)} \quad \frac{(1197p^3+56613.14p^2+28368.584p+191.184)}{p(800000p^2+408000p+3840)} \\ -Y_{12} &= \frac{\theta(p)}{\theta(p)} \quad \frac{(3p^3-1.14p^2+197.176p+77.616)}{p(800000p^2+408000p+3840)} \end{aligned}\right\} \quad (7.6)$$

An investigation into an RLC realization for Fialkow's transformed set of functions given by equations (7.6) is of particular interest for the following reasons:

1) Several RC realizations for Fialkow's set of functions, one of which had not previously been published, were given in Chapter 6.

2) Equations (7.6) has a different range of powers of p.

3) In contrast with RC or RL networks, RLC networks can yield polynomials of the required order with four nodes plus the reference node, node zero.

4) Fialkow[62] conjectured the existence of RLC non-series-parallel networks, with no series-parallel equivalent. Thus, could an RLC realization for equations (7.6) be a non-series-parallel network?

5) If an exact realization with only two internal nodes cannot be obtained, can an RLC realization with three internal nodes exist? If one exists, is it series-parallel or non-series-parallel?

Unfortunately, theoretical answers to the above questions are very difficult to obtain. Hence the computer program developed by the author and employing the techniques described in the present work was used to investigate the above questions.

In equations (7.6), $\theta(p)$ is a polynomial in p, with real positive coefficients, representing the product of the common factors which are needed to mask the presence of the negative coefficient appearing in the numerator of the transfer function $(-Y_{12})$. For an RLC network, $\theta(p)$ can take one of the following forms:

(i)	$(p+\alpha)$	linear factor
(ii)	$(p+\alpha)(p+\beta)$	two linear factors
(iii)	$(p+\alpha)(p+\beta)(p+\gamma)$	three linear factors
(iv)	$(p^2+2\rho\cos\theta+\rho^2)$	conjugate complex pair of factors
(v)	$(p+\alpha)(p^2+2\rho\cos\theta+\rho^2)$	linear factor and conjugate complex pair of factors.

7.3.1. The Proposed Design Strategy

In view of the success achieved in synthesizing several two-element kind networks, some of which were described in the previous chapter, it was anticipated that the same techniques would also be suitable for synthesizing RLC networks, with perhaps some minor modifications. This anticipation was justified for simple RLC networks, e.g. the example shown in Section 7.2. For the set of functions given in equations (7.6) no RLC realization had previously been published. Thus no reliable estimate could be made of the size or order of the network that would be most likely to yield a solution. Further, experience with adjusting two or more common factors simultaneously during network optimization, in cases where the initial starting network was remote from any acceptable solution, was limited.

A starting network was selected which contained the minimum number of

nodes and element which would yield the required spread of powers in the required polynomials. It was hoped that by adjusting the values of the common factors and altering the starting topology, a technique which proved successful in synthesizing two-element kind networks, a solution would be achieved without undue difficulty.

7.3.2. The Design Strategy Adopted

The optimization algorithm worked well in cases where there were up to 21 independent variables and a feasible topology (18 elements, 3 common factors and the correct number of nodes). Further, the program converged to a solution without undue difficulty where there were up to two extra nodes (Section 6.3). However, the case study considered in this section proved to be far more difficult than any other problem experienced before. The main reasons for this difficulty were:

1) An RLC realization for the set of functions given in equations (7.6) was not known before. Thus, the form of the final solution, or even whether a solution was possible or not, was not known.

2) The number of necessary common factors and nodes for a solution were not known.

3) The range in which the common factors should vary in favour of topological alterations were not defined in this particular problem.

For any set of short-circuit admittance functions, if a topology and one value of common factor which can yield a solution were known then, it is easy to obtain several different realizations for the same value of the common factor (see Section 6.2 and Krzeczkowski[24]). Further, it is also possible to obtain realizations for a range of common factor values around the value previously known[24].

The problem becomes more difficult if either the common factor value or a solution topology was not known. Potentially, the problem becomes very difficult if neither the topology nor the common factor value for a solution were known. In the problem considered here, not only a solution topology, and a value of a common factor are unknown, but also the necessary number of nodes is unknown and there are up to three common factors with unknown values or range.

In order to overcome these difficulties, several exploratory runs were made to gain better understanding of the way in which the program would react when remote from a solution ($F > 10^{-6}$). The strategy of adjusting the value of the common factors by the optimization algorithm (see Section 4.7) was abandoned and the following alternative strategy was adopted:

1) For a particular topology and a fixed set of element values, select the common factor values which give the lowest value of the overall error function F.

2) Fix these values of the common factors.

3) Adjust the network element values and alter the network topology using the synthesis program with the values of the common factors obtained in step 2).

4) If the rate of decrease of the overall error function F per iteration is less than 0.01% or F is greater than 10^{-6}, return to step 1).

5) If F is less than 10^{-6}, adjust the values of the common factors using the optimization algorithm (see Section 4.7).

6) If the rate of decrease of F is $\geq 0.01\%$ continue until a solution is obtained, otherwise return to step 1).

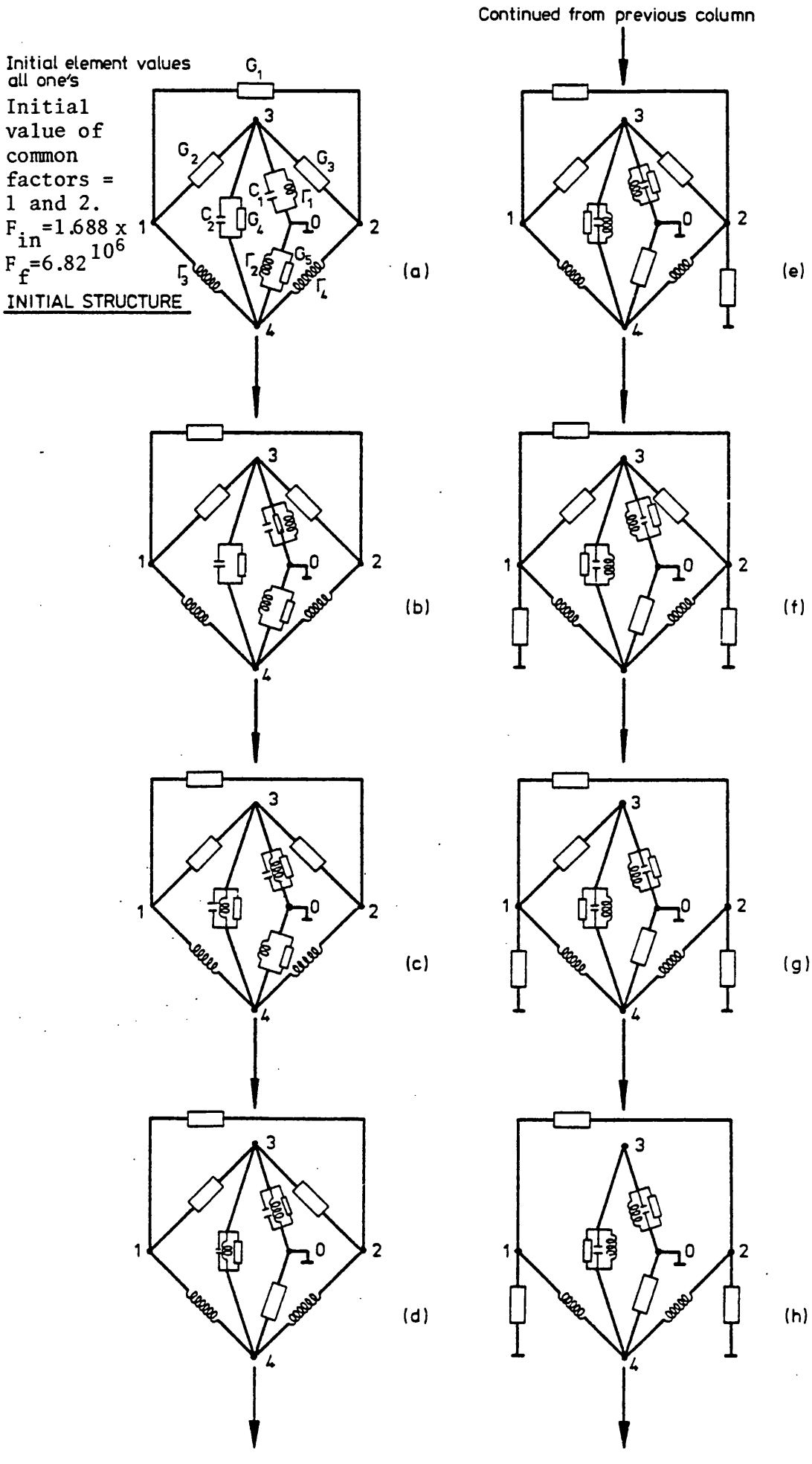


FIG [7·5] STRUCTURAL CHANGES LEADING TO RLC REALIZATION OF FUNCTIONS OF EQUATIONS (7·6)

