AUTOMATED SYNTHESIS OF LUMPED LINEAR THREE-TERMINAL NETWORKS

by

W.H. Savage, B.Sc.(Hons)
Leicester University 1975

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

JUNE 1979

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

SUMMARY

The classical techniques of network synthesis are restricted to designs in idealized elements with series-parallel configurations. This research is an investigation into the possibility of unrestricted synthesis employing alternative techniques which involve optimization by computer. In this method the values of the elements are modified such that an error function is reduced. If the current network is unable to satisfy the required network response then the components have to be modified. A method of coefficient matching was investigated with lumped, linear, passive three-terminal networks having a maximum of ten nodes.

The research utilized a design package developed by Drs. O.P.D. Cutteridge and A.J. Krzeczkowski. This formulated the problem for solution by an RC network with a fixed number of nodes. An effective analysis routine calculated the values of the coefficients and their first derivatives, for optimization by the conjugate gradient and Gauss-Newton algorithms. The rudiments of a method for the addition and removal of a single element had been developed.

Research was undertaken into three areas. Firstly, the efficiency and dependability of the optimization was improved. This involved research into the individual error functions, variation of common factors and the efficient utilization of the optimization algorithms. Secondly, modifications to the network topology were considered. The criteria to determine the need for a modification were improved and checks to ensure the continued efficiency implemented. An improved method of element addition (capable of multiple additions) was devised. Thirdly, the addition of groups of elements was investigated (i.e. node addition) and a successful method developed.

With these modifications implemented, the package was able to achieve more complex realizations than had previously been obtained. For example some seven node RC realizations with fifteen elements were automatically evolved from initial structures having five nodes and eight elements, a process which sometimes required a total of twenty-five topological modifications. Several theoretically interesting networks which were evolved automatically by the package are included.

ACKNOWLEDGEMENTS

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 thank the computer assistants of the Department of Engineering, particularly Dr. A.J. Krzeczkowski.

Finally, I wish to thank the Science Research Council for the award of a research studentship over the period 1975-78 and Racal-Redac Ltd. for financial support over the period 1978-79.

STATEMENT

The accompanying thesis submitted for the degree of Doctor of Philosophy entitled 'Automated Synthesis of Lumped Linear Three-Terminal Networks' is based on work conducted in the University of Leicester mainly during the period between October 1975 and July 1978.

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: W.H. Savage Date: 13 August 1979.

CONTENTS

LIST OF TABLE NUMBERS

LIST OF FIGURE NUMBERS

Page

CHAPTER 1

INTRODUCTION

The development of integrated circuits and the requirement for increasingly more demanding specifications have amplified the deficiencies of the traditional methods of network synthesis. Termed the classical techniques[1,2], these methods are all based upon series-parallel decompositions (series decomposition of impedance functions and parallel decomposition of admittance functions). This produces a network composed of small subnetworks which can be easily synthesized. In 1968, Fialkow[3] proved that not all feasible networks can be realized within the confines of a series-parallel topology. These techniques are also deficient in that they produce realizations with excess numbers of elements, and exclude consideration of element parasitics, variable Q factors, possible constraints on element values, etc. Alternative methods are required to ensure designs do satisfy the requirements when actual elements are considered.

The computer, which can perform many calculations rapidly and at low cost, enables methods of directed trial and error to be employed. Such methods are, therefore, no longer dependent on the topology of the network. Mass production techniques have led to a situation where larger sums can be expended on the initial design. Consequently, computing costs are unlikely to prove prohibitive and are anyway continuously being reduced (currently at a rate of 15% per annum[4]).

Any feasible techniques which arise from this research are applicable to many fields other than electrical network synthesis. Calahan[5] intimated that many other systems are analogous to electrical networks, having easily identifiable 'through' and 'across' variables analogous to current and voltage. Also the automated design concept has links with machine intelligence (more specifically pattern recognition) and, further, the use of optimization techniques is spreading to many fields.

In the area of network synthesis the computer was initially applied to the design of filters[6]. Theoretical design techniques had failed to satisfy the required performance criteria when physical elements had been used to construct the network. In 1956, Aaron[7] proposed using a least-squares approach for improving upon the design at each stage of the classical synthesis methods. Desoer and Mitra[8] (1961) further proposed an iterative optimization method which varied the values of the physical non-ideal elements of ladder filters, to reduce the variance between the desired and the actual transfer functions. Calahan[9] (1965) extended this idea with a method applicable to any linear network, but the intention remained to modify classical designs to nullify the effects of parasitics. However, the ultimate aim must be to complete the design entirely by computer. Director and Rohrer[10] (1969) instigated the next forward step by developing a method for the addition of elements, a possibility recognized by Calahan[9]. There are, however, no commercial packages currently available which can achieve this goal. Several of the large scale analysis packages have facilities for optimization and consider other aspects such as non-linearity and temperature. However, none appear to be capable of modifying the network topology[11].

At Leicester, several years of research into computer aided network design have been supervised by Dr. O.P.D. Cutteridge. A method of coefficient matching has been successfully used, although pole-zero and direct matchingwere considered by Wright[12]. Wright also examined interactive design as well as the predominantly used automated design.

Coefficient matching has long been recognized to have good properties of convergence[9] and has also proved of greater efficacy for topological modifications. Early results[12-15] disclosed the possibility of reducing the size of networks produced by classical or other techniques. Similarly, success has been achieved with additions of both elements [12-14,16-18] and of nodes[14]. Practical problems requiring restrictions on the values of elements etc., have also been solved[12].

The author's research described in this thesis is devoted to the synthesis of 3-terminal lumped, linear, time invariant networks by a coefficient matching method, and utilized the batch mode exclusively. Automated design in the batch mode requires less sophisticated equipment and users less skilled in the discipline, but can be hazardous in that it is possible for entire job-times to elapse without useful developments. However, it can also prove useful in overcoming in-built prejudices of an interactive user, i.e. preconceived concepts of the topology of a solution network. This factor precluded the discovery of a six node computer realization to the set of functions given by Lucal[19] (equations 1.1) despite repeated attempts, particularly by Hegazi[14] and also the author in early work.

$$\left.\begin{aligned} y_{11} &= \frac{36p^4 + 2058p^3 + 6552p^2 + 4638p + 36}{36p^3 + 216p^2 + 396p + 216} \\ -y_{12} &= \frac{36p^4 + 36p^3 + 72p^2 + 36p + 36}{36p^3 + 216p^2 + 396p + 216} \\ y_{22} &= \frac{36p^4 + 533p^3 + 1572p^2 + 1183p + 36}{36p^3 + 216p^2 + 396p + 216} \end{aligned}\right\} \quad 1.1$$

Coefficient matching can only be applied after the given set of design specifications have been transformed into the required functions (termed, the approximation stage). In this thesis the coefficients are those of the polynomials in p , the complex frequency variable, of the short-circuit admittance parameters. Typically[20], the approximation stage is performed by computer, using methods such as those of Remez. A starting network which satisfies the order of these polynomials, with or without common factors, is required. The method of analysis used was based on the nodal admittance matrix, which is small and easily formulated when compared to the alternatives, and had been developed at Leicester[13,21-24]. It is capable of rapidly calculating the network coefficients and their derivatives with respect to the elements. Coincident with the research in this thesis the possibility of an alternative method of analysis employing fast Fourier transforms[25,26] was investigated by Banerjee[27]. With the order of magnitude of problems considered here, there was no variation in accuracy or speed experienced.

Once the coefficient values have been calculated, an individual error function (f_i) is determined for each coefficient in order to provide a measure of the variation between the actual and desired values. These functions are formulated such that $f_i = 0$ when the two values are identical. As individual functions will vary rapidly with fluctuations in the element values, and so cannot all be reduced simultaneously, an overall summation of the errors (F) is used to evaluate the progress. (Generally this performance function is a squared summation.) Optimization routines are used to modify the element (and common factor) values such that error F is reduced. The variables are optimized in the logarithmic domain to ensure that the element values remain positive. Although active equivalents to negative elements can be obtained, they only approach the idealized behaviour over a limited range of voltages, currents and frequencies[28]. Furthermore, a passive rather than an active design may be preferable. If the current network is unable to satisfy the design requirements, then this has to be established and the network topology modified.

The only elements considered were lumped, linear resistors, capacitors and inductors (RLC networks) with no parasitic effects. Consequently this research can only be regarded as one stage in the complete design of an electrical network. However, the results produced may be of interest to theoreticians and prove beneficial in modelling situations.

The approximation stage and final tuning to the requirements (by, say, design centering[29]) are not considered. The major requirement of any design is that it should satisfy the desired performance criteria. Determining which of a set of possible solutions is the optimum design will depend on requirements such as cost, or the lowest sensitivity of the characteristics to element variations. These parameters are not dealt with and the design is directed towards producing a solution with the minimum number of nodes. This approach will tend to minimize the time taken to locate a realization (because the time for each analysis increases as the number of nodes increases) and also to minimize the number of elements. This in turn increases the speed of the optimization and possibly reduces the cost of production, if not the sensitivity. An adequate design was only considered to have been achieved when the error function F had been reduced to the limits of the computer accuracy. This is termed a computer realization, and not a solution, as it is uncertain if this will be an exact solution or merely a very good quazi solution.

With idealized RLC networks the solution of practical problems cannot be attempted. The motivation behind the research was to solve academic test problems with known solutions and to develop processes which could then be applied to other problems which did not have known solutions. If required, it would then be possible to consider more practically orientated problems with model representations of actual elements, etc. Two test problems were considered extensively. Two element kind solutions to Lucal's functions (eq.1.1) had been obtained by using classical series-parallel decomposition techniques[19,30,31]. However, all the solutions had contained at least seven nodes plus the reference node, which is two more than the minimum as indicated by the order of the polynomials. This problem is therefore of particular interest in the derivation of techniques for the addition of nodes. The second problem is due to Fialkow[3] (eq. 1.2),

$$\left.\begin{aligned} y_{11} = y_{22} &= \frac{1197p^3 + 56613.14p^2 + 28368.584p + 191.184}{800000p^2 + 408000p + 3840} \\ -y_{12} &= \frac{3p^3 - 1.14p^2 + 197.176p + 77.616}{800000p^2 + 408000p + 3840} \end{aligned}\right\} \quad 1.2$$

and is of particular interest because its realization must have a non-series-parallel structure and so cannot be solved by classical synthesis techniques. Several other test examples were derived from these two sets of functions.

An introductory grounding in the basic principles of the coefficient matching technique is given in Chapter 2. The formulation of the nodal admittance matrix and the significance of the short-circuit admittance parameters are also explained. The powers of the complex frequency variable, p , present in the functions make it preferable to calculate the polynomial coefficients by assigning real values to p . The analysis method employed, as developed by Cutteridge and di Mambro[13,21-24], is outlined briefly.

The individual error functions are non-linear functions of the network elements, hence a set of non-linear simultaneous equations are formed. The derivatives of the coefficients and thus the individual error functions are easily attainable (an advantage over other forms of modelling such as pole-zero or direct matching). Consequently, optimization algorithms which use this information are to be preferred to those which use only function evaluations. The most powerful algorithm for the solution of non-linear simultaneous equations is the Newton-Raphson (NR) method, known as the Gauss-Newton (GN) method when transformed to accommodate overdetermined equations. Here the individual error functions (f_i) are considered and not the overall error F as with the gradient descent methods. However, when far from a solution the GN can fail to converge. Consequently, two-part programs switching from a stable, but less powerful, gradient descent method to the GN method are often used. Di Mambro[13] found this type of two-part package to be superior to Levenberg's technique, which is a quasi-Newton method. Details of the optimization algorithms investigated conclude Chapter 2.

The research contained in this thesis is a continuation of the work of Krzeczkowski[16]. With the analysis as developed by Cutteridge and di Mambro[13,21-24], the element values were modified by the conjugate gradients method of Fletcher and Reeves[32] and the GN algorithm. Criteria had been developed to switch from one optimization routine to the other and a choice between two individual error function formulations was possible. Furthermore, algorithms had been included to instigate element removals and element additions based on closed form expressions for the optimum values of the virtual elements[33]. (There exists the possibility of an element connected between any two nodes. Any element which is not present is, in effect, set to zero and is referred to as a virtual element.) Although these facilities for topological modifications existed, there was no routine included to establish whether the synthesized network could be correctly matched to the required network. A description of such a routine, included by the author, and other safeguards such as checks on symmetry is given in Chapter 3. Also included is discussion of switching between the two sections in the optimization routines used, error function formulation and performance, variation of common factors, and improvements and modifications to the linear searches incorporated in the optimization routines. The original program, written in Algol, was restricted to three-terminal networks containing only resistors and capacitors (RC networks), with the possibility of only one common factor. A Fortran version was developed by the author which considered both RC and RLC networks with any number of common factors.

It has already been stated that there had been examples which had reduced solutions produced by classical techniques to solutions or quasi-solutions with fewer nodes. This suggests that a suitable design strategy would be to choose a start network with a number of superfluous nodes so that the optimization could remove nodes and elements, thus leaving a good approximation to the required network. Often, however, this proves impractical. The time taken for each analysis increases with the number of nodes. Further, the power of any optimization routine is reduced as the number of variables is increased. Consequently, progress in reducing the error with large, arbitrary, start networks can be negligible. An alternative strategy is to start with a minimal topology (i.e. comprising of the minimum number of elements and nodes possible to effectively synthesize the requirements) and add suitable elements to this initial guess. Also, as additions would generally improve any network which had been produced by the reduction of a larger network, element additions are of great significance.

The criteria which activate topological modifications are discussed in Chapter 3. Chapter 4 illustrates several possible methods of element addition, and a method based on the corrections given by the GN algorithm proved successful. Details of strategies devised to combat examples in which problems did arise with this method are also discussed.

Chapter 5 details the findings of the author with several methods for introducing new nodes into the network. A method was developed which solved many test examples.

A cross-section of results, both successful and unsuccessful, are illustrated in Chapter 6. Of particular interest are RC computer realizations to Lucal's functions with only six nodes plus the reference node, and RLC realizations to Fialkow's functions with a series-parallel structure. The general efficacy of the program developed is indicated by results from different start topologies to the same problem, and the same start topology with slightly differing test problems. The results are also compared to those achieved previously.

A final gathering together of the achievements of the research is contained in Chapter 7. The author also outlines several possible subjects for research, together with possible extensions to the package.

Three machines were used during the course of the research. Most of the results contained in Chapter 6 were obtained on a CDC Cyber 72. The original work was run on an ICL 1906A machine. When this shut down a Fortran version was transferred to an IBM 360 machine. The further transfer to the Cyber was performed because of the increased accuracy available.

CHAPTER 2

NETWORK THEORY, THE COEFFICIENT MATCHING TECHNIQUE AND THE OPTIMIZATION ALGORITHMS

2.1. Introduction

There are several possible methods of network representation which facilitate the use of computer aided design. These methods are interchangeable in that it is possible to transform the desired response from, say, time-dependent equations into the frequency domain[34] and from a desired frequency response into an approximate representation using polynomials in p[35], the complex frequency variable. These transformations are not always easily accomplished (particularly, say, obtaining the polynomial coefficients of narrow band networks) but are performed to expedite the design process. One such form of representation, coefficient matching, has been found to be particularly suitable and consequently has been employed in this research. The efficacy of this method was originally indicated by Calahan[9] and the major considerations were examined at Leicester before concentrating on this approach. The results obtained prior to, and during, this research would appear to validate this choice.