(a)

Common factors = 1.1 and 15.2

Final error function = 5.71

ADD RESISTOR

value 2.29 between

nodes 0 and 3

(b)

Initial error function = 5.70

Common factors = 1.65 and 16

Final error function = 4.32

ADD INDUCTANCE

value 2.208 between

nodes 3 and 4

(c)

Initial error function = 4.01

Common factors = 1.65 and 19

Final error function = 3.154

REMOVE INDUCTANCE

value 0.037 between

nodes 0 and 4

(d)

Initial error function = 2.98

Common factors = 1.65 and 20

Final error function = 5.068×10^{-3}

ADD RESISTOR

value 1.2 between

nodes 0 and 2

(e)

Initial error function = 5.066×10^{-3}

Common factors = 2 and 25

Final error function = 4.14×10^{-3}

ADD RESISTOR

value = 1.04 between

nodes 0 and 1

(f)

Initial error function = 4.1×10^{-3}

Common factors = 9 and 50

Final error function = 3.04×10^{-3}

REMOVE RESISTOR

value 2.5×10^{-4} between

nodes 2 and 3

(g)

Initial error function = 3×10^{-3}

Common factors = 12 and 55

Final error function = 2.01×10^{-3}

REMOVE RESISTOR

value 3.68×10^{-5} between

nodes 1 and 3

(h)

Initial error function = 2×10^{-3}

Common factors = 15 and 55

Final error function = 1.89×10^{-3}

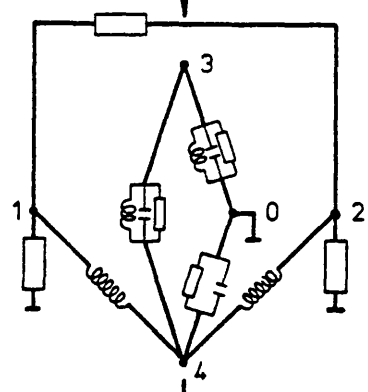
ADD CAPACITOR

value 2.4×10^{-4} between

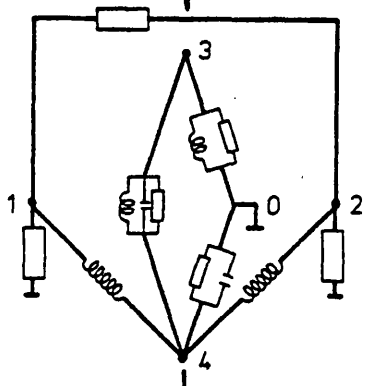
nodes 0 and 4

Figure [7.5] - Comments

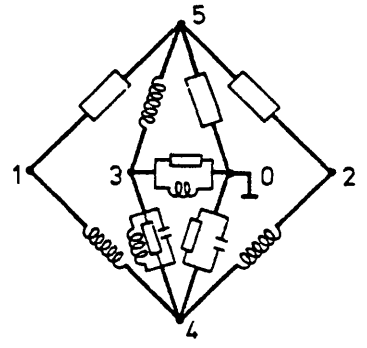
Continued from previous page



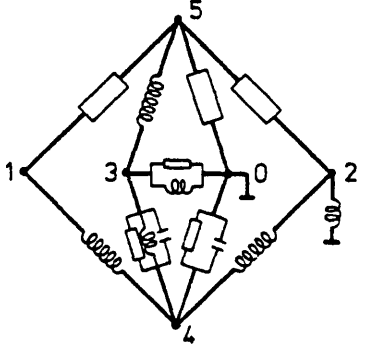
(i)



(j)

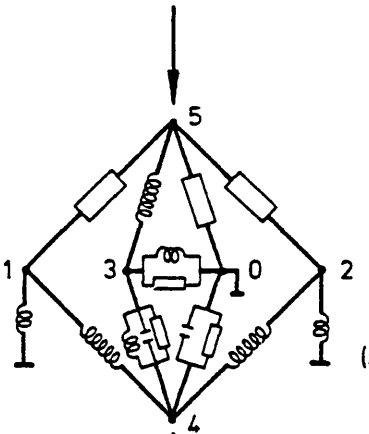


(k)

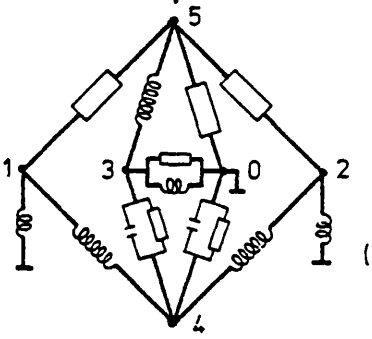


(l)

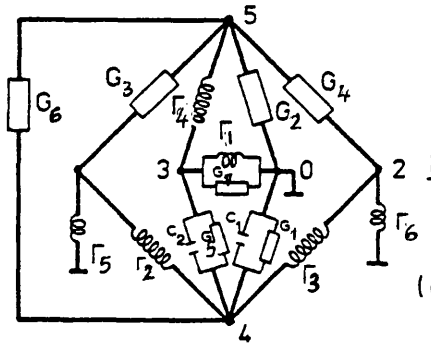
Continued from previous column



(m)



(n)



(o)

FINAL STRUCTURE

Final error function = 8.3×10^{-35}

Final value of common factors =
2.951.., 2.95101.. and 54.099 ..

Final element values :-

C_1=0.1407..,C_2=1.3243..,G_1=4.5219..,
G_2=0.2857..,G_3=G_4=0.0015,G_5=3.5517,
G_6=0.3112..,G_7=2.8744..,Γ_1=2.5589,
Γ_2=Γ_3=0.040425,Γ_4=1.2222..,Γ_5=Γ_6=0.029575.

FIG.[7·5] continued.

(i)

Initial error function = 9.94×10^{-4}

Common factors = 40 and 60

Final error function = 8.89×10^{-4}

REMOVE CAPACITOR

value 9.86×10^{-12} between

nodes 0 and 3

(j)

Initial error function = 8.78×10^{-4}

Common factors = 65 and 75

Final error function = 1.74×10^{-4}

ADD NODE 5 and INDUCTANCE

value 6.12×10^{-1} between

nodes 3 and 5

(k)

Initial error function = 1.06

Common factors = 1, 5 and 50

Final error functions = 2.12×10^{-2}

ADD INDUCTANCE

value 0.11 between

nodes 0 and 2

(l)

Initial error function = 2.01×10^{-2}

Common factors = 1.8, 3.5 and 50

Final error function = 5.7×10^{-4}

ADD INDUCTANCE

value 2.2×10^{-2} between

nodes 0 and 1

(m)

Initial error function = 5.2×10^{-4}

Common factors = 2,3 and 50

Final error function = 1.34×10^{-6}

REMOVE INDUCTANCE

value 3.37×10^{-2} between

nodes 3 and 4

(n)

Initial error function = 1.14×10^{-6}

Common factors = 2.5, 2.6 and 51

Final error function = 3.9×10^{-7}

ADD RESISTOR

value 0.12 between

nodes 4 and 5

(o)

Initial error function = 4.8×10^{-8}

Common factors = 2.9, 2.95 and 54

Figure [7.5] - Comments - Continued

The final strategy adopted for adjusting the value of the common factors was to use the strategy described above as long as F was greater than 10^{-6}. This involved several exploratory runs before any decision was made on the next topological modification and the optimum values of the common factors (see Section 4.7.2).

Figs.[7.5b] to [7.5.0] show the fourteen topological modifications which were made before a solution was attained. The modifications include the addition of one extra node. From the detailed analysis of the computer printout produced, there were several items of interest:

1) The final network which yielded the solution ($F < 10^{-33}$), Fig.[7.5.0] was an RLC non-series-parallel network containing five nodes plus reference node, node 0 and fifteen elements.

2) The total run time required and the number of topological alterations needed to reach a local minimum with two internal nodes were much greater than those with three internal nodes.

3) No element was added and then later removed.

4) The lowest value of F achieved before adding a new node ($F = 1.74 \times 10^{-4}$) was obtained with a series-parallel network.

5) The common factors values at this stage ($F = 1.74 \times 10^{-4}$) were high (65 and 75).

6) After the addition of the new node, the values of the common factors were greatly reduced (5 and 50).

7) The final values of two of the common factors were very similar (2.951 and 2.95101).

8) The values of the initial error function given in Fig.[7.5] are the values obtained after the selection of the optimum values for the common factors.

9) Whilst the final value of F obtained with the network of Fig.[7.5.0] was less than 10^{-20}, which means that the coefficients of the required network polynomials were matched correctly to at least the first ten significant figures, there is no completely satisfactory way of determining, when using a finite word length digital computer, whether a true solution or a very good quasi-solution has been obtained. However, a further investigation was carried out using a double precision version of the program on a CDC 7600 computer. A value of F was obtained as low as 10^{-35}. Even so, the values of several internal network elements were still seemingly irrational.

Two checks were made to justify the results obtained in Fig.[7.5] namely:

1) Starting from the topology shown in Fig.[7.5.0] and with the values of the three common factors fixed, for different initial element values, the program converged to the same solution each time. Further, the final value of the overall error function obtained at the solution was always of the same order ($F < 10^{-33}$).

2) Starting from the topology and element values shown in Fig. [7.5.0] with the element values fixed, for different initial values of the common factors, the program converged to the same solution each time. Further, the final values of the common factors obtained at the solution were always identical to those shown in Fig.[7.5.0] to at least the first fourteen significant figures.

In both cases, a solution was obtained in a very short time (less than two minutes on the CDC 7600 computer).

7.4. Conclusion

The results obtained in Section 7.3 are a further justification of the feasibility and efficiency of the coefficient matching technique in solving very difficult problems. In Section 7.3 substantial topological alterations were made (up to 14 topological alterations) until the final solution, which was not known before, was obtained. The final network was an RLC non-series-parallel network. However, this does not prove that the set of functions given in equation (7.6) cannot be realized by an RLC series-parallel network. It can only be considered as an indication that such a realization is very unlikely, if not impossible, to achieve using network evolution techniques. Further, it strongly demonstrates the capability of the coefficient matching technique for synthesizing networks which cannot be designed using theoretical or classical methods.

CHAPTER 8

CONCLUSION

8.1. General Conclusion

In this thesis, the automated design of lumped, linear, passive, 3-terminal networks, with no mutual inductances, has been described. The design procedure is based on the coefficient matching technique. The feasibility of this technique, for the synthesis of such networks, had previously been pointed out[14,15,22,23,24,26,31], and this method had been successfully used by earlier designers for the synthesis of RC networks, though with somewhat limited topological modifications[23,24]. In the present work, the scope of this technique has been further extended and generalized, on the basis of which fully automated design of a much wider range of problems can be achieved. A brief summation of the contents of each chapter of the thesis is as follows:

Chapter 1, introduces the general concept of automated network design using the coefficient matching technique.

In Chapter 2, the significance of the information obtained from the various equivalent forms of admittance functions has been pointed out. The important role played by common factors in network synthesis, has been discussed and a new concept of the degree of connectivity is introduced. This information is very helpful in estimating a feasible starting network.

In Chapter 3, a review of the coefficient matching technique and the optimization algorithm used are discussed, which form the basis of the automated network design technique adopted in the present work.

During the process of automated network design, topological modifications may be necessary at several stages. Methods for the automatic modification of the network topology, by node addition and node removal, as well as element addition and element removal, have been discussed in

Chapter 4. These techniques overcome most of the limitations of the previous work[23,24]. Adjustment of the common factors increases the range of feasible designs. Several techniques for the adjustment of different types of common factors, as for example linear common factors, conjugate complex common factors and quadratic common factors, were introduced by the author. The influence of more than one common factor was hitherto unknown. The techniques developed and discussed in Chapter 4 increase the efficiency and the scope of the automated network design.

A brief description of the Fortran program, developed by the author, which employs the techniques described in Chapters 3 and 4, has been presented in Chapter 5. This program proved to be rapid, accurate and efficient.

The effectiveness of the program is illustrated by the synthesis of several networks in Chapters 6 and 7. Examples of automatic removal of more than one node, and an example of the realization of a series-parallel quasi-equivalent of a non-series-parallel network have been presented in Chapter 6.

In Chapter 7, an example of non-series-parallel RLC realization, involving automatic node addition, has been described— RLC non-series-parallel realization was hitherto unknown.

The realization given in Chapters 6 and 7, some of which could not be achieved by using classical methods, highlight the generality and the advantages of the automated network design technique presented in this thesis.

8.2. Achievements

The author's main achievements, in the present work, may be summarized as follows:

1) Selection of a suitable starting network on the basis of the

information obtained from the equivalent forms of the short-circuit admittance functions (Section 2.4).

2) Establishing the concept of the degree of connectivity on the basis of the relationship between the coefficients of the required set of functions and the complexity of the required network (Section 2.5 and Appendix A). This concept is also very useful for the selection of a suitable starting network.

3) New criteria for automatic element removal (Section 4.5).

4) New techniques for automatic node removal (Section 4.5).

5) New criteria for automatic element addition (Section 4.6).

6) New techniques for automatic node addition (Section 4.6).

7) A new technique for estimating the value of the new element to be added (Section 4.6).

8) The range of feasible designs was increased by introducing the facility to alter the order of the required set of functions during network evolution (Section 4.7).

9) The ability to vary up to three common factors of different forms simultaneously (Section 4.7).

10) The development of computer-aided electrical network design program in Fortran IV (Chapter 5).

11) Ability to deal with difficult problems involving substantial topological alteration (Chapters 6 and 7).

12) The successful design of an RLC non-series-parallel, 3-terminal network involving node addition (Section 7.3).

13) Establishing the transfer phase-frequency characteristic of a non-series-parallel network (Appendix B).

8.3. Future Work

The automatic design of electrical networks, in general, is a very complex problem. Thus the aforementioned achievements are not 'absolutes', in the sense that they are incapable of further improvement, or the collections of methods incorporated would always prove successful for all lumped, linear, 3-terminal network design problems. The efficiency and scope of the program developed, can be further improved. For short-term studies, the modifications in the following areas are suggested; these modifications would improve the efficiency and the effectiveness of the present work to a considerable degree.

1) Optimization: The performance of the current optimization algorithm could be improved by using the special properties of multilinear functions[65].

2) Matching technique: The significance of the information obtained from the different but equivalent forms of the admittance functions was pointed out in Section 2.4. In the present work, this information was used only to select the starting networks and to check that the current network topology still yielded functions of the required order, after each topological modification. Further investigation is necessary in order to make maximum use of this information, in such operations as the selection of elements for addition or removal.

3) Topological Modifications: Although the program developed overcame most of the limitations of the previous works[23,24] in this field, there are yet other areas for further developments namely:

i) The program has no facilities for multiple topological alteration. Thus an investigation into the possibilities of adding more than one element at a time would be useful to undertake. Also, techniques for removing an element and adding another element, simultaneously, as well as adding a node and removing another, could be helpful.

ii) In Section 2.5 and Appendix A , the concept of the degree of connectivity was established. An investigation into an analytical formulation for this concept would be helpful as it could give a better estimate of the required number of nodes (and hence, the number of extra common factors).

4) Using the Program: During the course of this research, it was observed that for some types of network, a realization could be obtained more rapidly if a simple starting network was selected containing the minimum number of nodes and elements. In these cases, a solution was obtained, mainly, by adding elements but sometimes nodes as well. On the other hand, for other types of network, a solution could be obtained faster if the starting network contained more elements, and perhaps more nodes also, than those required. In these cases, a solution was obtained, mainly, by removing elements, but possibly nodes as well. Hence, further investigation is required for the selection of an optimum starting network which would converge most readily to a feasible solution.

Pole-zero and frequency response matching techniques have been successfully used for the realization of certain sophisticated networks, e.g. simulation of coaxial cable[21]. Solution of such networks could not be easily achieved using the coefficient matching technique. On the other hand, with a number of problems for which the coefficient matching technique has been successful, the pole-zero and frequency response matching techniques

have failed[21]. Hence, further investigation is necessary so that the limitations of these methods may be properly assessed, and a better and efficient automated network design program may be developed, incorporating the virtues of each of these techniques.

For long term, future studies, the following items could be areas of research:

1) In order to make the coefficient matching technique more useful for industrial applications, it would be helpful to include constraints on network element values, network sensitivities, the ability to include parasitics and to adjust the network element values and/or topology accordingly.

2) To investigate the possibility of including active elements and mutual inductance.

3) The development of a new optimization algorithm capable of solving unconstrained, underdefined problems is very important. One way to develop such an algorithm, could be by using random techniques[43-45]. Since random techniques are not effective in the region of the solution, it is probable that the approach will be hybrid in nature. Initially, the error space would be studied in a random fashion to locate the areas of possible solutions. Finally, to locate the best solution, conventional analytical methods (e.g. Gauss Newton), might be used to investigate into each area of possible solution.

APPENDIX A

THE CONCEPT OF THE DEGREE OF CONNECTIVITY

The concept of the degree of connectivity was introduced in Chapter 2. In this Appendix a further theoretical justification is presented.