Two alternative methods are the direct method (which matches a desired response, say the frequency, at a number of positions) and the pole-zero method. In all three methods the variables are modified such that the synthesized response equates to the desired response.

After a cursory inspection it may seem that direct matching would be desirable in that it is unnecessary to approximate to a set of transfer functions, and also tolerances can be easily handled. With other methods, if a complete design is required, the tolerance requirements have to be considered at the approximation stage and, unless an exact synthesis of the functions is obtained, it remains uncertain whether the network will satisfy the requirements. In particular, the polynomial coefficients can be ill-conditioned, with small variations of the values resulting in significant movements of the poles and zeros. However, just as it may prove difficult to choose particular values for the calculation of an approximation to a response, difficulty may be experienced in determining the positions for direct matching. Further, the magnitude of the network required to effectively synthesize the response may not be apparent from frequency characteristics. With coefficient matching the minimum is evident although, even then, extra common factors may be required. Also, direct matching is further removed from the classical methods of synthesis which may provide useful information concerning the feasibility of networks, etc.

Several other factors favour the use of coefficient matching. Although difficulty may be experienced in determining the values of the required coefficients, once this approximation stage has been accomplished the design will proceed in a similar and straightforward manner. Further, it will be clearly apparent when a realization has been achieved. The method has also been found to have favourable properties of convergence[9] as, unlike other methods, the coefficients and derivatives are multilinear functions of the network elements. Consequently, the derivatives can be easily and accurately formulated.

The values of the poles and zeros can be determined from the values of the coefficients, therefore pole-zero matching will tend to suffer from similar deficiencies to coefficient matching. The major difficulties with the pole-zero technique arise in establishing which synthesized pole (or zero) is to be matched to which particular desired pole (or zero).

Calahan[9] was the first to employ a coefficient matching technique, attempting to match the coefficients of a single desired network function T(s) . The NR method was used to determine iteratively the required element values, the progress monitored by the reduction of error functions to a value of zero at a realization. A sufficient number of variables were fixed in value to ensure that a defined system was obtained.

The method was improved by Cutteridge[36,37] who recognised that simultaneous matching of a multiplicity of network functions could be achieved with multi-terminal structures, thus fully specifying the network. A number of elements were varied, sufficient to ensure that the synthesized network could be matched accurately to the desired network, whilst the other elements were set to zero, thus producing a defined or overdefined system.

Further explanation of the coefficient matching technique employed in this thesis appears in section 2.4. This section also outlines the significance of equivalent networks containing normalizing variables and common factors.

Possible methods of analysis and various optimization routines had been considered prior to the commencement of this research. An analysis routine (section 2.3) using the nodal admittance matrix (section 2.2) had been developed by Cutteridge and di Mambro[13,21-24] and had been found to be at least as efficient as other methods[13,21].

The state-variable method of analysis does have advantages in that it can be easily extended to include non-linear elements, and also provides useful information about the performance of the network. Di Mambro[13] disregarded this alternative partially as a result of the findings of Pottle[38] who declared this method to be less accurate and more time consuming in determining the coefficient values. Later, however, Pottle[39] devised an improved method.

As the derivatives are readily available, optimization routines using this information are preferable to direct methods (such as the Simplex method) which use only function evaluations. The potency of the direct methods decreases as the number of variables is increased. Two-part programs combining a powerful Newton method with a stable gradient method were found to be preferable to quasi-Newtonian methods. Originally a combination of the conjugate gradient (section 2.5) and GN techniques (section 2.6) was employed but an alternative gradient method using second order derivatives (developed at Leicester by Cutteridge, Henderson and Dowson[40-42]) was investigated. The criteria developed to instigate switching between the gradient and GN techniques are described in Chapter 3.

2.2. The Nodal Admittance Matrix and the Short-Circuit Admittance Functions

Nodal admittance analysis has two major advantages over mesh impedance analysis.

(1) The nodal admittance matrix is more easily formulated automatically. It is not easy to determine the number of independent loops and to specify these loops on the computer.

(2) There will be fewer nodes than meshes in a network and consequently the arrays will be smaller, occupying less computer storage and the analysis will be performed more quickly. The speed of computation can also be aided by the sparsity of the nodal matrix enabling the use of various time saving techniques[43-46]. (These techniques are insignificant with the order of networks examined in this thesis and are not employed.)

Consider a 3-terminal network as shown in Figure 2.1. The external behaviour of the network is described by the relationship between the input (1 and 0) and output terminals (2 and 0) . Suppose that the network comprises of a total of n nodes (plus the reference node) and that the voltages of these n nodes above the reference node are $E_1, E_2 \ldots E_n$. The admittance of each branch between any two nodes i and j (Y_{ij}) is comprised of a combination of a maximum of three types of element.

Hence ,

$$Y_{ij} = (pC + G + L/p) \qquad (2.1)$$

where ,

p is the complex frequency variable

C is the capacitance in Farads between nodes i and j

G is the conductance in Siemens between nodes i and j

and L is the inverse inductance in Henrys^{-1} between nodes i and j .

If Kirchhoff's current law is applied to each node in turn and Ohm's law is used to determine the current in each branch, the n equations can be arranged to the form,

$$\begin{array}{lllll}
I_1 = Y_{11}E_1 & - Y_{12}E_2 & - Y_{13}E_3 & & - Y_{1n}E_n \\
I_2 = -Y_{21}E_1 & + Y_{22}E_2 & - Y_{23}E_3 & & - Y_{2n}E_n \\
0 = -Y_{31}E_1 & - Y_{32}E_2 & + Y_{33}E_3 & & - Y_{3n}E_n \\
\vdots & & & & \\
0 = -Y_{n1}E & - Y_{n2}E_2 & - Y_{n3}E_3 & & + Y_{nn}E_n
\end{array} \qquad (2.2)$$

where Y_{ii} is the sum of the admittances connected to node i . (There are no external drives to nodes 3 , 4 ... n)

From Cramer's rule, and expanding down columns one and two

$$\begin{aligned} E_1 &= Z_{11}I_1 + Z_{12}I_2 \\ E_2 &= Z_{21}I_1 + Z_{22}I_2 \end{aligned} \qquad (2.3)$$

or, in matrix form

$$\begin{bmatrix} E_1 \\ E_2 \end{bmatrix} = \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{21} & Z_{22} \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} \qquad (2.4)$$

with $Z_{11} = \frac{\Delta_{11}}{\Delta}$, $Z_{12} = \frac{\Delta_{21}}{\Delta}$, $Z_{21} = \frac{\Delta_{12}}{\Delta}$ and $Z_{22} = \frac{\Delta_{22}}{\Delta}$

where Δ_{ij} are the cofactors of the determinant of the admittance matrix Δ .

Hence, using the relationship that

$$\Delta\,\Delta_{1122} = \Delta_{11}\Delta_{22} - \Delta_{12}\Delta_{21} \qquad \text{(a form of Jacobi's theorem)}$$

$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \qquad (2.5)$$

where $y_{11} = \frac{\Delta_{22}}{\Delta_{1122}}$, $y_{12} = -\frac{\Delta_{21}}{\Delta_{1122}}$, $y_{21} = -\frac{\Delta_{12}}{\Delta_{1122}}$, $y_{22} = \frac{\Delta_{11}}{\Delta_{1122}}$

For networks comprised of reciprocal elements (such as RLC networks) then $y_{12} = y_{21}$. If $y_{11} = y_{22}$ (and hence $\Delta_{11} = \Delta_{22}$) the network is said to be symmetrical.

For RLC networks, the three short-circuit admittance functions (y_{11} , y_{12} , y_{22}) and, therefore, the four cofactors (Δ_{11} , Δ_{12} , Δ_{22} , Δ_{1122}) fully define the external performance of a network. The coefficients of the rational functions in p (the complex frequency variable) of these four cofactors are those used in the coefficient matching process. The term short-circuit is derived from the relationships

$y_{11} = \frac{I_1}{V_1}$ when nodes 2 and 0 are short-circuited

$y_{22} = \frac{I_2}{V_2}$ " " 1 and 0 " " "

and $y_{12} = \frac{I_1}{V_2}$ " " 1 and 0 " " " (2.6)

The roots of the polynomials Δ_{11} , Δ_{12} and Δ_{22} are the zeros and Δ_{1122} the poles of the network admittance functions. An alternative representation is

$$y_{ij} = r_{\infty}p + r_o + \sum_{i=1}^{m} \frac{r_i p}{(p + \alpha_i)} \qquad (2.7)$$

where r_{∞} is the residue of the pole at infinity,

r_o is the residue of the pole at zero,

and r_i is the residue of the pole at $-\alpha_i$.

If the residues of y_{11} , $-y_{12}$ and y_{22} of a particular pole are denoted by r_{11} , r_{12} and r_{22} then Cauer's residue condition states that

$$r_{11}r_{22} - r_{12}^2 \geqslant 0 \ . \qquad (2.8)$$

When $r_{11}r_{22} = r_{12}^2$, $r_{12} \neq 0$ the pole is said to be compact. A further point of interest arises from a study of graph theory[47]. This indicates that the 2-trees of Δ_{12} are included in the 2-trees of Δ_{11} and Δ_{22} . Consequently 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[14]. Therefore the two-trees of Δ_{01} and Δ_{02} for this power of p must be zero.

Further information as to the values of poles, residues and zeros for particular network configurations and types is available in the standard texts.

2.3. Calculation of the Coefficients and their Derivatives

An analysis routine which can calculate quickly and accurately the coefficient values and their derivatives with respect to the elements, is essential. Such a method was developed by Cutteridge and Di Mambro[13,21-24], who found it compared favourably with other methods available at that time. Reference to the state variable method and the findings of Pottle[38,39] has already been made.

Di Mambro[13] also considered the Adjoint[10,48-51] and similar methods. Although there is no actual data for a comparison of the accuracy and computation times. the Adjoint would seem to be a less direct method of calculating the derivatives than by simply determining the cofactor values.

A further method of analysis, using fast Fourier transforms[25-26], was investigated at Leicester by Dr. Banerjee. The results[27] were not found to be superior, in terms of either speed or accuracy, to the method of Cutteridge and Di Mambro for the order of networks investigated in this thesis.

The cofactors of the nodal admittance matrix are in the form of polynomials in p . When inductors are included in the network the admittances of each branch are multiplied by p to remove any inverse powers. It is impractical to manipulate the powers of p when determining the coefficient values, so $m + 1$ values are assigned to 2u p , where m is the highest order of the polynomials produced. The $m + 1$ values of p $(p_1 , p_2 , p_3 \ldots p_{m+1})$ give $m + 1$ equations of the form

$$\Delta(p) = C_0 p^0 + C_1 p^1 + C_2 p^2 + \ldots C_r p^r \qquad (2.9)$$

where $\Delta(p)$ is the value of a cofactor evaluated at a particular value of p , and c_i are the coefficient values to be calculated. Thus,

$$\begin{bmatrix} 1 & p_1 & p_1^2 ----- p_1^r \\ 1 & p_2 & p_2^2 ----- p_2^r \\ \cdot & & \\ \cdot & & \\ \cdot & & \\ \cdot & & \\ 1 & p_{r+1} & p_{r+1}^2 --- p_{r+1}^r \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ C_r \end{bmatrix} = \begin{bmatrix} \Delta(p_1) \\ \Delta(p_2) \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \Delta(p_{r+1}) \end{bmatrix} \qquad (2.10)$$

The values of the coefficients are obtained by multiplying each side of equations 2.10 by the inverse of the Vandermonde matrix (the m+1 by m+1 matrix). The values of the derivatives can be calculated in a similar manner. Values of $\frac{\partial c_i}{\partial x_j}$ will be obtained if values of $\Delta(p_i)$ are substituted by values of $\frac{\partial \Delta}{\partial x_j}(p_i)$. The speed of the calculation is increased by

(i) using the information that the derivative of a determinant with respect to one of its elements is equal to the cofactor of that element,

(ii) calculating the values for the cofactor Δ_{1122} first, since this has terms in common with the other cofactors.

For a more detailed explanation of the method see Di Mambro[13,21-24]. This method can also be applied to active devices[13,21].

2.4. Formulation of the Coefficient Matching Technique

Although this will not always be the case, the examples considered in this thesis required the cofactors of Δ_{11}, Δ_{12}, Δ_{22} and Δ_{1122} of the admittance matrix to be realized.

Let,

m be the total number of coefficients of the four cofactors, all of which have to be realized. (No further coefficients should be generated by the synthesized network.)

n be the total number of independent variables, comprising of elements and common factors

a_i $(i = 1$ to $m)$ be the required values of the coefficients (as modified by the presence of any common factors)

c_i $(i = 1$ to $m)$ be the values of the corresponding coefficients generated by the network variables, x_j $(j = 1$ to $n)$

k be the normalizing variable

The proximity of each actual coefficient (c_i) to the desired value (a_i) is indicated by the individual error functions f_i $(i = 1$ to $m)$. A progress function, F, is constructed from these individual error functions. If these f_i are devised such that $f_i = 0$ when (and only when) $c_i = ka_i$, then the design problem is in a form ideally suited to optimization by computer, namely, to modify the values x_j such that an overall error function F is reduced.

Calahan's original formulation[9] was

$$f_i = c_i - ka_i \tag{2.11}$$

This gives an absolute measurement of the error. However, this has a trivial solution (if sufficient of the x_j equal zero for all the c_i to be zero) which causes practical problems[16]. The formulation predominantly employed in this research was

$$f_i = \frac{c_i}{ka_i} - 1 \quad . \tag{2.12}$$

This gives a relative value of the variance between the actual and the desired coefficients which is theoretically preferable, particularly when the required coefficients vary by orders of magnitude. The progress function was constructed from the sum of the squares of the f_i , namely,

$$F = \sum_{i=1}^{m} f_i^2 \ . \tag{2.13}$$

It can be seen that the individual functions were identically weighted. This representation (equations 2.13) does not produce discontinuities in the values of the derivatives, as does the minimax representation (equations 2.14), and also enables

$$F = \sum_{i=1}^{m} |f_i| \tag{2.14}$$

the optimum value of k to be easily calculated. (See Appendix 1)

Although it is the coefficients of the cofactors which are matched, the external behaviour of a network is actually defined by the short-circuit admittance functions. As these functions are ratios of two of the cofactors, if all the cofactors are multiplied by the same factor the network will retain the same external characteristics. This extra factor can be any combination of three possible forms:

(i) a scalar quantity known as the normalizing variable

(ii) a power of the complex frequency variable p^n, where n is an integer

(iii) a polynomial in p. This can be factorized into individual common factors of the form $(p + \alpha)$. α is referred to as the value of the common factor whose form (positive real, complex, etc.) will depend on the types of the components in the network.

Krzeczkowski[16] determined that the optimization proceeded in the most efficient manner when the normalizing variable was set to its optimum value at each function evaluation. Each common factor will introduce an extra coefficient to each cofactor. The variation of the common factor values is discussed in section 3.5.

Transformations to the logarithmic domain and to the square domain have both been considered as possible methods to ensure that the variables retain positive values. However, potential solutions with negative valued elements are not ignored, but are mapped into the solution space where they can still have a detrimental effect. The square transformation has usually been employed[13,21], in preference to the logarithmic transformation, with the intention of avoiding potential numerical difficulties as the variables tend to zero values. Krzeczkowski[16] found that these difficulties did not arise. In fact, the logarithmic transformation has the advantage of natural scaling which Krzeczkowski found beneficial stating; "This constraint improves the rate of convergence of the algorithms most markedly".