Consider the network of Fig.[1] as a possible realization for the following four principal polynomials

$$\left.\begin{aligned} \Delta_{11} = \Delta_{22} &= K(A_0+A_1p+A_2p^2) \\ \Delta_{12} &= K(A_0+A_3p+A_2p^2) \\ \Delta_{1122} &= K(A_4+A_5p) \end{aligned}\right\} \qquad (1)$$

where $A_i > 0$, $i = 0, \ldots,5$ and $K > 0$.

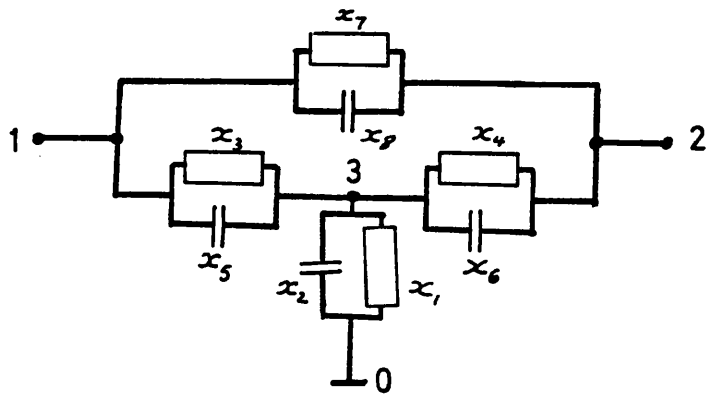


Figure [1] A Suggested Realization Topology for Equations (1)

The network has eight elements. There are, therefore, a total of eight variables, $x_1, x_2, \ldots,x_8$ whose values are to be obtained in terms of the coefficients of the polynomials of equations (1). The admittance matrix of the network is

$$\begin{bmatrix} [(x_5+x_8)p+(x_3+x_7)] & -(x_8p+x_7) & -(x_5p+x_3) \\ -(x_8p+x_7) & [(x_6+x_8)p+(x_4+x_7)] & -(x_6p+x_4) \\ -(x_5p+x_3) & -(x_6p+x_4) & [(x_2+x_5+x_6)p+(x_1+x_3+x_4)] \end{bmatrix} \quad (2)$$

Were the cofactors Δ_{11}, Δ_{12}, Δ_{22} and Δ_{1122} to be evaluated directly from the admittance matrix, a total of eleven simultaneous equations could be constructed, all multilinear functions of x_1, x_2,...,x_8 and K(Δ_{11},Δ_{12} and Δ_{22} are quadratics, Δ_{1122} is linear), by equating the actual coefficient values with the values required. Alternatively, equating the coefficients of the cofactors given in equations (1) with the 2-tree and 3-tree admittance products calculated directly from the elements of the network of Fig.[1], the same eleven equations in nine unknowns would be obtained. The network analysis problem requires the solution of eleven equations in nine variables (eight elements and the constant multiplier K). However, the number of equations to solve can be reduced if the information obtained in Chapter 2 is used and the constraints imposed by the given set of polynomials are considered, namely:

1) From equations (1), it can be shown that $\Delta_{11} = \Delta_{22}$ if and only if $x_3 = x_4$ and $x_5 = x_6$. When the latter conditions are satisfied, the equations yielded by Δ_{11} are identical to those yielded by Δ_{22}. So Δ_{11} or Δ_{22} is redundant. Hence, there are only eight equations to be solved in seven variables.

2) Because the coefficients of the highest powers of p in Δ_{11} and Δ_{12} are equal ($\Delta_{11}(p^2) = \Delta_{12}(p^2) = KA_2$) then:

either (i) $x_2 = 0$ and $x_5 = x_6 > 0$,

or (ii) $x_2 > 0$ and $x_5 = x_6 = 0$.

3) Because the coefficients of p^o in Δ_{11} and Δ_{12} are equal ($\Delta_{11}(p^o) = \Delta_{12}(p^o) = KA_o$) then:

either (iii) $x_1 = 0$ and $x_3 = x_4 > 0$,

or (iv) $x_1 > 0$ and $x_3 = x_4 = 0$.

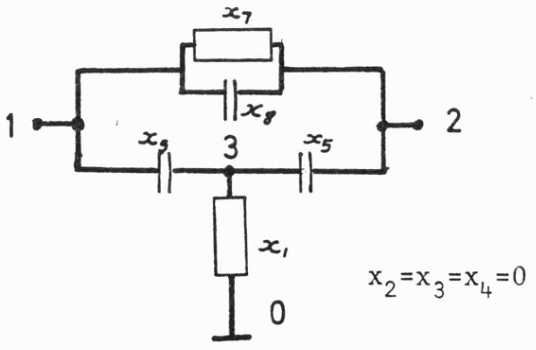
4) From (i), (iii) and Fig.[1] , if $x_2 = 0$ and $x_1 = 0$, then the network becomes disconnected. Similarly, from (ii), (iv) and Fig.[1] , if $x_3 = x_4 = x_5 = x_6 = 0$, then the network becomes disconnected.

From (i) and (iv) , if $x_2 = x_3 = x_4 = 0$, then a feasible topology is obtained as shown in Fig.[2.a], Case 1.

From (ii) and (iii) , if $x_1 = x_5 = x_6 = 0$, then a feasible topology is obtained as shown in Fig.[2.b], Case 2.

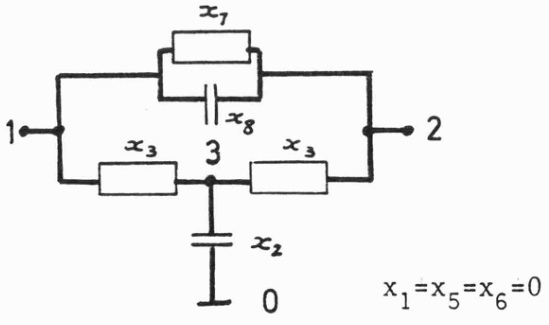
The two networks of Fig.[2] have the only topologies which can yield the functions of equations (1), if the constraint that the networks have no more than four nodes is imposed.

Case 1



(a)

Case 2



(b)

Figure [2] The Two Feasible Topologies Realizing Equations (1)

For cases 1 and 2 , there are only six equations to be solved in five variables.

Case 1

From item 4) above, the network of Fig.[2.a] satisfies the conditions 1), (i) and (iv). The network element values and the normalizing variable, K, can be obtained by solving six equations in five unknowns. From equations (1) and (2) and Fig.[2.a] we want to solve

$$\left.\begin{aligned} \Delta_{12} &= K(A_2p^2+A_3p+A_o) \\ \Delta_{1122} &= K(A_5p+A_4) \\ \Delta_{11}-\Delta_{12} &= (A_1-A_3)p\,K \end{aligned}\right\} \qquad (3)$$

Construct six equations

$$KA_2 = x_5^2+2x_5x_8 \qquad (3.1)$$

$$KA_3 = x_1x_8+2x_5x_7 \qquad (3.2)$$

$$KA_o = x_1x_7 \qquad (3.3)$$

$$KA_5 = 2x_5 \qquad (3.4)$$

$$KA_4 = x_1 \qquad (3.5)$$

$$K(A_1-A_3) = x_1x_5. \qquad (3.6)$$

From equations (3.3) and (3.5)

$$x_7 = A_o/A_4 \ . \qquad (3.7)$$

From equations (3.4) and (3.6)

$$x_1 = 2(A_1-A_3)/A_5 \ . \qquad (3.8)$$

From equations (3.4) and (3.5)

$$K = 2(A_1-A_3)/A_4A_5 \ . \qquad (3.9)$$

From equations (3.5) and (3.6)

$$x_5 = (A_1 - A_3)/A_4 \ . \tag{3.10}$$

From equations (3.1), (3.9) and (3.10)

$$x_8 = \frac{A_2}{A_5} - \frac{A_1 - A_3}{2A_4} \ . \tag{3.11}$$

But from equations (3.2), (3.7), (3.8) and (3.9)

$$x_8 = \frac{A_3}{A_4} - \frac{A_o A_5}{A_4^2} \ . \tag{3.12}$$

From equations (3.11) and (3.12), these sets of equations are consistent if and only if

1. Either $x_8 > 0$, when

$$A_4(2A_2A_4 - (A_1 - A_3)A_5) = 2A_5(A_3A_4 - A_oA_5)$$

or 2. $x_8 = 0$, when

$$A_oA_5 = A_3A_4$$

and $2A_2A_4 = A_5(A_1 - A_3)$, see Fig.[3] .

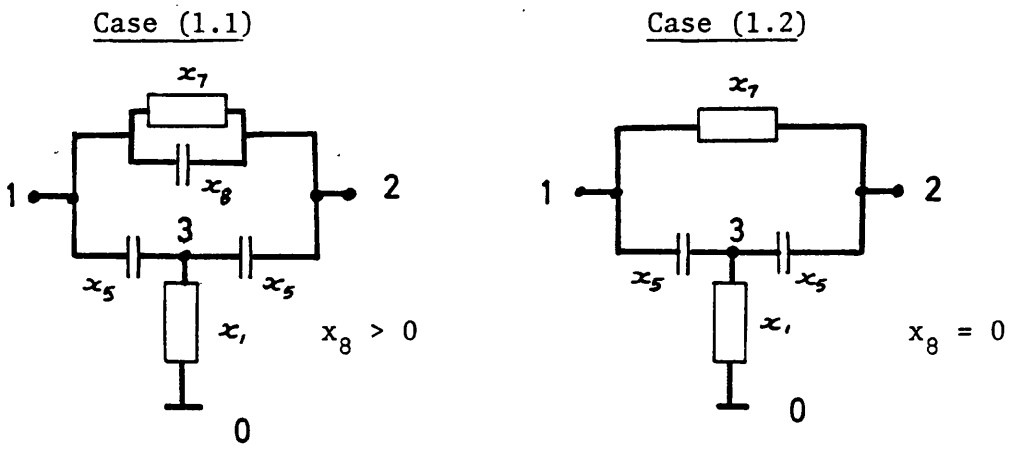


Figure [3] The two feasible topologies resulting from Case 1 to realize equations (1)

Case 2

From item 4), the network shown in Fig.[2.b] satisfies the conditions 1), (ii) and iii) . The network element values and the normalizing variable, K, can be obtained by solving six equations (equation (3)) in five unknowns.

Construct six equations

$$KA_2 = x_2x_8 \tag{4.1}$$

$$KA_3 = 2x_3x_8+x_2x_7 \tag{4.2}$$

$$KA_o = 2x_3x_7+x_3^2 \tag{4.3}$$

$$KA_5 = x_2 \tag{4.4}$$

$$KA_4 = 2x_3 \tag{4.5}$$

$$K(A_1-A_3) = x_2x_3 \tag{4.6}$$

From equations (4.1) and (4.4)

$$x_8 = A_2/A_5 \ . \tag{4.7}$$

From equations (4.4) and (4.6)

$$x_3 = (A_1-A_3)/A_5 \ . \tag{4.8}$$

From equations (4.5) and (4.6)

$$x_2 = 2(A_1-A_3)/A_4 \ . \tag{4.9}$$

From equations (4.4) and (4.5)

$$K = 2(A_1-A_3)/A_4A_5 \ . \tag{4.10}$$

From equations (4.2), (4.8), (4.9) and (4.10)

$$x_7 = (A_3A_5 - A_2A_4^2)/A_4A_5^2 \quad . \tag{4.11}$$

But from equations (4.3), (4.8) and (4.10)

$$x_7 = \frac{A_o}{A_4} - \frac{(A_1 - A_3)}{2A_5} \quad . \tag{4.12}$$

From equations (4.11) and (4.12), these sets of equations are consistent if and only if

1. Either $x_7 > 0$, when

$$A_4(2A_2A_4 - (A_1 - A_3)A_5) = 2A_5(A_3A_4 - A_oA_5)$$

or 2. $x_7 = 0$, when

$$A_2A_4 = A_3A_5 \text{ and } 2A_oA_5 = A_4(A_1 - A_3) \text{ , see Fig.[4]}$$

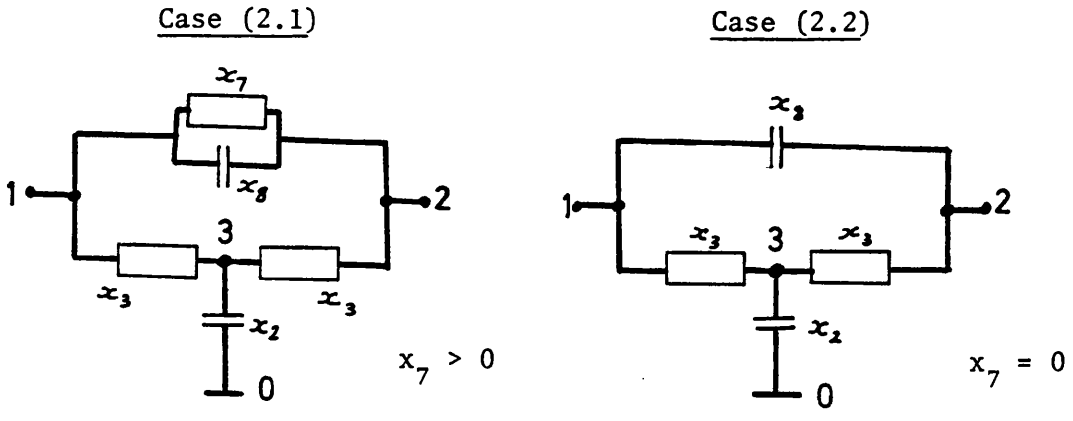


Figure [4] The two feasible topologies resulting from Case 2 to realize equations (1)

The two cases described above represent all the possible topologies which can be selected from the general topology shown in Fig.[1]. Both these cases may realize the set of polynomials given in equations (1) under certain conditions imposed by the given set of polynomials. Hence the numerical values of the coefficients will decide which, if any, of the

two networks shown in Fig.[2] yields the required set of polynomials.

To summarize, for a set of equations

$$\Delta_{11} = \Delta_{22} = K(A_o+A_1p+A_2p^2)$$

$$\Delta_{12} = K(A_o+A_3p+A_2p^2)$$

$$\Delta_{1122} = K(A_4+A_5p) \ ,$$

the required spread of powers can be obtained from a network composed of three nodes plus a reference node. Because of the constraints imposed by the required set of equations (symmetry, compactness at the pole at zero, and compactness at the pole at infinity), there are two alternative networks only which may yield the required solution. Each of these two networks has its own limitations. These limitations depend upon the coefficient values of the required set of polynomials.

Case 1 cannot realize equations (1) unless

1. either $x_8 > 0$, when

$$A_4(2A_2A_4-(A_1-A_3)A_5) = 2A_5(A_3A_4-A_oA_5)$$

or 2. $x_8 = 0$, when

$$A_oA_5 = A_3A_4$$

and $2A_2A_4 = A_5(A_1-A_3)$.

Case 2 cannot realize equation (1) unless

1. either $x_7 > 0$, when

$$A_4(2A_2A_4-(A_1-A_3)A_5) = 2A_5(A_3A_4-A_oA_5)$$

or 2. $x_7 = 0$, when

$$A_2A_4 = A_3A_5 \text{ and } 2A_oA_5 = A_4(A_1-A_3) .$$

Accordingly, none of the topologies shown in Fig.[2] can realize equations (1) if the coefficient values will not satisfy any of the aforementioned requirements. Such coefficient values are as those of equations (5)

$$\left.\begin{aligned} \Delta_{11} = \Delta_{22} &= K(1+3p+p^2) \\ \Delta_{12} &= K(1+2p+p^2) \\ \Delta_{1122} &= K(1+p) \end{aligned}\right\} \qquad (5)$$

i.e. $A_o = 1$, $A_1 = 3$, $A_2 = 1$, $A_3 = 2$, $A_4 = 1$ and $A_5 = 1$.

The coefficient values given in equations (5) do not satisfy any of the aforementioned requirements.

Case 1

$(x_8 > 0)$ $\quad A_4(2A_2A_4-(A_1-A_3)A_5) = 1(2x1x1-(3-2)x1) = 1$

$$2A_5(A_3A_4-A_oA_5) = 2x1(2x1-1x1) = 2$$

$(x_8 = 0)$ $\quad A_oA_5 = 1x1 = 1$, $A_3A_4 = 2x1 = 2$

$$2A_2A_4 = 2 \text{ , } A_5(A_1-A_3) = 1$$

Case 2

$(x_7 > 0)$ $\quad A_4(2A_2A_4-(A_1-A_3)A_5) = 1$

$$2A_5(A_3A_4-A_oA_5) = 2$$

$(x_7 = 0)$ $\quad A_2A_4 = 1x1 = 1$, $A_3A_5 = 2x1 = 2$

$$2A_oA_5 = 2 \text{ , } A_4(A_1-A_3) = 1$$

However, the set of polynomials in equations (5) can be realized by introducing a common factor $(p+1)$, which necessitates the introduction of an extra node, and a realization can be obtained as shown in Fig.[5].