2.5. The Conjugate Gradients Algorithm

To prevent the error function F increasing between successive iterations, it was found necessary to modify the linear search of the original algorithm published by Fletcher and Reeves[32]. The details of these modifications can be found in section 3.7. The original Algol version was transcribed into Fortran by the author.

The efficacy of several alternative optimization routines had been investigated prior to the commencement of this research. Although the basic GN algorithm has been improved by the implementation of factors such as a linear search, working in the domain of transformed variables, etc., the algorithms still only converge consistently from values close to those at a minimum of the error. In order to develop more powerful optimization routines, two-part programs have been investigated. These generally employ either, a gradient descent method or, a modified NR technique which can approximate to a gradient method (e.g. Levenberg) prior to the implementation of the GN . Both alternatives were examined at Leicester prior to this research. Approaches using a true gradient descent method were found to be superior to the quasi-Newton techniques[13,36,52]. Of the improvements on the basic steepest descent algorithm, the conjugate gradients method was found to be superior to the variable metric method of Fletcher and Powell[53] and its modified counterpart (1970).

As a further alternative, it is possible to construct simple function minimization procedures which exploit the multilinear properties of the network coefficients. Such a method was described by Massara and Fidler[54]. This was shown to have rapid initial convergence characteristics but k , the normalizing variable, was omitted without explanation.

The conjugate gradients method is guaranteed to locate the minimum of any quadratic function of n arguments in at most n iterations. As this is not the case with network synthesis, the process is iterative with the search set to the steepest descent direction at the first iteration and reset after each n iterations.

Using the superscript k to denote values at the kth iteration, the search direction is governed by

$$\underline{p}^{k-1} = \underline{g}^{k-1} + \beta\, \underline{p}^{k-1} \qquad (2.15)$$

where $\underline{g}$ is the gradient vector of F with respect to the n variables of $\underline{x}$

and $\beta = 0$ if $k = rn + 1$

(where r is an integer of value zero or greater)

$$\text{or} \quad \beta = \frac{\underline{g}^{k-1} \cdot \underline{g}^{k-1}}{\underline{g}^{k-2} \cdot \underline{g}^{k-2}} \,. \qquad (2.16)$$

This modifies the vector of the current values of $\underline{x}$ (x_j, $j=1$ to n) such that

$$\underline{x}^{k} = \underline{x}^{k-1} + \lambda\, \underline{p}^{k-1} \qquad (2.17)$$

where λ is a scalar calculated (by a linear search) to minimize the error value F .

To reiterate, the CG is only useful in the initial stage of a two-part optimization algorithm as the good progress initially obtained tails off.

2.6. The Gauss-Newton Algorithm

The Taylor Series expansion of a function $f(x)$ is

$$f(x + \delta x) = f(x) + \delta x f'(x) + \frac{\delta x^2}{2!} f''(x) + \ldots \qquad (2.18)$$

If δx is such that $f(x + \delta x) = 0$ and second order terms and above are ignored,

$$- f(x) = \delta x . f'(x) \qquad (2.19)$$

Generalizing to n functions $\underline{f}$ of n variables $\underline{x}$

$$- \underline{f}(\underline{x}) = J.\underline{\delta x} \quad - \text{ the symbolic NR algorithm}$$

i.e. $$\underline{\delta x} = - [J]^{-1}.\underline{f}(\underline{x}) \qquad (2.20)$$

where $[J]$ is the Jacobian matrix $(n \times n)$ of first derivatives.For an overdetermined system (m equations in n variables where $m > n$) a least squares solution to the set of linear equations can be obtained by premultiplying by J^T , the transpose matrix of the Jacobian.

$$- J^T \underline{f(x)} = J^T J \underline{\delta x}$$

giving $$\underline{\delta x} = - [J^T J]^{-1} \underline{J}^T \underline{f(x)} \quad - \text{ a symbolic form of the Gauss Newton algorithm} \qquad (2.21)$$

The corrections $\underline{\delta}$ can best be found by using a routine which uses Householder's transformations.

The variables are modified thus:

$$\underline{x}^{k} = x^{k-1} + \lambda \ \underline{\delta x}^{k-1} \qquad (2.22)$$

where λ is a scalar value obtained by a linear search to minimize the value F. This encourages convergence of the method.

2.7. A Steepest Descent Algorithm Utilizing Second Order Derivatives

This further gradient descent method was investigated, in place of the CG method, as the preliminary stage of a two-part program. This method had been devised at Leicester by Cutteridge and Henderson[40,42] and an improved version, as developed by Dowson[41], was supplied to the author. The method had previously been successfully applied to a transistor modelling problem with which difficulty had been experienced using the alternative gradient methods which used only first order derivatives[36]. This method produces several minima at each iteration and, consequently, enables restarts to be made from alternative positions if the optimization is proceeding at a slow rate in an initial direction.

The conventional method of steepest descent can be written as

$$\delta x_j = - \mu \frac{\partial F}{\partial x_j} \qquad (2.23)$$

where, $\mu > o$ for descent and μ is chosen (by a linear search) to minimize the value of F.

Extending this to include the second partial derivatives of F

$$\delta x_j = - \mu \left\{ \frac{\partial F}{\partial x_j} + \left(\sum_{i=1}^{m} \frac{\partial^2 F}{\partial x_i \ \partial x_j} \right) \right\} \qquad (2.24)$$

or, in vector matrix form,

$$\underline{\delta x} = - \mu [G + H \underline{\delta x}] \qquad (2.25)$$

where G is the gradient vector

and H is the Hessian in F .

Hence,

$$\underline{\delta x} = - (H - \lambda I)^{-1}.G \, . \qquad (2.26)$$

Thus $\lambda = + \infty$ and $\lambda = - \infty$ correspond to the directions of steepest ascent and descent, respectively.

The eigenvalues, λ_i , are calculated to locate the discontinuities of the function $F(\lambda)$. This enables all the minima of this multimodal function to be efficiently determined.

In practice, the moduli of the corrections are limited so that the variation of each variable per iteration does not exceed a prescribed value. This introduces further discontinuities into the function $F(\lambda)$. Their positions can be calculated[41], thus facilitating the linear search. It is usual to proceed from the minimum producing the best reductions in error unless the restart facility is invoked.

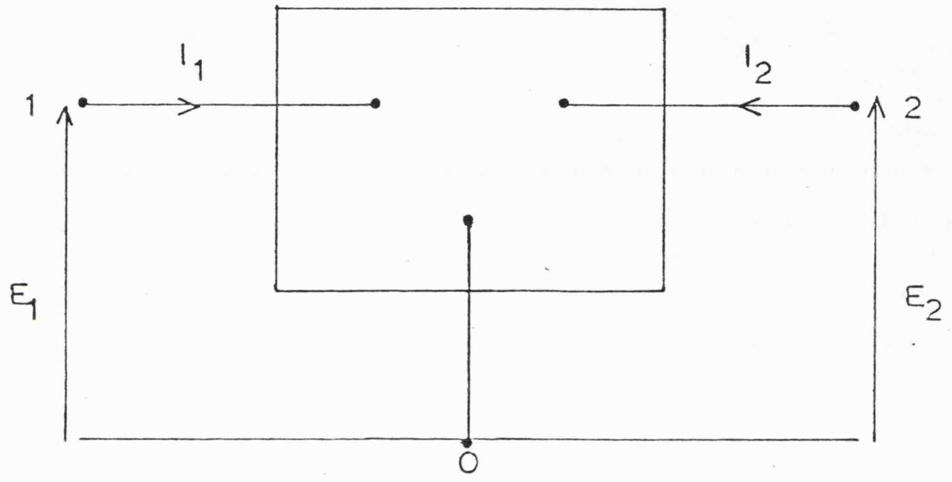


FIGURE 2.1: THREE-TERMINAL RLC NETWORK

CHAPTER 3

THE DEVELOPMENT AND STRATEGIES OF AN AUTOMATED DESIGN PACKAGE

3.1. Introduction

Although the basic concepts of the coefficient matching technique have been described, there are many possible options available, within this framework, which will affect the performance of an automated package. This chapter contains discussion of several possible variations, together with the author's ideas on the options, which, when combined, will provide the optimum package. As each point is discussed the original and final treatments are described, thus clarifying the achievement of this research. The inherent strategies of network evolution are also described. Among the topics considered are:

(i) the inclusion of algorithms to ensure that the coefficients are matched to the desired values as efficiently as possible (section 3.3)

(ii) the optimum error function formulations (section 3.4)

(iii) the representation and variation of common factors (section 3.5)

(iv) switching criteria for the two-part optimization algorithms (section 3.6)

(v) modification of the linear searches included in the optimization routines (section 3.7).

(vi) development of criteria to activate both the addition and removal of individual elements and nodes (sections 3.8, 3.9 and 3.10)

(vii) the strategy towards the acceptance of element additions (section 3.2)

Chapters 4 and 5 are supplementary to this chapter. In Chapter 4 the author details the results of several possible methods of element addition, considering both the actual choice of element and its value on addition. Also included in this chapter is detailed discussion of the alternative strategies applied to several particular additions which failed using the general approach. Chapter 5 considers the specific topic of node addition.

There are two possible evolutionary strategies which can be adopted in network design. Firstly, elements can be added to a starting topology with a small structure just capable of realizing the polynomials of the required network (termed a minimal structure). Alternatively, a larger starting topology with several nodes more than the required number will, hopefully, remove unnecessary nodes and obtain a reasonable approximation to the requirements. After consideration of the relative merits of the two options, the first strategy was adopted. Although both alternatives had previously produced worthwhile results,

(1) the time required for each iteration increases significantly as the number of nodes increases (approximately doubles on increasing an RC network from five to seven nodes),

(2) optimization routines work more effectively with fewer variables. Consequently, if the original

network is too large, an excess of redundant elements can prevent the optimization routine from locating the required values of the elements which are capable of providing a good approximation to the desired network. (Generally, the first minimum is the most difficult to locate when the optimization commences with arbitrary element values),

(3) even when a larger network has been reduced it is unlikely that an exact realization will have been produced. Therefore, element additions will still be required.

Obviously, if a solution is known to a network similar to that required, then this would provide a suitable start structure.

3.2. The Strategies Governing the Evolution of a Design

A starting topology, which satisfies the order of the desired network polynomials, is specified at the start of each design. Certain other network characteristics (such as symmetry) may be required of the start topology. The values of the elements, and any common factors present, are varied by the optimization routines such that the overall error function F is reduced. F is a squared summation of the individual error functions, f_i , which provide a measure of the variance between the actual and the desired values of the coefficients. Since the element values are required to be positive, it may prove necessary to remove an element, and possibly an associated common factor, when mapping to a negative valued element occurs. (Active devices are only equivalent to a negative element over a small range of voltages, currents and frequencies[28].)

Generally, the optimization will continue, provided the job-time has not been exceeded, until one of the following situations has been diagnosed:

(i) the synthesized network is a sufficiently accurate representation of the desired network,

(ii) the synthesized network is a better approximation to the desired network than can be obtained anywhere in the surrounding region of vector space, provided the topology remains constant. (This may be a local, rather than a global, minimum of the error.)

(iii) the progress may be unacceptably slow (as gauged by the error F), with no indication of imminent improvement (gauged by the sum of the squares of the Gauss Newton corrections -SSQGNC)

(iv) the network may be incompatible with the GN algorithm (i.e. the variables are not independent).

If situations (i) or (iv) are diagnosed then the program will stop. Otherwise, the program will investigate the possibility of topological additions.

The program will include, first of all, elements which do not introduce further common factors, then elements which produce extra common factors and, finally, groups of elements which introduce a new node. With the method of node addition developed by the author, and similarly with other methods, node addition will introduce an extra common factor and a further three elements. Optimization routines work with increasing efficiency, in terms of both the results achieved and the time required to achieve them, with decreasing numbers of variables. Hence the three preference categories for additions. Furthermore: the time required for each analysis increases as the number of nodes increases; the optimization routines deal more effectively with elements than common factors; and element additions without the introduction of extra common factors are performed more efficiently than both those with an accompanying common factor and node addition.

Within each of these three groups an order for addition is established. The element addition, say, which the algorithms indicate will reduce the error by the largest amount is included first, etc.

Once the addition has been performed the variables are optimized (with the possibility of removals) until the algorithms diagnose one of the four defined situations. If situation (i) arises the package terminates. Otherwise a comparison is made of the error value at this endpoint with the error prior to the last addition. If the error has increased, then the program restarts from the next best addition from the previous endpoint. Otherwise, a further addition is made from the new endpoint. To ensure that the design proceeds efficiently, nodes are only included from positions where a local minimum has been obtained (i.e. case (ii)).

After each element removal it is necessary to establish if the network still satisfies the orders of the desired polynomials. A fault of this type could be remedied by the inclusion of further elements[13,18]. At such a position, however, the error will increase to a high value (one or more of the coefficients are no longer represented) as with the introduction of an extra common factor. Consequently, the additions will not be as reliable as those from a minimum. The author adopted a strategy whereby a restart was made using the next best alternative to the previous addition.

It is possible that the removal of an element will effectively remove a node. Similarly, node removal can be indicated by elements shorting out (section 3.9). With several of the earlier examples, the author found that the addition of a node resulted in the removal of one of the original nodes, without eventually reducing the error (i.e. the networks were equivalent). To avoid this type of unnecessary operation, node removals were not performed. When removal was indicated, the program restarted with an alternative to the previous addition.

3.3. Topological Considerations

The original Algol package had a very limited repertoire of possible topological modifications. The only element types considered were resistors and capacitors (RC networks) and although element additions and removals were possible, changes in the number of nodes were not. The package simply matched a set of synthesized coefficients to a prescribed range of desired coefficient polynomials. Several faults of the program were pertinent to these limitations, namely,

(i) if the synthesized network was at any stage producing zero values for coefficients which were required to be matched, the program continued to optimize although little could be achieved under these conditions,

(ii) if extra coefficients were generated (by the synthesized network) outside the prescribed range of optimization, the program continued although the results were worthless.

(iii) the number of common factors was fixed initially at one or zero and remained at this value irrespective of the requirements.

The Fortran version developed required only the orders of the desired polynomials and their coefficient values and not the actual orders of the start network. The orders of the polynomials produced by the synthesized network were calculated initially, and after each topological modification, and the number of common factors determined. (If the start network was unable to produce the polynomials required the program terminated. However, elementary topological modifications to rectify such a fault could quite easily be included[13,18].) It was only necessary for the initial number of common factors to be declared if they were to be assigned specific values, otherwise they were set to integer values.

The coefficients of Δ_{12} are fully represented in the Δ_{11} and Δ_{22} polynomials[14]. Consequently, if a particular power of p has identical coefficient values in the polynomials of Δ_{11} , Δ_{12} and Δ_{22} , the same combination of elements form these coefficients. Accordingly, a network with an element which contributes to the particular coefficient in at least one polynomial, but not all three, cannot exactly realize the requirements. Consider Lucal's set of functions, namely,

$$y_{11} = \frac{36p^4 + 2058p^3 + 6552p^2 + 4638p + 36}{36p^3 + 216p^2 + 396p + 216}$$

$$-y_{12} = \frac{36p^4 + 36p^3 + 72p^2 + 36p + 36}{36p^3 + 216p^2 + 396p + 216}$$

$$y_{22} = \frac{36p^4 + 533p^3 + 1572p^2 + 1183p + 36}{36p^3 + 216p^2 + 396p + 216} \qquad (3.1)$$

The coefficients of p^0 and p^4 are identical for Δ_{11} , Δ_{12} and Δ_{22} .

For a true RC realization of these functions there should be no purely capacitive, or purely resistive, path between nodes 1 (or 2) and 0. Similar conditions apply to RLC and RL networks. Intuitively it would seem that the temporary inclusion of elements which contravened these conditions would increase the capacity for topological modifications, thus increasing the likelihood of a realization. (A comparative example where this supposition is borne out is included in Chapter 6.) However, in practice, the inclusion of these elements did not produce radical changes in the topology. Also, in many instances, the element addition algorithms were encumbered by these extra elements. For example, elements which produced negligible reductions in error were included prior to elements which could have provided a significant reduction. (In these instances it is possible to relate the size of the GN corrections obtained on introducing an element at its optimum value to the expected change in its value - see section 4.4.) Furthermore, as the error F decreased the values of the elements prohibiting the equality of these coefficients became smaller in relation to the other elements. The close proximity of these small values to the negative domain produced problems with mapping to negative values, thus several element additions were attempted which proved worthless because of the requirement for positive valued elements.

It should be noted that if individual elements are to be represented by models (as the capacitor of Figure 3.1) then this criteria could not be imposed.

The author considered it to be generally advisable to retain a symmetrical network (by the simultaneous addition of a pair of elements) when this was required (i.e. $\Delta_{11} = \Delta_{22}$). This facet was ignored in the original Algol program. The results were similar to those obtained with the previously discussed restrictions to element additions.

Although it seems reasonable to postulate that the inclusion of all possible elements will provide a more powerful alternative, in practice the unsymmetrical additions rarely produced significant improvements, and then only before the error had been reduced by a significant amount. Similarly, the problems encountered increased as the error was reduced (say, to 10^{-6}). On occasions, the inclusion of single elements, or of symmetrical pairs, will not be possible. (The introduction of a node is required.) However, the algorithms will indicate that single elements which destroy the symmetry can be included at very small values. This is because, with a computer of finite accuracy, it is not possible to synthesize an exactly symmetrical network.

Algorithms were included to diagnose symmetrically matched nodes from the values of the coefficients and derivatives obtained from a unit valued analysis. Similarly, element removals were performed symmetrically.

The designs which considered only resistors and capacitors are valid because of the practical difficulties associated with inductors. (Inductors required to produce reasonable Q factors at low frequencies are bulky and expensive, etc.) However, the character of feasible networks that can be produced by an RC network is limited and, because of the problems of instability associated with an active network (these can result from the effects of parasitics and consequently be unforeseen), RLC designs are of merit.

The designs in this research usually commenced from an RC start network and included inductors if required. Generally, the algorithms successfully included the best inductor. This produced topologies which were predominantly RC dominated and several possible RL dominated configurations were not considered, at least not initially. The program considered only RC and RLC networks so it was not possible to add the optimum capacitor to an RL equivalent of an RC network. This limitation of the current package could quite easily be rectified.

3.4. Individual Error Functions

The original program provided an option between two possible error functions, namely,

Formulation 1 $f_i = \frac{c_i}{ka_i} - \frac{ka_i}{c_i}$

Formulation 2 $f_i = \frac{c_i}{ka_i} - 1$.

where, to reiterate,

c_i are the coefficient values produced by the synthesized network,

a_i are the required coefficients as modified by the presence of any common factors

and k is the normalizing variable.

Previous research at Leicester[14,16] had indicated that option 1 was the more efficient alternative. However, this formulation also has a minimum when $c_i = -ka_i$ and so cannot be applied to the design of networks containing active devices. (The synthesized coefficients produced by an RLC network cannot have negative values with positive valued elements but may with an active network.)

Originally k was considered as a variable but set to an optimum value (see Appendix 1) at each function evaluation. As this optimum value for k is a function of c_i and a_i, its effect can be incorporated into the other variables, for example, for formulation 2

$$\frac{\partial f_i}{\partial x_j} = \frac{\partial c_i}{\partial x_j} \cdot \frac{1}{ka_i} + \frac{\partial k}{\partial x_j} \cdot \frac{c_i}{k^2 a_i} \qquad (3.2)$$

Thus, in theory, a further variable can be optimized by the GN algorithm. Krzeczkowski[16] stated that both methods of representation were equivalent but the author found that significant differences did arise when a substantial reduction in error was produced by one GN iteration. Typically, in a situation where the latter method of representation reduced the error from, say, 10^{-2} to 10^{-5}, the former reduced the error by a smaller factor, say, to 10^{-4}. The latter method of representation was adopted in this research.

It is possible to derive optimum values for k and one common factor for the further alternative

<u>Formulation 3</u> $f_i = \frac{ka_i}{c_i} - 1$.

This was examined both when the common factor was set to the optimum value and also when varied normally. (The jobs were run shortly after the commencement of the research and the criteria for switching between CG and GN iterations were similar, but not identical, to those described in section 3.6.)

The most striking feature of the test examples was the disappointing results (Table 3.1) obtained with formulation 3 when applied to Fialkow's functions,

$$y_{11} = y_{22} = \frac{1197p^3 + 56,613.14p^2 + 28,368.584p + 191.184}{800,000p^2 + 408,000p + 3840}$$

$$-y_{12} = \frac{3p^3 - 1.14p^2 + 197.176p + 77.616}{800,000p^2 + 408,000p + 3840} \qquad (3.3)$$

The common factor and several elements that were not connected to nodes 1 and 2 were invariably reduced to very low values. This effect was particularly marked when the common factor was set to its optimum values but was apparently caused by the error formulation itself, as indicated by the small common factor values obtained from starting values between 1 and 10 with the common factor varied normally (Table 3.2). These results refer to the topology of Fig.3.2. Several local minima can be obtained (Table 3.3) with the largest changes affecting the capacitor between nodes 0 and 5 (C_{05}) and the common factors.

The optimum values for the common factor and k appeared to result in large modifications to these values before the optimization had significantly modified the element values. Consequently, the optimization always commenced from a disadvantageous position. The fact that some runs were successful and that the global, rather than a local, minimum was always located, does indicate the power of this formulation.

Results with a larger network (Fig.3.3), applied to Lucal's functions (equations 3.1), showed a significant improvement (Table 3.4). The two common factors were set to the optimum value in alternate iterations and varied normally in the others.

A brief study was undertaken to ascertain if improved results could be obtained using formulation 3 by starting with an ordinarily varied common factor, then switching to the optimum values. In general, it proved difficult to determine when the values had been sufficiently modified to obtain maximum efficiency. A premature switch reproduced the same deficiencies and a delay resulted in the location of a local rather than the global minima.

To summarize, the results obtained by formulation 3 were not superior to those obtained by the original formulations. Further, formulation 3 does not enable closed form expressions to be developed for the optimum value of a new element. These optimum values had provided the basis of the original method of element addition and are still used to establish an order of preference for addition. The optimum values calculated for formulation 2 were not comparable to those of formulation 3, particularly at high errors when these values are of most use. However, they would probably constitute a good starting position for a linear search.

Formulation 2 was employed in preference to formulation 1 for the following reasons:

(a) the results confirmed that 1 is a more powerful formulation than 2 in that the optimization proceeds successfully from a wider range of starting values. However, from starting values of unity (the mean value in the logarithmic domain) the results were invariably very similar;

(b) closed form expressions for the optimum values of virtual elements are available for method 2 but not 1 . (Linear searches can prove time consuming and inaccurate)

(c) As discussed in section 3.6, the author preferred to proceed initially using the CG algorithm before employing the GN . When using formulation 2 it is easier to gauge when the elements have been sufficiently altered to enable the GN to progress. Formulation 2 has a maximum error value of M (see Appendix1)where M is the number of coefficients synthesized, whereas formulation 1 has no maximum value. Consequently, it is more difficult to determine automatically at a high error value if the synthesized network is a good approximation with a poor topology, or if the element values are far removed from those required. Consider, for example, the attempted synthesis of Fialkow's functions with the network of Fig.3.4. As the poles at zero and infinity are not compact for these functions, this topology cannot provide a good approximation to the network. With formulation 2 a minimum can be obtained with an error value of 3.56. However, with formulation 1 the error cannot be reduced below a value of 1.18×10^3, although this is not a minimum as signified by the SSQGNC. This value of error is higher than the starting error values for networks of corresponding topology synthesizing other sets of functions,

(d) method 1 cannot be used to optimize networks containing active devices.

Although method 2 was used predominantly, when difficulties were encountered formulation 1 was considered as a possible alternative without ever producing significant improvements.

3.5. Common Factors

The original package allowed for only one common factor, to be fixed, or varied, as required. Algorithms had not been included to determine analytically the number of common factors required initially or, after topological modifications. Hegazi[14] investigated networks with a maximum of three variable common factors and considered it necessary to reduce the values of the GN corrections for the common factors and, in certain situations, to vary only the common factors, and not the elements, and vice versa. The author considered it satisfactory to vary the common factors in the same manner as the elements. This approach can be justified by a comparison of the lowest error values attained (by the author and Hegazi) for similar computer realizations with the same computer accuracy.

Initially, only RC networks were considered and the package was modified to manipulate a maximum of three common factors $(p + A)(p + B)(p + C)$, where A, B and C were varied. There are two basic difficulties with this method. Firstly, the formulation of the derivatives cannot be readily generalized and thus is laborious, requiring separate options for each given number of common factors. (see Appendix 4). Secondly, common factors having complex values are feasible when RLC networks are considered.

Minima of error with complex common factors will not simply be ignored when this method of representation is employed. Consider the network of Fig.3.5 which is generated during an example to realize Fialkow's functions. The inductor (I_{03}) has just been included from a minimum of error 4.99×10^{-6}. On entering the GN, at an error of 8.39×10^{-3}, the corrections for the common factors (values 0.773 and 0.774) are 542 and - 543, whereas the largest correction for an element is 0.752. The GN algorithm was unable to reduce the error. When an alternative method of representation was used (discussed below) the optimization passed through a region where the common factors were complex, but produced a minimum of error of value 1.38×10^{-8} with real common factors.

$$(p + A)(p + B)(p + C) = p^3 + p^2 (A + B + C) + p (AB + AC + BC) + ABC \qquad (3.4)$$

As an alternative to varying the common factors, the polynomial coefficients (e.g. ABC) can be varied. These values must remain positive for an RLC network and this representation permits any combination of real and complex common factors, although it is possible that a complex pair of the form $(p^2 + A)$ will cause numerical difficulties because of the zero valued p coefficient. This method also has the advantage that general formulae for increasing, or decreasing, the number of common factors can be easily obtained. Furthermore, the number is limited only by the space allocated. Moreover, only first order derivatives exist whereas with the alternative method there can be second and higher order derivatives, dependent on the number of common factors. It could therefore be expected that this method would improve the optimization, even with RC networks, as the algorithms use only the first derivatives. This was verified with several test examples.

It is possible to obtain a good approximation for the value of any common factor to be removed. For example, if a node, prior to its removal, has admittances totalling C farads and G siemens connected to it, then the common factor removed will be of value G/C . The author's research considered only RC and RC dominated RLC networks, and node removals were not attempted. Consequently, common factors were predominantly removed accompanying the removal of a resistor or an inductor. As these had been reduced to a low value prior to their removal, the associated common factor would also be small. Thus, the term ABC , say, would be small. Consequently, it proved sufficient to remove the ABC variable and to not calculate the approximate value of the common factor and adjust the remaining terms.

3.6 The Use of Two-Part Optimization Routines

The program originally employed a strategy whereby a series of GN iterations were performed immediately. If this series failed to progress sufficiently, the original values were varied by a CG iteration and a further series of GN iterations were performed. Thus, the more powerful GN algorithm was applied to starting values obtained from the more stable CG algorithm. The author was of the opinion that the progress achieved by this strategy was unsatisfactory, even with small RC networks (5 nodes plus earth), exhibiting faults which were amplified when applied to larger networks. The major faults exhibited when examples were run from arbitrary values were

(1) Even with topologies known to realize a solution (as Fig.3.3) the entire job-time often elapsed (up to 600 seconds on an ICL 1906A machine) without producing a significant reduction in the error value. It was noted in these instances that isolated CG iterations were reducing the error by a larger factor than an entire series of GN iterations.

(2) Elements were unnecessarily removed at high errors from solution, and other, topologies.

(3) Common factors, and their associated elements diverged from the values at the solutions. This fault was less evident with error formulation 1 of section 3.4.

A modification to the implementation of the GN algorithm, whereby the maximum variation per iteration was limited for each variable, produced little improvement. Only rarely was the GN algorithm able to reach a minimum from a high initial error. Further, in situations where large changes in one or more variables were required, this limiting process was detrimental to the optimization.

Several factors were considered before the original strategy was modified so as to proceed initially using the CG algorithm alone, until the GN algorithm could proceed effectively.

(1) Using this approach with examples with which little had been achieved in 600 seconds, the solution was located in around 200 seconds. Similar improvements were experienced with a cross-section of examples. One noticeable effect of using the CG extensively was that, when several possibilities existed, there was a tendency for the same minimum to be located from different sets of arbitrary starting values. This particular minimum may not necessarily be the global minimum but, as networks evolve quite successfully via local minima, this is not a problem.

(2) A CG iteration can be performed in less time than a GN iteration. This is particularly true when the values are remote from those at the minimum of error where the GN corrections are large, requiring repeated function evaluations in the linear search.

(3) Although large fluctuations in the magnitude of the variables occur in the CG section, if the error has been reduced to a value of 10^{-2} , then the variables will generally have attained values of the correct order of magnitude. Proceeding beyond this point did

occasionally produce difficulties with the removal of elements. This is not strictly a fault of the optimization but merely exposes a limitation of the element removal algorithms of section 3.8. Also, when the GN algorithm is able to proceed, it will generally reduce the error at a faster rate than the CG algorithm.

(4) From arbitrary starting values, the initial CG iterations can modify the variables in the opposite direction to the actual requirements. This would appear to invalidate the original strategy whereby one CG iteration was expected to provide a significantly improved starting position for the GN algorithm.

(5) When remote from a minimum, it can prove difficult to gauge whether the GN will progress. Large GN corrections can be obtained when a suitable topology has values far removed from those required, but the number of iterations required before a minimum is located can be ambiguous. Alternatively, the corrections can be large because the variables are dependent, even when the number is significantly lower than the number of coefficients.

When commencing from arbitrary values, the first minimum will generally prove to be the most difficult to locate. However, after an addition (and particularly at high errors) elements previously included may require significant modifications to their values. These changes are achieved more readily using the CG rather than the GN algorithm. Also, after many additions the initial error will be increased to a higher value, equivalent to the errors obtained with the variables set to arbitrary values. Consequently, optimization using the CG section was employed after each topological modification, as well as with the initial network.

The final conditions developed were as follows:

If (1) the CG had diverged (i.e. been unable to locate a lower error) on 5 occasions.

or (2) 100 consecutive CG iterations had been performed,

the program switched to the GN algorithm, irrespective of the error value. Otherwise, the CG algorithm was employed until the error had been reduced below a value $M \div 3$, where M is the total number of coefficients. Again, unless conditions (1) or (2) had been violated, the CG algorithm was employed until the error had reduced below a value of 10^{-2} , provided that the error was being reduced at a prescribed rate, namely, a 5% reduction over the previous five iterations.