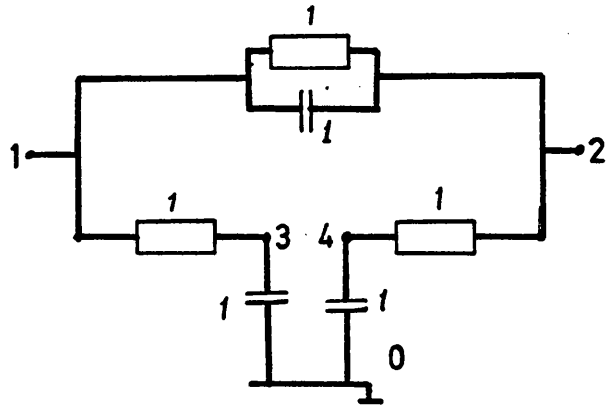


Figure [5] Network Realizing Equations (5)

Equations (5), after the addition of the new common factor will be the form of equations (6)

$$\left.\begin{aligned} \Delta_{11} = \Delta_{22} &= K(p+1)(1+3p+p^2) \\ \Delta_{12} &= K(p+1)(1+2p+p^2) \\ \Delta_{1122} &= K(p+1)(p+1) \end{aligned}\right\} \qquad (6)$$

Although the introduction of the common factor $(p+1)$ increases the number of the non-zero coefficients by four, not all of the extra four coefficients are independent. To multiply the four principal polynomials by the same factor necessitates the addition of an extra node. Accordingly, the number of possible elements increased from eight elements to sixteen elements, i.e. the number of elements doubled. Thus the number of alternative topologies to choose from within the frame of a total of five nodes is increased. Further, addition of an extra node, increases the number of 2-trees and 3-trees in general. As a result, the inconsistency in the set of equations used to evaluate the network element values is overcome.

Let us consider that the degree of connectivity is some measure of the

number of different topologies which are possible for a network with a given number of nodes. A network with a minimum number of nodes which can yield the required spread of powers to synthesize a given set of functions will possess a minimum number of different topologies. Thus such a network has a low degree of connectivity. Such a network may, or may not, be able to satisfy the constraints imposed by the required set of functions, if any. For some network functions, a higher degree of connectivity may be needed in order to achieve a solution as shown in the example given above. A higher degree of connectivity is obtained by adding extra nodes to the network with minimum number of nodes. The more extra nodes that are added, the higher the degree of connectivity that is obtained.

Unfortunately, the author did not have enough time to develop an analytical formula to express the degree of connectivity. Such a formula will be very useful in order to estimate the minimum number of nodes required for a feasible starting network. A network composed of the correct number of nodes is able to achieve a solution much faster as shown in Section 7.2.

APPENDIX B

TRANSFER PHASE-FREQUENCY CHARACTERISTIC OF NON-SERIES-PARALLEL NETWORK

It was pointed out in Chapter 1 that non-series-parallel networks have been considered extensively in the present work. The reason is that non-series-parallel networks cannot be realized using classical methods for network synthesis and hence provide good test examples for the computer design techniques developed in the present work. Further, the special characteristics possessed by this type of network have yet to be investigated. What is so special about non-series-parallel networks which do not have a series-parallel equivalent? So far, non-series-parallel networks with no series-parallel equivalent have only been considered as an academic problem. All the discussions[7,8,23,24,27] were concentrated on the implementation of the negative coefficient in the reduced numerator of the transfer admittance function. No one, to the author's knowledge, has investigated other aspects of the network performances yielded by this type of network. In particular, what performance characteristics can a non-series-parallel network yield, and is the same characteristic obtainable with a series-parallel network? One property of interest is the transfer phase-frequency characteristic, which is often examined by a circuit design engineer. An investigation into this characteristic proved to be very fruitful and led to a special result which has not appeared in literature before.

Zeros of the transfer admittance function may be located in the right-half-plane of the complex frequency p. Such transfer functions are known as non-minimum-phase functions. Transfer functions with no zeros in the right-half-plane are minimum-phase functions.

The set of admittance functions given by Fialkow[8], which can only be realized by non-series-parallel network with no series-parallel equivalent are

$$\left.\begin{aligned} Y_{11} = Y_{22} &= \frac{1197p + 56613.14p^2 + 28368.584p + 191.184}{800000(p^2+0.51p+0.0048)} \\ -Y_{12} &= \frac{3p^3 - 1.14p^2 + 197.176p + 77.616}{800000(p^2+0.51p+0.0048)} \end{aligned}\right\} \quad (1)$$

(the common factor (p+1) has been removed from the equations).

The partial-fraction form of equations (1) was published by Cutteridge and Krzeczkowski[27]. The transfer admittance in equations (1) can be written in pole-zero form as

$$-Y_{12} = \frac{3(p+0.3918)(p-0.3959+j8.1166)(p-0.3959-j8.1166)}{800000(p+0.5004)(p+0.0959)} \quad (2)$$

The corresponding pole-zero plot is shown in Fig.[1]

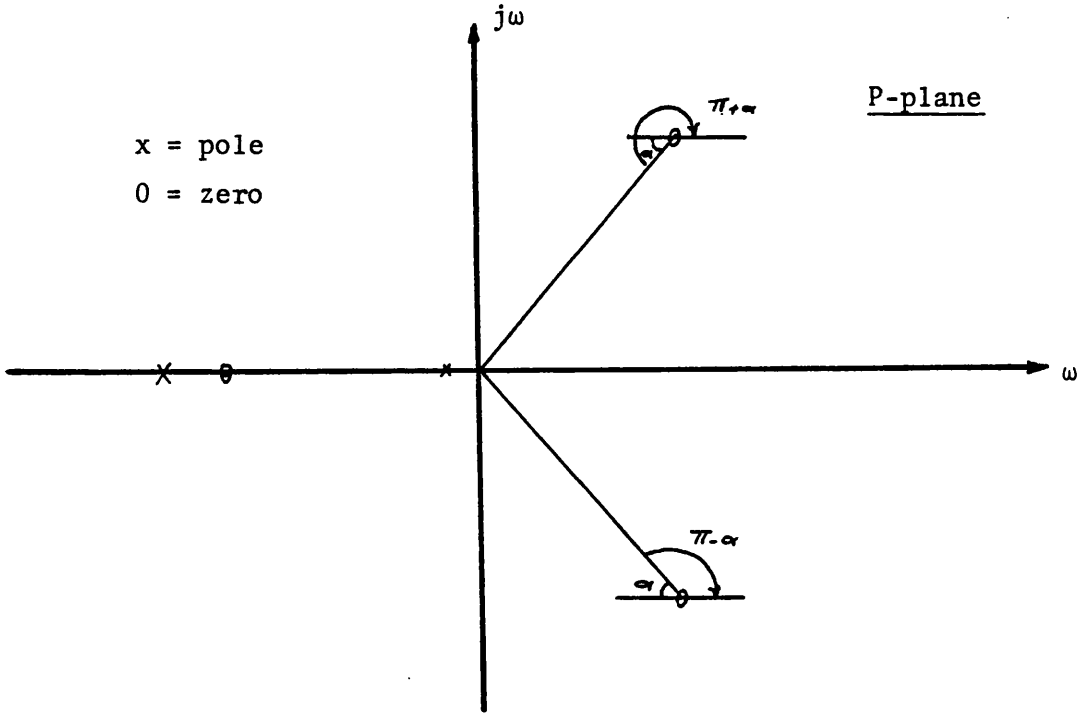


Figure [1] Pole-zero plot of equation (2)

From equation (2) and Fig.[1], it can be observed that the transfer function $-Y_{12}$ has a conjugate complex pair of zeros in the right-half-plane, i.e. it is a non-minimum-phase function. Further, from equations (1) and (2) and Fig.[1] ,

$$Y_{11(p)}\Big|_{p\longrightarrow 0} = \frac{191.184}{3840} \qquad \text{phase } Y_{11(jo)} = 0°$$

$$-Y_{12(p)}\Big|_{p\longrightarrow 0} = \frac{77.616}{3840} \qquad \text{phase } -Y_{12(jo)} = 0°$$

$$Y_{11(p)}\Big|_{p\longrightarrow \infty} = \frac{1197p}{800000} \qquad \text{phase } Y_{11(+j\infty)} = +\frac{\pi}{2}$$

$$-Y_{12(p)}\Big|_{p\longrightarrow 0} = \frac{3}{800000}\,p \qquad \text{phase } -Y_{12(+j\infty)} = +\frac{\pi}{2}$$

For the transfer function $-Y_{12}$, the net increase in phase can be analysed for variation of ω from 0 to ∞ as follows:

Net increase in phase for one zero in the left-half-plane (LHP) is $+\frac{\pi}{2}$.

Net increase in phase for one pole in the LHP is $-\frac{\pi}{2}$.

Phase contribution at $\omega = 0$ for a conjugate complex pair of zeros in the right-half-plane (RHP) $= +\pi + \alpha + \pi - \alpha = 2\pi$.