A steepest descent algorithm, developed at Leicester[40-42], was examined as a possible alternative to the CG algorithm. This algorithm used second order derivative information and had been considered primarily as a method of overcoming problems that were sometimes experienced with element additions (see Chapter 4). However, it was also applied to networks with variables set to arbitrary values. Its value as a plausible alternative optimization routine was undermined by the increase in time required per iteration as the number of variables increased. On occasions, with RC networks containing seven nodes, only seven consecutive iterations were performed in 600 seconds on an ICL 1906A machine. Consequently, the use of this algorithm in this manner, as an alternative to the CG algorithm, is not feasible, at least with the lumped, linear, passive networks considered. Furthermore, although the algorithm is more powerful than the CG algorithm (the error reduction per iteration is greater), it was not sufficiently powerful for a strategy of employing the GN after each iteration to be successful.

This algorithm had proved beneficial when applied to a transistor modelling problem with eight variables and eight equations. Two-part optimization programs, using a gradient descent and a GN algorithm, had located the solution from a few arbitrary starting values[36] but, in general, a local minimum of error had been located. Unlike a first order gradient descent method, the linear search of the second order method provides a number of local minima. If the error is reducing slowly, it is possible to restart from one of these alternative minima and, consequently, the solution can be located more often. However, in network synthesis the location of a local minimum need not prove detrimental. Often, new elements are either added successfully, or will be removed in the process of locating a better minimum. Also, the error functions representing the transistor modelling problem contained exponential terms. These could be expected to provide an extra level of difficulty.

3.7. Modifications to the Linear Searches of the Optimization Routines

Several restrictions were imposed on the linear search section of the CG algorithm of Fletcher and Reeves[32].

(i) The maximum number of function evaluations per iteration was limited to eleven (to prevent the program remaining in a loop as sometimes occurred),

(ii) the linear search scalar was prevented from searching in the negative domain,

(iii) the calculation of the square-root of a negative value was averted. (This can occur in the interpolation section.)

If for these, or any other reasons, the algorithm produced a one per cent increase in the error, the values were reset to those prior to the iteration and the next iteration performed.

The original linear search for the GN algorithm comprised of a golden-section search followed by a quadratic interpolation. Problems were only experienced originally when the error increased following the location of a local, rather than a global, minimum[14]. These problems were overcome by comparing the best error value obtained by the linear search to that prior to the iteration. If an increase of two per cent, or more, was recorded, the search was conducted in an alternative region.

Further problems arose when larger networks were considered. With RC networks containing seven nodes (as in Fig.3.3) several difficulties were experienced.

(1) The error was only being reduced to values of 10^{-12} with solution topologies, whereas errors as low as 10^{-20} had been obtained with five node RC networks.

(2) After successive CG iterations had substantially reduced the error, difficulty was often experienced in locating a lower error in the linear search of the GN algorithm.

(3) Local minima, as defined by the SSQGNC , were rarely located with non-solution topologies. In some cases

the program had to return to the CG section as the GN failed to locate a value of the linear scalar producing a lower error value. In other cases the error F fluctuated in value but the SSQGNC was relatively high compared to values previously obtained at a minimum.

It should perhaps be noted that these networks were predominantly series-parallel structures. It was later observed that non-series parallel structures are more readily optimized (i.e. the solutions are often reduced to lower errors and the minima are signified by lower values of the SSQGNC .)

It was postulated that the GN linear search would only fail to locate a lower error when close to a minimum. Elements were therefore included at these positions of failure, but the difficulties were not overcome. At this stage of the research the original method of element addition (based upon the optimum values of virtual elements[33] - section 4.2), was still employed. The efficacy of this method is greatly reduced when activated at a position which is not a true minimum of the error. As a result, what generally occurred was that an incorrect addition would be made, the linear search would again fail and so a series of poorly chosen elements were included. It is preferable, because of the reduced efficiency experienced, not to include elements in such a position, even with the improved method for element addition developed by the author (section 4.4). However, element additions at such positions can, if used in moderation, prove useful in by-passing occasional instances where the optimization is only proceeding at a poor rate.
The inclusion of an element at a value higher than the optimum value, as in the author's method of addition, can aid the optimization by-pass such a region of difficulty, even if the element included is actually superfluous to the requirements. However, it was not such isolated

instances of difficulty that were being experienced at this stage, but a persistent problem.

In an alternative approach to overcome these difficulties, attempts were made to emphasize the positions of the minima. Overall error functions, as in equations 3.5,

$$F = \sum f_i^{2n} \qquad \text{where } n \text{ is an integer} \geq 2 \qquad (3.5)$$

are possible but, with the individual error functions considered, an optimum value for k cannot be obtained without performing a linear search. A similar situation was produced by considering individual error functions of the form

<u>Formulation 1a</u> $f_i = \left(\frac{c_i}{ka_i}\right)^n - \left(\frac{ka_i}{c_i}\right)^n$

<u>Formulation 2a</u> $f_i = \left(\frac{c_i}{ka_i}\right)^n - 1$

These still emphasize the error functions of larger values in relation to the smaller ones and expressions for the optimum value of k can be derived (see Appendix 1). With n equal to two the situation was slightly improved.

A much more successful approach to the problem was to increase the accuracy of the linear search. The golden search section was removed as it had been contributing little and is not certain to isolate the region around the global minimum. The GN corrections had been scaled down, when necessary, to give a maximum value of three. Earlier research at Leicester had indicated that this value would, in general, shift λ (the linear search scalar) to a value of approximately one. However, it was not uncommon for the values to be less than 10^{-2}, particularly when a poor set of GN corrections had been obtained. The accuracy of λ was improved by further scaling of the values of the GN corrections. The error F was evaluated with λ equal to 1.0 and 0.8 giving values F(1.0) and F(0.8) . If F(0.8) was less than F(1.0) then the corrections were reduced by a factor of ten and the new F(1.0) and F(0.8) evaluated. Otherwise, F(1.2) was evaluated and the three values were used, in repeated quadratic interpolations, to obtain a value of λ between 0.8 and 8 . If this minimum value of F gave a two per cent increase on the error value prior to this iteration, then the corrections were further scaled by a factor of ten and a further search instigated. If ten such scalings occurred, λ was set to what was effectively a value of 10^{-10} . (In fact this only occurred when the error had been reduced to the limits of the computer's accuracy i.e. a computer realization.) The increased accuracy of the values of λ overcame the faults described previously. Minima were more easily obtained and easily defined from the values of the SSQGNC .

3.8. Element Removal

To ensure that they retain positive values, the variables are transformed into the logarithmic domain. However, potential minima with negative values are mapped into the solution space and the optimization attempts to give these elements negative values. As the transformed variables cannot be reduced to zero values, the necessary elements have to be removed whilst having a positive value. As different networks will require elements over a very wide range of values, it is not possible to have a carte-blanche policy of removing elements with values below a prescribed figure. (Practical considerations may require the values to be within certain limits.) However, it is possible to inspect the gradients and determine the extent to which an element contributes to the network[13,21]. The program originally employed a method, devised by Cutteridge[15], which proved efficient with smaller networks in removing elements prior to their reduction to insignificant values. Cutteridge stated that, (for the GN corrections)

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

i.e. if δ_j^k denotes the correction of element j on iteration k then

$$\text{If} \quad -\delta_j^{k-4}\ ,\ -\delta_j^{k-3}\ ,\ -\delta_j^{k-2}\ ,\ -\delta_j^{k-1}\ ,\ -\delta_j^{k} > 0\ .$$

$$\text{and} \quad |\delta_j^k| > |\delta_j^{k-1}| > |\delta_j^{k-2}| > |\delta_j^{k-3}| > |\delta_j^{k-4}|$$

$$\text{and} \quad |\delta_j^k - \delta_j^{k-1}| > |\delta_j^{k-1} - \delta_j^{k-2}| > |\delta_j^{k-2} - \delta_j^{k-3}| > |\delta_j^{k-3} - \delta_j^{k-4}| \tag{3.6}$$

As a safeguard against unnecessary removals, the original program required that Cutteridge's criteria should be satisfied on two separate entries into the GN section[14,16], termed indications to remove an element, before any element was to be removed. The modification to the two-part optimization program, giving repeated CG iterations prior to entering the GN , improved the efficacy of the algorithm and removed the necessity for dual indications. However, despite the repetition of results which generally occurred, two indications were required in the author's version of the program before any element was removed. (This was primarily to increase the probability of locating the correct minimum when difficulty was experienced at low error values - see section 4.6.)

The criteria had been developed primarily during the design of five node, eleven element networks. As the number of variables is increased, so the variation of each variable per iteration will tend to decrease. When applied to larger networks the criteria based on only five consecutive iterations proved insufficient i.e. element removals were indicated when all that was required was a significant reduction in the value of the elements. To overcome this, the conditions of equations 3.6 had to occur at five iterations, consecutive or otherwise, with all the relevant corrections of magnitude greater than one, before this was taken as an indication to remove an element. Also, if a series of GN iterations had ended because of slow progress, if the criteria of equations 3.6 had been satisfied on at least one iteration and if the correction of the relevant element had a negative value of magnitude greater than one, then this was also taken as an indication to remove an element. An element was also removed if the correction of an element was so large and negative in relation to the others, that a value of the linear search scalar greater than ten was obtained.

Problems did arise with isolated T-networks connected to the external nodes, as in Fig.3.6. With unsymmetrical networks, elements would be unnecessarily removed (a resistor in this case). Of the two similar elements connected to nodes one and two, one would have an increasingly positive correction and the other an increasingly negative correction. It is not possible to suppress element removals. If this is attempted then either the corrections will increase in size with no reduction of error, or an element of small value will remain with a very large correction which will overshadow the other values.

Minor problems also arose with the removal of elements which had already been reduced to a very small value. An extra algorithm to investigate the gradients (as described by di Mambro[21]) would be beneficial in these instances as a supplement to the original algorithms. It should be noted that the error will invariably increase as a result of the removal of a finite valued element. Also, pairs of elements were removed, when necessary, to retain the symmetry of a network. Multiple removals, as described by Cutteridge[15] and Hegazi[14], were not attempted.

3.9. Node Reduction

As has been stated earlier, the author predominantly employed a strategy of design by building from a small start network, rather than by reducing a large network. If at any stage the removal of a node had been indicated, the program restarted from the position of the last addition with the next best alternative. After most topological additions the error will be increased to a high value. It was noted that, on many occasions, node removals were required when

(i) the introduction of a new node had resulted in the removal of one of the existing nodes ,and produced an equivalent network (i.e. the minimum error values obtained were identical)

(ii) a local minimum had been obtained on the introduction of a new node (producing an increase in the error value), resulting in the removal of this new node after the addition of a further element.

Furthermore, certain start topologies had elements and nodes removed until the topologies were unable to produce the orders of the coefficients required. Consequently, on many occasions node removals failed to produce a superior approximation to the requirements and the author felt that a significant amount of time could be saved by ignoring these possibilities, without seriously impairing the design process. Examination of the solutions to problems appeared to indicate that, if a solution can be produced with n nodes, a very similar solution could be obtained with $n + 1$ nodes, etc. This finding further validates the adopted strategy. However, node removals were genuinely required on several instances and the facility should be included.

Node removals can be indicated in two manners. Firstly, elements can be removed repeatedly by the normal methods until a node has been effectively removed. For a node to be electrically meaningful there must be connected to it either, two or more types of elements to at least two other nodes, or, a minimum of three elements of the same type. (If only three elements of the same type are connected to a node it is preferable to perform a wye-delta transformation and remove a node in this way. This will facilitate the removal of any of these three elements, if required.)

Alternatively nodes can short out (i.e. all the elements connected to the node increase towards an infinite admittance). This was diagnosed by algorithms similar to those for indicating element removals except in this situation the GN corrections are becoming increasingly positive.

3.10. Criteria to Activate Element Addition

Ideally, elements are to be included only when it is known that the current topology is unable to realize the requirements, namely, when a global minimum of error has been located with the error above an acceptable value. The location of a minimum of the error is signified by a low value of the SSQGNC with no associated reduction in the error value. In the original program a minimum was defined as having a SSQGNC less than 10^{-7} with an error greater than 10^{-7} . With some minima it was not possible to achieve this value of the SSQGNC (although increasing the computer accuracy will decrease the attainable value), whilst on other occasions it was possible to significantly reduce error after this SSQGNC value had been attained. Furthermore, the possibility of locating a minimum of error, that was not a solution, with an error less than 10^{-7} was totally discounted. The author developed several criteria to determine when a minimum had been located.

If the SSQGNC was less than 10^{-10} and the error had been reduced by less than one per cent by the last iteration, then this was declared as a minimum. When minima occur at higher values of the SSQGNC it is necessary to define a minimum by the oscillating values of the SSQGNC with the error constant. A counter, ISS , was used to assess when these fluctuations did, in fact, signify a minimum. For the value of ISS to increase, the error must not have decreased by five per cent over the last five GN iterations. Otherwise, if the SSQGNC was less than 10^{-1} , but was greater than the previous value, and the error had not been reduced by more than one per cent by the last iteration, ISS was increased by one. At high errors the SSQGNC may fluctuate in a different manner. Hence, if the SSQGNC was greater than ten, but its last value was less than 10^{-1} , and if the error had not been reduced by more than one per cent by the last iteration, then ISS was increased by one. If ISS attained a value of three then this was defined as a minimum. Similarly, if the GN was halted because the error was not decreasing, but ISS had a value of greater than zero, then this was also declared as a minimum.

On occasions the optimization will be unable to locate a minimum. The original program performed a maximum of one hundred GN interations, at which stage a further element was included. With the version developed by the author, elements could only be included if the error, at such a cessation of the optimization, had been reduced below the value prior to the last addition. Algorithms were therefore included to gauge how the optimization was progressing, to ensure that reduced errors were obtained wherever possible.

A minimum of 60 consecutive GN iterations were performed unless the variables were dependent (signified by large GN corrections and repeated failure of the least squares algorithm). A further GN iteration was performed after this point, up to an overall maximum of 150 iterations, when ever

(i) the SSQGNC had been reduced on the last five iterations by an overall amount in excess of three per cent,

or (ii) if over the last five iterations the error had been reduced by more than one per cent and the SSQGNC by an amount greater than five per cent.

Then, finally, yet further GN iterations were performed, to an overall maximum of 200 , whilst ever the SSQGNC was less than one and the error had been reduced by the previous iteration.

Nodes were included only when there were no possible element additions. To ensure that the network remained stable, node additions were only attempted from positions declared as minima. The actual methods for introducing an element and a new node are described in Chapters 4 and 5 respectively.

3.11. Summary

The main theme of this chapter has been to discuss the modifications made to the original Algol program, follow through the development to the final Fortran version and, where appropriate, to discuss the factors which influenced the choice of strategy. The major modifications made by the author are as follows:

(1) algorithms have been introduced to increase the flexibility and efficiency of the program. The program will no longer attempt to optimize networks which cannot be equated to the required functions,

(2) options are now available which restrict the topological modifications, thus improving the efficiency of the design process. These concern symmetrical networks and what are, nominally, networks with compact poles at zero and/or infinity,

(3) the program can now design RLC networks as well as RC networks,

(4) a restart strategy has been incorporated which performs alternative additions when, with the original inclusions, the optimization had failed to locate an error value lower than that prior to the addition.

(5) the author has included algorithms to both determine the number of common factors required and to vary them whereas, originally, the number had been set to one or zero by the initial data and had remained at that value irrespective of the actual requirements,

(6) the efficacy of the two-part optimization program has been improved,

(7) the efficiency of the linear searches of the optimization routines have been improved, significantly improving the performance of the package,

(8) the criteria for the removal of elements have been improved,

(9) algorithms have been included to recognise when a node removal is required,

(10) the author improved the criteria which determine when topological additions are required,

(11) an alternative, improved method has been developed for the introduction of elements (detailed in Chapter 4),

(12) an effective method for node addition has been developed (Chapter 5) whereas previously the number of nodes had remained constant.

INDIVIDUAL ERROR FUNCTIONS	INITIAL VALUES OF NETWORK VARIABLES										
	10^3	10^2	10^1	5	3	1	0.3	0.5	10^{-1}	10^{-2}	10^{-3}
case 1	69	67	66	76	55	51	53	L	L	L	L
case 2	L	L	R	65	61	61	L	R	134	66	N
case 3	E	67	N	E	E	N	48	50	53	E	B

Jobs run in Algol on an ICL 1906A machine.

Case 1 - error formulaticn 1.

case 2 - error formulation 3 with common factor varied normally.

case 3 - error formulation 3 with common factor set to optimum value.

nnn - indicates time in seconds to locate the global minimum (value 1.12 x 10^{-8} with formulaticn 2).

L - indicates local, not global, minimum located.

B - indicates location of a local minimum with a common factor ($< 10^{-7}$) and elements at what appear to be undesirable values.

R - indicates that elements were removed.

N - indicates that there were no significant developments in 200 seconds.

E - indicates jobs terminated with error messages (usually associated with very small valued common factors, say, $< 10^{-6}$)

TABLE 3.1. RESULTS FOR THE NETWORK OF FIGURE 3.2 RUN FROM ARBITRARY STARTING VALUES WITH ALTERNATIVE INDIVIDUAL ERROR FUNCTIONS

INDIVIDUAL ERROR FUNCTIONS	INITIAL VALUES OF NETWORK VARIABLES										
	10^3	10^2	10^1	5	3	1	0.5	0.3	10^{-1}	10^{-2}	10^{-3}
case 1	4.97	2.08	2.74	3.25	3.66	2.21	2.73	1.22×10^{-2}	1.72×10^{-2}	2.13×10^{-2}	2.06×10^{-2}
case 2	8.63×10^{2}	4.93	1.87×10^{-1}	3.56×10^{-2}	6.77×10^{-3}	3.26×10^{-1}	7.34	5.04×10^{-1}	3.38×10^{-2}	1.97×10^{-2}	1.17×10^{-3}
case 3	3.16×10^{-2}	3.89×10^{-2}	2.24×10^{-3}	1.14×10^{-3}	5.31×10^{-4}	5.64×10^{-4}	7.02×10^{-3}	2.93×10^{-3}	7.44×10^{-3}	1.18×10^{-3}	7.77×10^{-2}

TABLE 3.2. VALUES OF THE COMMON FACTORS IN THE RUNS OF TABLE 3.1 AFTER FIVE CG ITERATIONS

NETWORK	MINIMA				
COMPONENTS	1 : 2.00×10^{-2}	2 : 7.55×10^{-3}	3 : 3.12×10^{-8}	4 : 1.37×10^{-2}	5 : 1.28×10^{-2}*
C_{13}	1.46×10^{-3}	1.47×10^{-3}	1.50×10^{-3}	1.49×10^{-3}	1.51×10^{-3}
C_{03}	5.80×10^{-1}	5.87×10^{-1}	5.97×10^{-1}	6.10×10^{-1}	6.01×10^{-1}
C_{05}	2.17×10^{-1}	4.01	4.54	1.07	8.15×10^{-2}
C_{04}	2.24×10^{1}	2.10×10^{1}	2.00×10^{1}	2.14×10^{1}	2.22×10^{1}
G_{14}	7.02×10^{-2}	7.00×10^{-2}	7.00×10^{-2}	7.05×10^{-2}	7.03×10^{-2}
G_{35}	7.02×10^{-2}	2.25×10^{-1}	5.34×10^{-1}	2.95×10^{-1}	8.11×10^{-1}
G_{45}	7.64×10^{-3}	5.88×10^{-1}	2.27	8.91	4.62×10^{-1}
G_{04}	1.03×10^{-1}	1.05×10^{-2}	1.02×10^{-1}	8.99×10^{-2}	8.94×10^{-2}
common factor	9.88×10^{-3}	1.39×10^{-2}	1.12	8.96	1.61×10^{1}

* numbers refer to the error value, using formulation 2, at the specific minima.

TABLE 3.3. VALUES FOR THE NETWORK OF FIGURE 3.2 AT THE VARIOUS LOCAL MINIMA OF ERROR

INDIVIDUAL ERROR FUNCTION	INITIAL VALUES OF NETWORK VARIABLES				
	10^{2}	10^{1}	10^{0}	10^{-1}	10^{-2}
case 1	254	238	263	326	R
case 2	R	R	287	R	N
case 3	300	321	221	R	N

Symbols as in Table 3.1 except N indicates no significant developments in 600 seconds.

TABLE 3.4. RESULTS FOR THE NETWORK OF FIGURE 3.3 RUN FROM ARBITRARY STARTING VALUES WITH ALTERNATIVE INDIVIDUAL ERROR FUNCTIONS

INDIVIDUAL ERROR FUNCTION	INDIVIDUAL VALUES OF NETWORK VARIABLES				
	10^2	10^1	10^0	10^{-1}	10^{-2}
case 1	1.84×10^1	5.75	4.31×10^{-1}	1.21×10^{-1}	1.14×10^{-1}
	2.00×10^1	6.33	5.90×10^{-1}	1.32×10^{-1}	1.24×10^{-1}
case 2	2.42×10^2	1.31×10^1	1.55	2.50×10^{-1}	1.79×10^{-2}
	2.63×10^2	1.36×10^1	1.66	2.84×10^{-1}	2.08×10^{-2}
case 3	1.07×10^2	2.11×10^1	5.00	2.55×10^1	1.31
	3.98×10^1	5.26	1.29	1.02×10^{-1}	1.65×10^{-1}

note second common factor set to value 1.1 times that given.

TABLE 3.5. VALUES OF THE COMMON FACTORS IN THE RUNS OF TABLE 3.4 AFTER FIVE CG ITERATIONS.

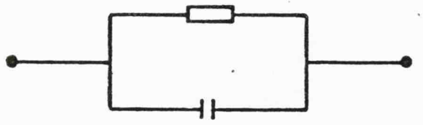


FIGURE 3.1: MODEL REPRESENTING A LOSSY CAPACITOR

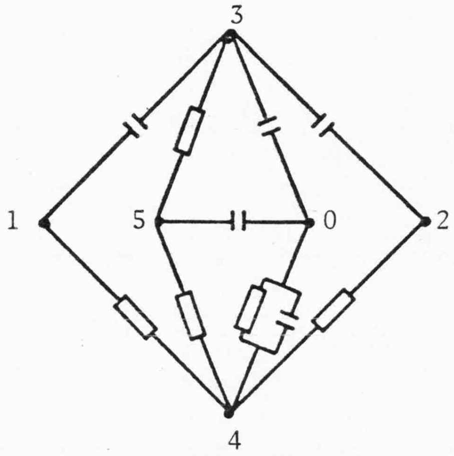
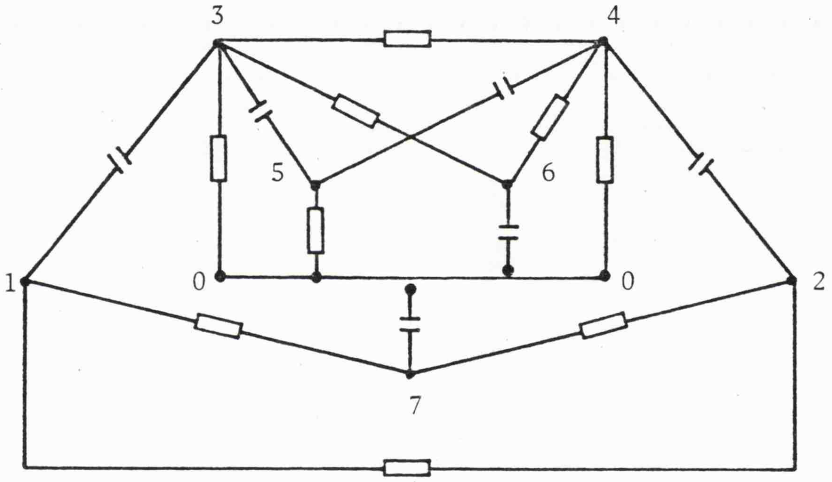


FIGURE 3.2: SYNTHESIZED NETWORK MATCHED TO FIALKOW'S FUNCTIONS



$C_{13} = 21.03$ $G_{36} = 0.6984$

$C_{24} = 5.032$ $G_{46} = .5044$

$C_{35} = 3.164$ $G_{05} = 7.739$

$C_{45} = 2.285$ $G_{03} = 50.66$

$C_{06} = 0.8468$ $G_{04} = 9.818$

$C_{07} = 0.5556$ $G_{17} = G_{27} = 0.2778$

$G_{34} = 1.290$ $G_{12} = 0.02778$

Common factors = 1.00 and 1.42

FIGURE 3.3: REALIZATION OF LUCAL'S FUNCTIONS

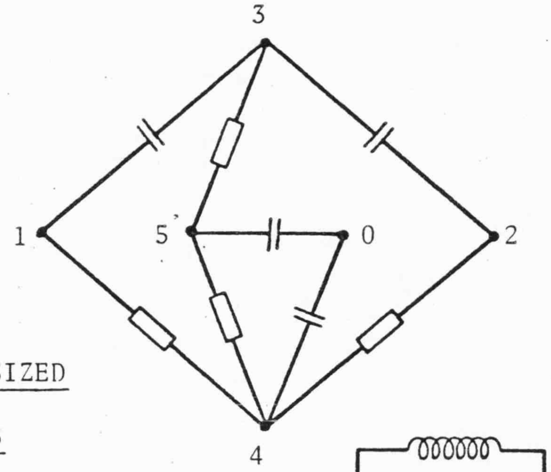


FIGURE 3.4: SYNTHESIZED NETWORK MATCHED TO FAILKOW'S FUNCTIONS

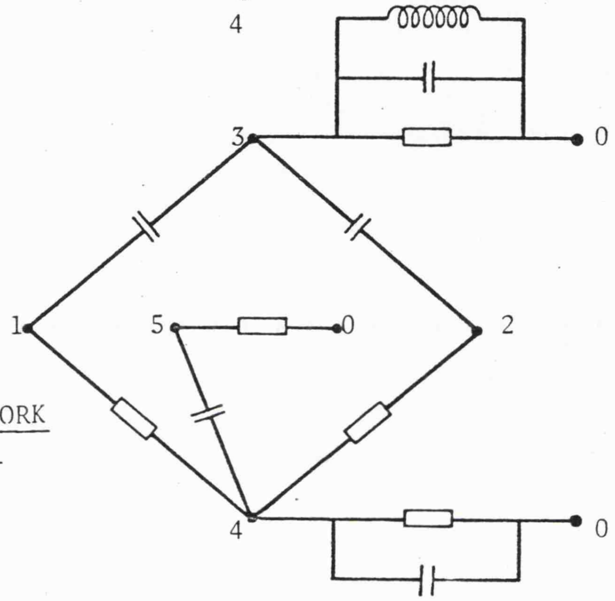


FIGURE 3.5: RLC NETWORK MATCHED TO FIALKOW'S FUNCTIONS

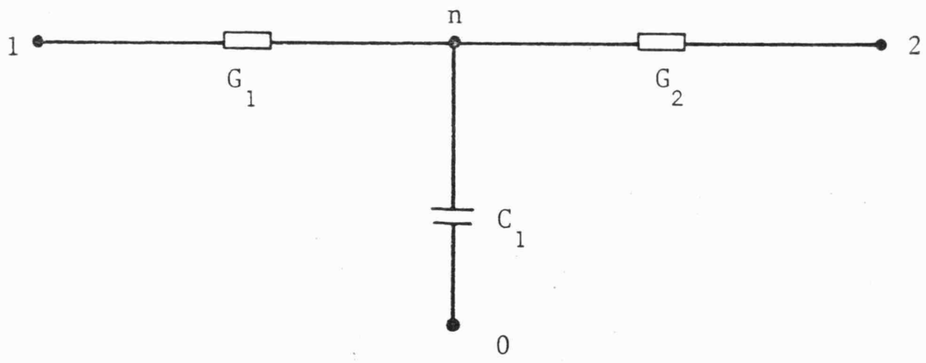


FIGURE 3.6: T-NETWORK CONNECTED TO EXTERNAL NODES

CHAPTER 4

ELEMENT ADDITION

4.1. Introduction

For a topology of the order investigated (i.e. containing up to ten nodes) the two-part optimization program as employed in this research is capable of locating the solution values, if one exists, from arbitrary starting values. However, it is unlikely that a solution topology will be chosen to start the design of a network not previously solved. The better approach at this stage, rather than repeatedly optimizing different networks, is to include extra elements when the best approximation for the synthesized network has been achieved, whilst retaining the original values of the existing variables.

Element addition, which is given priority over node addition, is activated by criteria described in section 3.10. These relate to situations where the rate of progress is slow as well as to the normal situation where a minimum of the error has been located.

The original program did contain a method for element addition (based on the optimum values for the introduction of a virtual element[33]) and its efficiency is discussed. This method had been used at Leicester with some success[14,16,17]. Two other possible methods for element addition are discussed, including a steepest descent method similar to that employed by Rohrer[48]. This method had been found to be of limited use when applied to direct matching of real frequency characteristics and pole-zero matching by Wright[12]. A more successful method based on the GN algorithm was devised by the author. This method is similar, in many ways, to a method devised by di Mambro[13,18] based upon the Levenburg algorithm.

As the components are represented by idealized linear models, the program can only be considered as an initial step in the ultimate goal of automated network design, producing results which are probably of greater interest to the network theoretician than the engineer. It was felt that, because of the idealized nature of the components, the techniques developed would only be fully justified if exact realizations, rather than quasi-solutions, were obtained. Furthermore, it seems reasonable to postulate that difficulties may arise when larger networks are considered. For these reasons, considerable effort was directed to the development of techniques to overcome the difficulties encountered with element additions, even after a good quasi solution had been obtained. Although several of these techniques did achieve a degree of success, none were infallible. However, later results indicated that these difficulties may not be so predominant as was thought at one time. Many successful runs were obtained, some of which are contained in Chapter 6.

4.2. Element Addition Based on the Optimum Values for Virtual Elements

It is possible, with several error functions, to derive expressions[33] for the value of a virtual element which will reduce the error by the maximum amount and for the corresponding error value. These are termed the optimum values (see Appendix 3). With the remaining functions a linear search has to be employed. Researchers at Leicester[14,16] had considered these optimum values to form the basis of the best methods for element addition. It was postulated that, if the optimum element value was positive, then the element would remain positive when all the values were optimized. This concept is similar to that for the steepest descent methods but is more powerful in that quantitative measures are produced, whereas the gradient of a zero valued element may not be representative of the situation as a whole[12]. This method was originally devised by Cutteridge[33] who suggested that the best virtual element to include was that which would reduce the error by the greatest amount on inclusion at its optimum value. This criteria was subsequently modified slightly by Krzeczkowski[16] and Hegazi[14].