Phase contribution at $\omega = +\infty$ for a conjugate complex pair of zeros in the RHP $= +\frac{\pi}{2} + \frac{\pi}{2} = +\pi$.

Net increase in phase for a conjugate complex pair of zeros in the RHP $= +\pi - 2\pi = -\pi$.

To summarize, for one zero in the LHP and,

for two zeros in the RHP and,

for two poles in the LHP,

the net increase in phase for the variation of ω from 0 to $+\infty$

$$= +\frac{\pi}{2} - \pi + 2x - \frac{\pi}{2}$$

$$= +\frac{\pi}{2} - \pi - \pi$$

$$= + \frac{\pi}{2} - 2\pi = - \frac{3\pi}{2}$$

$$= - - 270° .$$

Hence, if the phase-frequency characteristic of non-series-parallel network is plotted it would look as shown in Fig.[2].

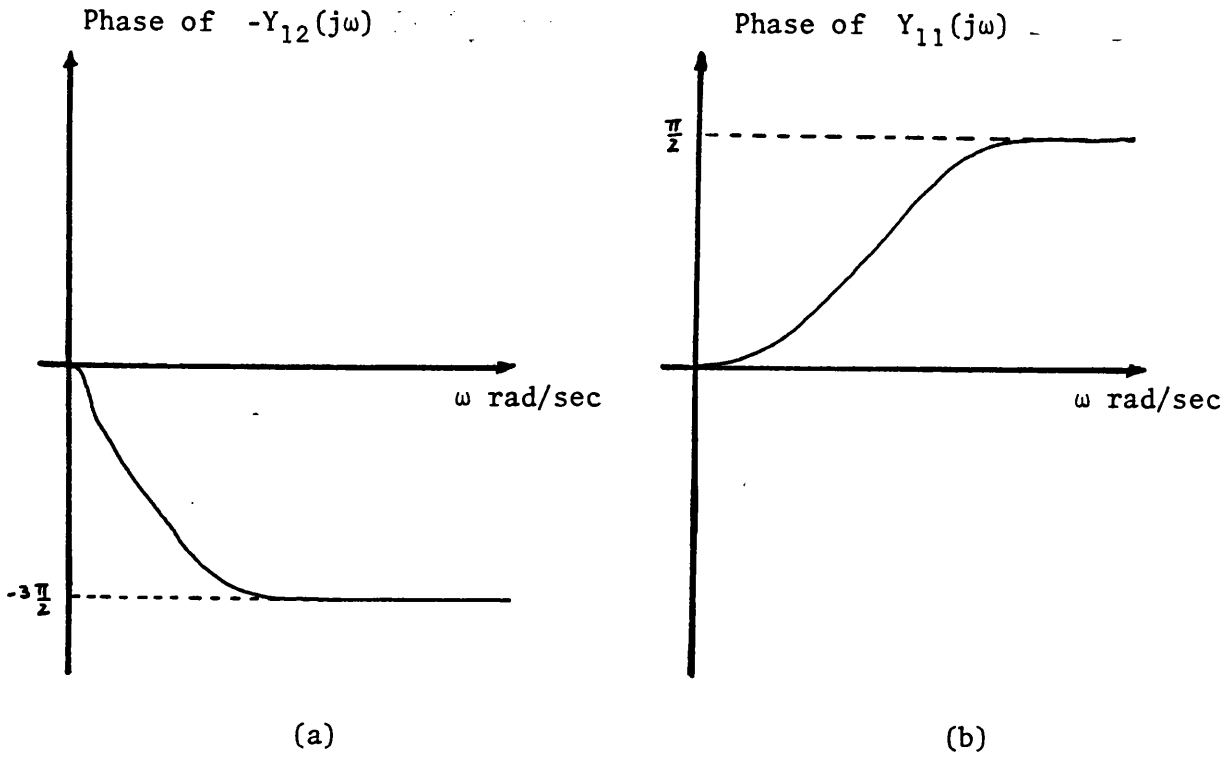


Figure [2]. Phase-Frequency Characteristics of Non-Series-Parallel Network

The result shown in Fig.[2] is a further justification for why these type of networks cannot have a series-parallel equivalent. The form of the phase-frequency characteristic shown in Fig.[2] differs from any phase-frequency characteristic previously known for series-parallel, non-minimum phase networks. A similar form of parts of the characteristic shown in Figure [2] may be obtained by a series-parallel non-minimum phase network, but an identical form cannot be obtained. Thus in order that the phase characteristic to be continuous for all networks in general, a more general form to represent the increase in phase should be considered.

This generalized form may be defined as $(+\frac{\pi}{2} \pm n\pi)$, where n is a non-negative integer ≥ 0 . The reason for this salient characteristic is that the transfer function, which is a cubic over quadratic rational polynomial, has a negative coefficient in its numerator. Hence it has a conjugate pair of complex zeros in the right-half-plane and a zero on the negative axis. This characteristic necessitates that the transfer function of non-series-parallel network with a negative coefficient in the numerator of $-Y_{12}$ be asymptotic to $(-3\frac{\pi}{2})$ instead of $(+\frac{\pi}{2})$.

This especial characteristic could be useful in circuit design. Recently[24], it has been suggested that non-series-parallel networks could be used as 'building blocks' in the construction of more sophisticated topologies with no series-parallel equivalent. One such application could be to use non-series-parallel networks as the main building block of Phase Equalizers with a series-parallel network in the feedback loop.

To summarize, the non-series-parallel network proved to have a special transfer phase-frequency characteristic which was not considered before. This characteristic merits further investigation from the application point of view in circuit design.

REFERENCES

1. VAN VALKENBURG, M.E., Introduction to Modern Network Synthesis. New York: Wiley, 1960.

2. WEINBERG, L., Network Analysis and Synthesis. New York: McGraw-Hill, 1962.

3. DARLINGTON, S., "A Survey of Network Realization Techniques", IRE Trans. Vol. CT-2, December 1955.

4. ADAMS, K.M., "On the synthesis of three-terminal networks composed of two kinds of elements", Philips Res. Rep., 1958.

5. OLIVARES, J.E., "Synthesis of N-port RC networks", Ph.D. dissertation, University of California, 1959.

6. FIALKOW, A., and GERST, I., "The transfer functions of general two-terminal-pair RC networks", Quart. Appli. Math., Vol.10, pp.111-127, 1952.

7. FIALKOW, A., "An Equivalence Theorem", IEEE Trans. Circuit Theory, Vol. CT-19, pp.9-14, 1972.

8. FIALKOW, A., "A limitation of the series-parallel structure", IEEE Trans. Circuit Theory, Vol. CT-15, pp.124-132, 1968.

9. KUO, F.F., and MAGNUSON, W.G., Computer Oriented Circuit Design. New Jersey: Prentice Hall, 1969.

10. BOWN, G.C.S., "Design and optimization of circuits by computers". Proc. IEE, Vol. 118, pp.649-662, 1971.

11. FAZARINC, Z., "Designer-oriented CAD". IEEE Trans. Circuit Theory, Vol. CT-20, pp.673-682, 1973.

12. CALAHAN, D.A., Computer-aided network design. New York: McGraw-Hill, 1972 (1968).

13. CALAHAN, D.A., "Computer solution of the network realization problem", Proc. 2nd Allerton Conf. on Circuit and System Theory, pp.175-193, 1964.

14. CALAHAN, D.A., "Computer design of linear frequency selective networks", Proc. IEEE, Vol.53, pp.1701-1706, November 1965.

15. TEMES, G.C., and CALAHAN, D.A., "Computer-aided network optimization - The state-of-the-Art". Proc. IEEE, Vol.55, pp.1832-1863, 1967.

16. ROHRER, R.A.,"Fully automated network design by digital computer: preliminary considerations", Proc. IEEE, Vol.55, pp.1929-1939, 1967.

17. DIRECTOR, S.W., and ROHRER, R.A., "The generalized adjoint network and network sensitivities", IEEE Trans. on Circuit Theory, Vol.CT-16, pp.318-323, 1969.

18. DIRECTOR, S.W., and ROHRER, R.A., "Automated network design - the frequency domain case", IEEE Trans. on Circuit Theory. Vol.CT-16, pp.330-337, August 1969.

19. DIRECTOR, S.W., "Survey of circuit-oriented optimization techniques". IEEE Trans. on Circuit Theory, Vol.CT-18, pp.3-10, 1971.

20. DIRECTOR, S.W., and ROHRER, R.A., "On the design of resistance n-part networks by digital computer". IEEE Trans. Circuit Theory, Vol.CT-16, pp.337-346, 1969.

21. WRIGHT, D.J. "Studies in optimization as an aid to circuit synthesis and design". Ph.D. thesis, University of Leicester, 1974.

22. DI MAMBRO, P.H., "An investigation into the uses of coefficient matching in network synthesis". M.Sc. thesis, University of Leicester, 1970.