The relative reduction in error produced by an optimum valued element addition will decrease as the error F is decreased. Similarly, the optimum element value will decrease to a value much smaller than that required at the next minimum of error (for typical values see Table 4.1). It is specified that the error F must be continually reduced and consequently, at low errors, the optimization can progress at only a pedestrian rate. In such cases the introduction of new elements at values above the optimum values (thus increasing the error initially) will substantially aid the optimization. However, as quantitative measures for the required values of the new elements cannot be obtained by this method, empirical methods have to be employed. (Such as setting the new element to a value equal to the mean value of the current elements[14].) Introduction of the new element at an underestimated value will produce little benefit, whilst introduction at an overestimated value can completely disrupt the network, often resulting in unnecessary element removals. It is probably as a result of the slow reduction in error experienced with elements introduced at low values that Krzeczkowski[16] based his element additions on the virtual element with the largest optimum value.

The efficacy of this method of element addition is reduced when an element has to be included from a position where the optimization has failed to locate a minimum. This may occur because of, say, a poor topology, or, localized difficulty with the optimization. As has been stated, at a minimum with a lower error value, say less than 10^{-2}, there is little difference between the error value at the minimum and that when the virtual element is included at the optimum value. Away from a minimum a large disparity appears in the values, influenced by factors which could be altered by the elements which are already present. Although an element which is able to provide a better approximation may be chosen, it is more likely that the new element will either be removed, or that the optimization will result in the unnecessary removal of other elements.

At even lower values of the error, the optimum values calculated become meaningless. (Indicated by negative values for the optimum error values.) The question arises, "At what stage do the values become meaningless?". Most importantly, "When will the optimum element values be of incorrect sign?". There are two questions to consider. Are the optimum values so small that, firstly, inaccuracies of the derivatives can alter the sign of the optimum element value and, secondly, are the minor variations of the existing elements from the exact values at the minimum of any consequence?

To summarize, the introduction of the element (at its optimum value) which produces the greatest reduction in the error works very effectively at high errors (greater than 10^{-2}). At errors below this value the elements which can improve the approximation can be determined. However, introduction of the element at its optimum value will cause the error to be reduced only very slowly, and a better value at which to introduce the element can only be estimated. Then, at an error above the best that the optimization can achieve, the optimum values become meaningless.

A further point to consider is that the variables may no longer be independent, even when the number is significantly lower than the number of coefficients. This is undetectable from the optimum values but is an important consideration because, although the CG algorithm will perform normally, the GN algorithm will be ineffectual.

Consider as an example the values of Table 4.1. The optimum values are positive for all the virtual elements except for the symmetrical pair of resistors R_{01} and R_{02}. As indicated, all the other virtual elements can be included to produce an improved approximation to the network. For this and several other topologies synthesizing the network described by Fialkow, it is possible to replace the resistor R_{04} by the elements R_{01} and R_{02}, and produce an equivalent network. As this is the global minimum, nothing can be gained from the addition of this pair of elements, thus validating the negative optimum values.

4.3. Element Addition by the Method of Steepest Descent

It should be noted that this method was only investigated because the results obtained using the optimum value method of section 4.2 initially appeared to be worse than they actually were. Many element additions signified by this method appeared to be incorrect but in fact failed merely because of the values at which they were introduced.

An alternative method, based on the gradients and similar to the steepest descent method of Rohrer[48], was developed. The steepest descent method had not been examined previously at Leicester in association with a coefficient matching technique, but had been examined by Wright[12]. Wright found that he could only add elements successfully to the simplest of networks.

Any new element should be included on the pretext that it should contribute something extra to the network that the original elements could not produce. In an attempt to delocalize the effects of the steepest descent method, the gradients of the individual error functions were considered. Originally, the virtual element included was that which produced the largest gradient of opposite sign for the individual error function with largest modulus. This algorithm greatly favoured elements connected to nodes 1 and 2 but did give useful additions.

Slight variations to this theme were also investigated but the results obtained were inferior to those obtained with the method using optimum values and would share similar limitations.

4.4. Element Addition Using the GN Algorithm

At low errors, the addition of elements at the optimum value to a structure optimized to the minimum of error can produce a large GN correction for the new element and a correspondingly slow rate of optimization. It was noted that on some occasions this large correction was positive in sign, whilst negative on others. Also, when elements had been included at larger values, say 10^{-3}, to speed the optimization and when even higher values had been required, a large positive correction had been reduced by a factor similar to the increase in the value of the element. Closer examination revealed that, for the linear elements examined in this thesis, the GN algorithm, which approximates to the first two terms in the Taylor Series, was sufficient to give approximate values for the new minimum with the virtual element included from an existing minimum. This provides an ideal platform on which to base a strategy for element addition. If the correction for the new element is positive, then the next minimum will occur with this new element at a positive value, hence, this element is a possible addition.

Although the optimization was performed in the logarithmic domain, the program still contained variables in the natural domain. The Jacobian and the corrections were calculated in the natural domain and the transformation to the logarithmic domain was carried out as follows. Firstly, each correction was divided by its corresponding element value. Then the elements were modified as below;

$$x_j^k = x_j^{k-1} * \exp(\lambda \delta_j^{k-1}) \quad \text{for all } j \tag{4.1}$$

where,

the superscript k denotes the values at iteration k ,

x denotes the value of a variable,

δ denotes the value of a transformed correction

and λ is a scalar which minimizes the error function (the linear search scalar).

The approximate value for the new element is therefore obtained by multiplying the modified correction by the corresponding element value, and adding this value to the original optimum value. It can be seen from Table 4.1 that these approximate values are generally of the correct order of magnitude. (In this case the approximations are correct to within a factor of two and are all underestimates.) Although these results were obtained for a small network with only twelve variables, this approximation holds for networks of twice this size and possibly larger still. Discrepancies only occurred between the two values when elements were introduced from local minima. Additions from such positions are generally productive, either successfully incorporating the new element or removing it to produce a better minimum. It can also be seen that the values for the new elements extend over a wide range (from 10^{-7} to 0.2), making empirical choices of the element values very difficult.

As the corrections are calculated initially in the natural domain, it is apparent that mapping to minima with negative values can occur. If a negative value was indicated for the virtual element, then this element was not considered for addition. However, there will be minima with a positive valued virtual element but with one of the other originally positive elements becoming negative. For the network of Fig.4.1, if the resistor R_{05} is included, an exact solution can be realized with a negative value for R_{04}. It is usual to find in these instances that there is a better minimum (than prior to the addition) with the negative element set to zero (i.e. removed). Hence, in general, when this situation arises it does not cause significant problems. With this example a minimum of value 1.84×10^{-8} is obtained from which several alternative realizations can be produced.

The optimum values were used to determine which of the possible elements to include. The element which produced the greatest reduction in error invariably proved the best addition at low errors but occasionally, at high errors, elements were included only to be removed after the subsequent addition.

To facilitate the programming (and because it is not possible to divide by zero to transfer to the logarithmic domain), the virtual elements were included, in turn, into the current topology at the initially low value of 10^{-6}. The network was analysed and the optimum values calculated. If the optimum element value was positive the element was set to this value. If the optimum value was negative, with magnitude less than 10^{-6}, then the element was reset to the absolute optimum value. At this stage the GN corrections were determined. Provisionally, a positive correction for the virtual element was stored for possible element addition and a negative correction stored for possible node addition (see Chapter 5). The error values used to determine the order of addition were those calculated at this stage and not those previously estimated (i.e. the optimum error value).

Occasionally, when a virtual element is introduced into a network the elements will no longer be independent. Such occurrences are characterized by large GN corrections. As large modifications may genuinely be required, criteria have to be developed to distinguish between the two situations. The element was rejected as a possible addition if any of the following conditions were violated:

(1) if three or more of the corrections for the original elements had values either

    (a) greater than 100
    or (b) less than 100

(2) if the approximate value for the new element was greater than that of any of the existing elements and if two or more of the corrections for the original elements had values either

    (a) greater than 100
    or (b) less than 100 .

Problems in discerning between the two situations are reduced as the computer accuracy is increased. However, when elements are to be included from a position which is not a minimum, the dependence of the variables is not so clearly indicated.

Elements were included at the optimum values unless their GN corrections had a value greater than ten, in which case the approximate value indicated by the corrections (termed the indicated value) was used.

This method of element addition is similar to an apparently successful method developed by Cutteridge and di Mambro[13,18]. Their method, however, was based upon the Levenberg algorithm for optimization. This can approximate to the direction of steepest descent and is thus able to consider many possible additions simultaneously. The corrections were repeatedly analysed as λ, the damping factor, was varied and the element giving the largest correction included. Multiple additions were thought to be possible but did not generally prove beneficial and so were avoided.

In comparison with the method developed by the author, this method would appear to be slightly inferior. First of all, the author was able to perform multiple additions (section 4.7). Also, when the variables become dependent the corrections increase in size. Hence, it would seem to be possible, especially with the large number of elements considered simultaneously, to choose an element of this type rather than the one which shows the greatest tendency to go positive. Furthermore, inspection of Table 4.1 shows that the element which exhibits the greatest tendency to go positive when the GN algorithm is invoked (i.e. the largest indicated value), is the resistor R_{05} which is, in fact, the only possible element addition which produces a local minimum rather than a solution. Using the author's method, this element would be the last element to be included.

One problem which could not be detected from the GN corrections was that on occasions (at high errors) new elements would immediately short out (as described in section 3.9).

Elements which necessitated the introduction of an extra common factor were handled in a slightly different manner. Initial investigations were conducted by employing a linear search to determine the optimum values for the common factor and the element. (Once the common factor has been assigned a value the optimum element value can be determined.) However, almost invariably (with the exceptions occurring when the error was particularly high) the search merely produced two very small values. The common factor and the element were therefore set to the arbitrary values of one and 10^{-6}, respectively. These arbitrary values increased the error value and, consequently, a positive GN correction in this situation indicates only the possibility of a minimum of error below this new high error value and not below the original minimum error value. Also, the introduction of the common factor means that the GN analysis is not performed from a true minimum, resulting in a further reduction in the efficiency of the method. However, the derivation of the GN corrections can be justified as an approximate value for the new element is obtained, thus facilitating the location of a better minimum if one does exist.

No element addition was accepted unless, on its inclusion, an error value below that prior to the addition of the new element was located. If not, the program restarted with an alternative addition.

4.5. The Influence of the Logarithmic Transformation and the Indicated Values on the Speed of the Optimization

Consider as a specific example the introduction of the three elements R_{12}, C_{45} and R_{03} to the network of Fig.4.1. The GN corrections calculated on the introduction of the elements at the optimum values are shown in Table 4.2. The modifications required to realize a solution increase in the order R_{12}, C_{45} and R_{03}.

Elements were introduced originally at the optimum value and the optimization was performed in the logarithmic domain. The elements were reintroduced firstly, with the optimization performed in the natural domain and, secondly, with the elements set to the indicated values. The results are given in Table 4.3.

The additions as performed in the original version of the program were obviously unacceptable. In all but the most trivial of examples, as with the addition of the resistor R_{12} , the logarithmic transformation was not a significant factor in the poor performance of the optimization algorithms and should be retained for its advantageous properties. An obvious improvement was obtained on introducing the elements at an approximation to its value at the subsequent minimum, (i.e. the indicated value). These results were typical of those obtained with all the networks examined. In general, the lower the error value prior to the element addition, the greater is the benefit obtained by including the element at the indicated value.

4.6. Two Examples Where Difficulty was Experienced with Element Addition

In the main, difficulties with the element addition appear to arise at low errors when the introduction of the new element does not produce a large reduction in the error. Two specific examples are given in Figs. 4.2 and 4.3. Both are synthesizing Lucal's functions (eqns.3.1). Figure 4.4 relates to an alternative initial path taken by the example of Fig.4.3.

With example 1 (Fig.4.2) no difficulty was attached to the location of a computer realization. (There are in fact three realizations readily obtained by the introduction of the elements R_{12}, C_{12} and C_{57}) The difficulties arose with the attempted introduction of other possible element additions. When included in the normal manner, at the indicated values, all the other possible elements were immediately reduced in value by several orders of magnitude. Whilst some were removed, others attempted to recover from these small values. The problem with this example appeared to arise because of the close configuration of the elements.

The major problem with example 2 (Figs.4.3 and 4.4) occurred on the addition of the capacitor C_{57}. The problem here is of removing the large element C_{05}.

Although similar problems occurred with other element additions, the subsequent discussion of possible methods to facilitate the introduction of elements is focused upon these two examples.

It should be noted that these faults did not apparently arise because of any inaccuracy of the analysis. Runs from identical start points but with different levels of computer accuracy differed slightly and, some, but not all, succeeded in locating the minimum of example 2, Fig.4.3d. However, the runs with the highest accuracy were not always the successful ones. Comparison of a large sample of runs showed that all accuracies were equally likely to proceed successfully, the accuracy of all being sufficient to locate the minimum when the GN proceeded along a path which led to a correct minimum. As a result of these findings, little research was undertaken into possible methods of increasing the accuracy of the analysis routines.

4.7. Multiple Element Addition

After an element addition has been determined by the methods of section 4.4, it is possible to determine a possible second addition by introducing the first element at its optimum value, and re-examining, in a similar way, all the remaining virtual elements. For two elements to be considered for simultaneous addition, both must have positive GN corrections and the independence of the variables must be retained. The author found that any two element additions determined in this manner would invariably be identical to any two consecutive, individual additions. In theory the process can be applied repeatedly and three elements have been simultaneously included to produce a solution (see section 6.7).

The process (examined with many examples) appears to be limited by element removals. Consider example 2, Fig.4.3. It is possible to add the two elements R_{16} and C_{57} to the network of Fig.4.3a apparently because no removals are required between the two additions when performed separately. When the simultaneous addition was performed the program had no difficulty in removing the element C_{05}. However, no second element could be found to simultaneously add with the element C_{05} to the network of Fig.4.3b. Similarly, when the alternative route of Fig.4.4 was taken, it was not possible to add a second element with the resistor R_{67} to the network of Fig.4.4a. Hence, it will not always be possible to manoeuvre around a position of difficulty.

A by-product of multiple additions is the possibility of evolving solutions more rapidly. At low errors it may prove substantially quicker to determine a second addition than to optimize a further topology requiring, say, 100 GN iterations. Alternatively, time may be wasted searching for a further addition which does not exist.

This method will not help to locate difficult minima of the same kind as in example 1. In fact, from the previous minimum to that shown in the figure (where the resistor R_{56} is included), the only possible double additions indicated are the three which produce computer realizations. This indicates one drawback with multiple additions. Many possible additions, which may lead to what are superior approximations when factors such as sensitivity are considered, are not signified.

Other disadvantages were noted. At higher errors (greater than 10^{-2}) elements were incorrectly signified as possible additions. On one occasion it appeared that four elements could be included whereas, in fact, all four were removed and the optimization located an alternative minimum with a reduced error value. On a further occasion a second element was indicated as a possible addition when the addition of the first alone was sufficient to produce a solution.

4.8. Detailed Use of the GN Corrections on Restarting After Element Addition

It was outlined in section 4.4 that, when a virtual element was included at the optimum value to an existing minimum of error, the GN corrections indicated the approximate values of the variables at the subsequent minimum. The effect of modifying the values of some, or all, of the existing variables to the approximate values indicated for the subsequent minimum (as opposed to leaving them at the values prior to the new element addition) was investigated. As the number of variables increases, so the accuracy of these approximate values decreases. With the two specific examples outlined previously, the GN corrections for the existing elements proved of little value.