23. DI MAMBRO,P.H., "A study of the evolutionary approach to network synthesis using coefficient matching". Ph.D. thesis, University of Leicester, 1974.

24. KRZECZKOWSKI, A.J., "Theory and design of lumped linear three-terminal RC network". Ph.D. thesis, University of Leicester, 1976.

25. CUTTERIDGE, O.P.D., "Automated synthesis of non-series parallel 3-terminal RC networks". Proc. Fifth Colloquium on Microwave Communication (Budapest, Hungary), June 1974.

26. CUTTERIDGE, O.P.D., "Electrical network and general system synthesis using optimization techniques". (in Optimization and Design, eds. M. Avriel, M.J.Rijckaert and D.J. Wilde), pp.76-80, New Jersey: Prentice Hall, 1973.

27. CUTTERIDGE, O.P.D.,and KRZECZKOWSKI, A.J., "Further non-series parallel 3-terminal RC networks having no series-parallel equivalent". IEEE Trans.Circuit Systems, Vol.CAS-22, pp.381-385, 1975.

28. CUTTERIDGE, O.P.D., and DI MAMBRO,P.H., "Simultaneous generation of the coefficients of network polynomials and their partial derivatives from the nodal admittance matrix", Electron. Lett., pp.308-310, 1970.

29. CUTTERIDGE, O.P.D. and DI MAMBRO, P.H., "Simultaneous generation of the partial derivatives of network polynomial coefficients - Further details and results". Electron. Lett., Vol.7, pp.3-4, 1971.

30. CUTTERIDGE, O.P.D., and KRZECZKOWSKI, A.J., "Improved methods of synthesizing linear networks by coefficient matching", IEEE Trans. Circuit Systems, Vol.CAS-22, pp.486-489, 1975.

31. CUTTERIDGE, O.P.D., "Numerical experience with some two-part programmes for the solution of non-linear simultaneous equations". University of Leicester, Department of Engineering, Report 72-20, September 1972.

32. SHU-PARK CHAN, Introducing topological analysis of electrical networks. New York: Holt, Rinehart and Winston Inc., 1969.

33. VLACH, J., Computerized approximation and synthesis of linear networks. New York : John Wiley, 1969.

34. LUCAL, H.M., "Synthesis of three-terminal RC networks", IRE Trans. Circuit Theory, Vol.CT-2, pp.308-316, 1955.

35. FIALKOW, A., "Series-parallel grounded two-ports". Quart.Appli.Math., Vol.24, pp.195-213, 1966.

36. CUTTERIDGE, O.P.D., "Parallel decomposition of 3-terminal RC networks". Proc.IEE, Vol.112, pp.2220-2226, 1965.

37. HANSEN, P. and WANET,G., "Synthesis of LC 3-terminal networks by mixed-integer programming", IEEE Tran. Circuit Systems, Vol.CAS-21, pp.199-203, 1974.

38. KENDALL L. SU, Time-domain synthesis of linear networks. New Jersey: Prentice-Hall, 1971.

39. CAUER, W., Synthesis of linear communication networks. New York: McGraw-Hill, 1958.

40. FLETCHER, R., ed. Optimization. London : Academic Press, 1969.

41. MURRAY, W., ed., Numerical methods for unconstrained optimization. London : Academic Press, 1972.

42. SARGENT, R.W.H., and SEBASTIAN, D.J., "Numerical experience with algorithms for unconstrained optimization", (in Numerical methods for non-linear optimization, ed. F.A. Lootsma), pp.45-69, London: Academic Press, 1969.

43. BECKER, R.W., "Random search optimization by seeking modes", Ph.D. thesis, Missouri University, U.S.A., 1971.

44. WOZNY, M.J., "On random search strategies for parameter optimization". Proc. 9th Allerton Conf. on Circuit and System Theory, pp.390-395, 1971.

45. KJELLSTRÖM, G., "Network optimization by random variation of component values". Ericsson Techniques, pp.135-151, 1969.

46. OPAČIĆ, J., "A heuristic method for finding most extrema of a non-linear functional", IEEE Trans. System, Man and Cybernetics, Vol.SMC-3, pp.102-107, 1973.

47. HILL, J.D., "A search technique for multimodal surfaces", IEEE Trans. Systems Science and Cybernetics, Vol.SSC-5, pp.2-8, 1969.

48. McMURTRY G.J., "Adaptive optimization procedures", (in Adaptive, leaning and pattern recognition systems, ed. J.M. Mandel and K.S. Fu), Academic Press 1970.

49. BRANIN, F.H., "Widely convergent methods for finding the multiple solutions of simultaneous non-linear equations", IBM Jour. Research and Development, Vol.16, pp.504-522, 1972.

50. BROWN, K.M., and GEARHART, W.B., "Deflation technique for the calculation of further solutions of a non-linear system", Num.Math.,Vol.16, pp.334-342, 1971.

51. GOLDSTEIN, A.A., and PRICE, J.F., "On descent from local minima". Math. Comp., Vol.25, pp.569-575, 1971.

52. NEILL, T.B.M., and BOND, D.J., "Inversion of the nodal admittance matrix by the use of state equations". Int. Journal Circuit Theory and Appl., Vol.1, pp.249-261, 1973.

53. TEMES, G.C., and ZAI, D.Y.F., "Least pth approximation". IEEE Trans. Circuit Theory, Vol.CT-16, pp.235-237, 1969.

54. CUTTERIDGE, O.P.D., and KRZECZKOWSKI, A.J., "An automated network design program". University of Leicester, Department of Engineering, Report 76-4, May 1976.

55. FLETCHER, R., and REEVES, C.M., "Function minimization by conjugate gradients", Computer Journal, Vol.7, pp.149-154, 1964.

56. WILKINSON, J.H., and REINSCH, C., Handbook of Automatic Computation, Vol.II, Berlin: Springer-Verlag, 1971.

57. NAG Library Manual (Mark 4), Numerical Algorithms Group, Oxford University, Computing Laboratory.

58. CUTTERIDGE, O.P.D., and DI MAMBRO, P.H., "Some examples demonstrating feasibility of evolutionary approach to linear network synthesis". Electron. Lett., Vol.10, pp.30-31, 1974.

59. CUTTERIDGE, O.P.D., "Computer synthesis of lumped linear networks of arbitrary structure". (in Network and Signal theory, eds. J.K.Skwirzynski and J.O.Scanlan), pp.105-111. London : Peter Peregrinus, 1973.

60. VILLALAZ, P.A., and SPENCE, R., "Scheme for the elimination of redundant model complexity". Electron. Lett. pp.38-41, 1972.

61. CUTTERIDGE, O.P.D., "Expressions for optimum values to grow virtual elements", unpublished manuscript, 1974.

62. FIALKOW, A.,Private communication with Dr. O.P.D. Cutteridge, 19 November 1975.

63. HACHTEL, G.D., and ROHRER, R.A., "Techniques for the optimal design and synthesis of switching circuits". Proc. IEEE, Vol.55, pp.1864-1877, 1967.

64. FIDLER, J.K., and SEWELL, J.I., "Symbolic analysis for computer aided circuit design - The interpolative approach". IEEE Trans. Circuit Theory, Vol.CT-20, pp.738-741, 1973.

65. MASSARA, R.E., and FIDLER, J.K., "Computer optimization of frequency selective networks", Proc. 1974 European Conference on Circuit Theory and Design, IEE Conference publication No.116, pp.133-140.

SUMMARY

During the past decade, many techniques for computer-aided circuit design have been suggested and investigated, but none have been developed to the stage where the designer is redundant. The research described herein concerns the application of one technique, the method of coefficient matching, to the synthesis of lumped, linear, passive, 3-terminal networks with no mutual inductances. The author developed a program which, with further development, could perform the entire design process with no designer interaction. Further, the author considers problems where the classical synthesis methods are unsuitable, e.g. the synthesis of non-series-parallel networks with no series-parallel equivalent.

The coefficient matching procedure is based on selecting a starting network which yields the correct polynomial structure and achieving a solution by component value adjustment and network evolution. The closer the starting network to a feasible topology, the more rapid the convergence to a solution. It is shown that the suitability of a starting network can be analysed on the basis of the information obtained from the different but equivalent forms of the admittance functions. The significance of common factors is discussed and the influence of various types of common factors on the network realization is investigated.

For cases when the initial starting network is remote from any feasible solution, sophisticated techniques allowing substantial topological modification during network evolution are required. These techniques were developed by the author on the basis of element and node addition and elimination.

A Fortran IV program has been developed by the author welding together all these aforementioned techniques for topological modification. The

program makes large topological modification automatically during the design process. The effectiveness and efficiency of these techniques and the program are illustrated by a variety of synthesis examples.