When minima were located with the element additions of example 1 , the relative changes in the values of the existing variables were small. In fact, the CG section modified the values by an amount greater than the changes actually required, and consequently, the effect of any initial modifications to the values in relation to the GN corrections was immediately nullified.

The values of the variables at the minima for the topologies of Figures 4.3(b) and 4.3(d), and the values of the GN corrections with the subsequent element addition included at the optimum value, are given in Table 4.4. It can be seen that three significant changes occurred in the values between the two minima:

(i) the capacitor C_{05} was removed, i.e. set to zero,

(ii) the capacitor C_{07} was increased in value by a factor of twelve,

(iii) the resistor R_{17} was increased in value by a factor of thirty.

The element removal was not indicated by the initial GN analysis as the corresponding correction was of value minus 0.5 whereas, for a negative valued element the correction should have a value of less than minus unity. Similarly the large increases required in the other two elements were not apparent, possibly because the values were shifted by the removal of the element.

In an attempt to establish which elements would have to be removed subsequently, elements with relatively large, negative GN corrections (less than minus 0.25) when a GN analysis was performed with a virtual element included at the optimum value, were reduced in value and the GN corrections recalculated. (Would the GN corrections become increasingly negative if the removal was required?) However, once the elements constituting the original minimum had been altered, the position of the subsequent minimum, as suggested previously, was no longer indicated by the GN corrections. This failure to indicate the subsequent minimum had not occurred when the virtual element had been increased in value (sections 4.4 and 4.10).

4.9. Discussion of Alternative Values at Which to Add New Elements

Although the indicated value does provide a good approximate value for the new element, on occasions the new element will be driven to a low value and, either removed, or, the sneed of the optimization reduced to an unacceptable level. Slight modifications to the indicated value, say by a factor of 10 , made little difference. Only occasionally, when apparently by chance the optimization proceeded by a particular route, were the difficult minima located when a modification of this magnitude was made.

At the suggestion of Dr. Cutteridge, the element C_{57} of example 2 was introduced at a value 10^6 times the indicated value. This produced large GN corrections of opposite, oscillating signs (approximate magnitude 10^6) for the elements C_{57} and C_{05} . Gradually the element C_{05} was reduced in value and removed, thus producing the correct topology. However, the new element retained an inordinately large value giving a situation similar to additions at the optimum value. Namely, one large correction (negative for the element C_{57}) dominated and the optimization proceeded at an unacceptably slow rate.

With the particular example described above, the correct element had been removed. Further examples were investigated to establish if this treatment would repeatedly perform in this manner, i.e. to indicate which elements it is necessary to remove. However, the correct elements were not removed when an element removal was required and, furthermore, removals still occurred when none were necessary. This procedure was therefore of little value, generally.

4.10 The Behaviour of the GN Corrections When the Value of the Virtual Element is Increased

When a virtual element is included at its optimum value at a low valued minimum of error, a possible element addition is indicated by a corresponding large, positive GN correction. This large correction will dominate the optimization which consequently proceeds at a slow rate because the error must be continually reduced. If the value of the virtual element is increased by a factor of , say , ten , then the GN correction should decrease by the same factor if the same minimum is still being indicated.

Consider the addition of the capacitor C_{57} to the network of Figure 4.3b. If the optimum value of 4.73×10^{-9} is increased by factors of ten, the same minimum will be indicated until the value has been increased to 4.73 x -4 . At this stage the dominant GN correction has been reduced to 420 and the error increased to 2.11×10^{-5} from 6.53×10^{-8} . There would seem, therefore, to be much greater scope for larger modifications to the elements whilst the error is reducing. However, in practice, the improvement to the optimization is not sufficient to be of any consequence, at least in the logarithmic domain. When the program was modifed to operate in the natural domain, the scope had been increased sufficiently to allow a step length of λ equal to 0.25 to be taken. After this iteration, however, the variables were in an undefined region of space and the optimization converged onto the dominant minimum, reducing the new element to a negative value. (Again, the logarithmic transformation had not proved detrimental.) This example was typical of the results obtained whenever this procedure was invoked.

4.11 Element Addition with the Value Fixed Initially

At low errors the optimization is aided by including the new elements at the indicated value, thus increasing the initial error. Hence, it is possible for the optimization to reduce the value of the new element, possibly even remove it, and attempt to relocate the previous minimum. It is known that this indicated value will be a good approximation to the value of the element at the next minimum. It should therefore be possible to fix the new element at this value, optimize the other variables and force the values of the other variables into the region of vector space surrounding the required minimum.

Initially the constraints were released when the error had reached arbitrary values of 10^{-2} and 10^{-3}. These values proved to be too high, the new element being removed once the constraints were released. A more successful policy was to release the constraints on the value of the new element when the error had been reduced below a value one hundred times that of the previous minimum. Similarly, the constraints were released when a topological modification was indicated.

Four possible element additions to the network of Figure 4.2 were examined. These were the elements R_{36}, C_{16} , C_{14} and R_{34} , all of which had failed with the normal method of addition. Of these four, two (R_{36} and C_{16}) were successful. A third (C_{14}) was successful when the start values were modified as indicated by the original GN corrections. The fourth example persisted in removing the element R_{56} but, when this operation had been performed, the optimization was unable to locate a better minimum. (The difficulty may have been due to the series-parallel structure of the new network.)

The additions to the networks of Figures 4.3b and 4.3d were also performed by this method. Both of these examples could be regarded as successful as the optimization reduced the value of the elements which should have been removed to very small values. The fact that these elements were not actually removed was due to the limitations of the element removal algorithms, rather than a failure of the method.

Consider in more detail the addition of the element C_{57} to the network of Figure 4.3b. This requires the removal of the element C_{05}. i.e. with no constraints the minimum occurs with a negative value for the element C_{05} . When the element C_{57} was constrained to its indicated value, the minimum was shifted to a position where the value of the element C_{05} was small but positive. Similarly, it is not difficult to envisage a situation where an element which has a positive value at the true minimum, has a negative value at the minimum with the new element fixed at the indicated value. This could cause problems with the implementation of a method which was generally found to be successful. Slight variations to the indicated value will shift the position of the minimum and possibly shift any negative valued elements into a positive region. However, this could prove difficult (as with the aforementioned addition of the new element R_{34}) and it will possibly be quicker to by-pass an addition which causes such problems.

4.12 Simultaneous Addition and Removal of Elements

This approach is only relevant to the type of difficulty encountered with example 2. With this example difficulty is experienced with element removal on two occasions, namely,

(i) attempting to reach the network of Figure 4.4 c from that of Figure 4.2 b i.e. removing the resistor R_{27} (original value 0.205) after the addition of the resistor R_{67} (final value 0.548),

(ii) attempting to reach the network of Figure 4.3d from that of Figure 4.3c, i.e. removing the capacitor C_{05} (original value 1.21) after the addition of the capacitor C_{57} (final value 0.195).

In both cases the elements included and removed are of the same type, are connected to a common node and have similar values. This is possibly the root cause of the problem. Also, the second case requires the removal of an element of much larger value than the new element, possibly indicating why this removal is the more difficult to perform.

The following strategy was adopted. Any elements indicated as possible additions which on addition did not produce a lower error were stored. When all of the possible element additions had failed, the program reintroduced the elements whilst simultaneously removing, in turn, one of the several elements of the same type connected to a node also connected to the new element. This strategy proved successful with several examples which synthesized Lucal's functions (Section 6.5). This method failed, however, when the same start topologies were used to synthesize the similar network functions of equations 6.4, 6.6 and 6.7. However, it is possible that on these instances the GN was mapping to a minimum with a negative valued element which did not have a minimum with an error lower than that of the previous minimum when the negative valued elements were constrained to a value of zero.

4.13 Element Addition with the Existing Element Values Reset Arbitrarily

This method is also of greater significance when the difficulty arises from diagnosing necessary element removals. When the network of Figure 4.3c was restarted from values of unity, the element C_{05} was removed and the minimum located. However, completely arbitrary element values of this type will not always be ideal. (In this example the required values are all in the region 0.01 to 50 .) It is also possible that from such arbitrary values, the optimization will fail to distinguish between situations where an element genuinely requires to be removed or, merely reduced by an order of magnitude. (As with the T networks of Figure 3.6 mentioned in Section 3.8.)

An alternative to completely arbitrary values can be obtained by using the values from the previous minimum. This should have a similar effect to the successful multiple additions approach. When the element values of the network of Figure 4.3a were substituted for those of Figure 4.3b, the optimization again proceeded successfully. However, replacing the values of Figure 4.4c by those of Figure 4.4a failed to produce the desired effect. This was possibly because node 7 had just been included and so these three elements could not be reset to previous values.

4.14 An Alternative Optimization Algorithm

This algorithm was included as an alternative to the CG algorithm. Based on the method of steepest descent, it considers the second order derivatives also. The algorithm was devised by Cutteridge and Henderson[40,42] but an improved version, developed by Dowson[41], was actually used. The basis of the method was described in Section 2.7.

This method had proved to be considerably more powerful than a conventional, first order gradient descent method. Using this algorithm alone, the solution had been obtained, from many starting values, for a transistor modelling problem with eight equations and eight unknowns. This problem had previously proved too difficult for a two-part optimization program consisting of a gradient descent method followed by a Newton-Raphson algorithm[36].

Modification of a variable, λ , shifts the search direction from the steepest descent through to the steepest ascent direction. Discontinuities in the error produce several local minima for various values of λ . Usually, the algorithm proceeds from the minimum which has the lowest overall error value, restarting from alternative minima should the progress prove slow.

The results obtained with the lumped linear networks examined in this thesis were rather poor compared to those with the transistor problem. With the increase in the number of variables and the extra computation required to determine the values of the derivatives, each iteration required an inordinately long time to perform. An example of an RC network with 7 nodes and 17 variables performed only 7 complete iterations in 600 seconds on an ICL 1906A machine. Also, the algorithm did not show a tendency to move to the next minimum after the addition of a new element, as occurs with the GN algorithm, the effects being rather localized. Consecutive iterations were performed using the new algorithm on the usual situation, with the new element introduced at the indicated value, and the other variables at the values immediately prior to the addition. The net effect appeared to be that the initial elements were modified only slightly to compensate for the introduction of the new element. There was no apparent attempt to shift the variables to the vicinity of the new minimum.

The introduction of the element C_{57} to the network of Figure 4.3b was examined in detail. When individual iterations were followed by blocks of GN iterations, it was the removal of the new element C_{57} which was indicated and not that of the required element C_{05} . The values of the variables at the alternative minima (with the error reduced) were examined. On no occasion were the three large changes required for the next minimum reflected in the values. (i.e. removal of C_{05} and large increases in the elements C_{07} and R_{17})

4.15 Summary

The capabilities and limitations of the original method for element addition were discussed and the develonment of a superior method, utilising the GN corrections, described. This method not only determines whether the introduction of a particular element is feasible, but also indicates the approximate value of the new element at the new minimum. Hence, the major limitation of the original method of using the optimum element values was overcome.

Occasionally, the initial increase in the error resulted in the removal, or reduction to an unsatisfactory value, of the new element. Several methods to overcome such difficulties were briefly examined. The method which would probably prove effective with the greatest number of examples was optimizing with the new element initially fixed at its optimum value. However, difficulties can arise with this method, and an automated system to overcome all eventualities would probably involve an extra degree of difficulty than the other algorithms already in the program.

Although these difficulties occurred with sufficiently few examples to enable many realizations to successfully evolve, such occurrences will presumably increase as larger, more difficult problems are considered.

Virtual Element(s)	Optimum value	GN correction	Indicated value	Actual value
G_{01},G_{02}	-2.08×10^{-9}	-6.11×10^{6}	-	-
G_{03}	3.07×10^{-9}	2.55×10^{7}	7.85×10^{-2}	1.39×10^{-1}
G_{05}	9.55×10^{-10}	4.31×10^{8}	4.12×10^{-1}	1.26 *
G_{12}	2.16×10^{-12}	5.51×10^{4}	1.19×10^{-7}	2.50×10^{-7}
G_{13},G_{23}	3.82×10^{-10}	7.12×10^{4}	2.72×10^{-5}	5.85×10^{-5}
$G_{15}.G_{25}$	2.60×10^{-8}	3.16×10^{4}	8.22×10^{-4}	1.94×10^{-3}
G_{34}	5.87×10^{-8}	4.70×10^{5}	2.76×10^{-2}	5.38×10^{-2}
C_{01},C_{02}	1.45×10^{-10}	1.88×10^{6}	2.72×10^{-4}	4.39×10^{-4}
C_{12}	4.66×10^{-13}	3.83×10^{5}	1.79×10^{-7}	3.50×10^{-7}
C_{14},C_{24}	3.17×10^{-10}	5.05×10^{4}	1.60×10^{-5}	3.37×10^{-5}
C_{15},C_{25}	2.90×10^{-10}	4.76×10^{5}	1.38×10^{-4}	2.92×10^{-4}
C_{34}	9.90×10^{-8}	8.14×10^{4}	8.05×10^{-3}	1.74×10^{-2}
C_{35}	4.57×10^{-8}	1.21×10^{6}	5.52×10^{-2}	1.17×10^{-1}
C_{45}	5.20×10^{-7}	4.51×10^{5}	2.34×10^{-1}	5.57×10^{-1}

* This is the value at the quasi solution of error 1.86×10^{-8} and not that at the solution with the element R_{04} having a negative value.

TABLE 4.1. VALUES RELATED TO THE ADDITION OF ELEMENTS TO THE NETWORK OF FIGURE 4.1.

NEW ELEMENT	EXISTING VALUES								
	C_{13}, C_{23}	C_{03}	C_{04}	C_{05}	R_{14}, R_{24}	R_{04}	R_{35}	R_{45}	common factor
G_{12} (5.51×10^{4})*	-3.91×10^{-5}	-3.73×10^{-5}	1.10×10^{-3}	-4.60×10^{-3}	-4.30×10^{-6}	5.00×10^{-5}	3.55×10^{-2}	-1.03×10^{-2}	2.87×10^{-2}
C_{45} (4.51×10^{5})	-3.92×10^{-5}	-3.73×10^{-5}	-9.77×10^{-3}	4.33×10^{-2}	-8.70×10^{-7}	4.76×10^{-5}	-2.06×10^{-4}	7.52×10^{-2}	-2.19×10^{-2}
G_{03} (2.55×10^{7})	-3.93×10^{-5}	-3.74×10^{-5}	-5.44×10^{-4}	1.19×10^{-1}	-8.28×10^{-7}	-7.66×10^{-1}	-1.47×10^{-1}	-3.29×10^{-4}	-8.47×10^{-2}

* GN correction value.

TABLE 4.2. GN CORRECTIONS FOR THREE SPECIFIC ADDITIONS TO THE NETWORK OF FIGURE 4.1

NEW	METHOD 1		METHOD 2		METHOD 3	
ELEMENT	GN Iterations	Error	GN Iterations	Error	GN Iterations	Error
G_{12}	300	1.93×10^{-8}	11	S	29	S
C_{45}	300	2.97×10^{-8}	94	S	12	S
G_{03}	300	3.12×10^{-8}	150	2.18×10^{-8}	11	S

Method 1 - element included at optimum value, optimized in logarithmic domain.

Method 2 - element included at optimum value, optimized in natural domain.

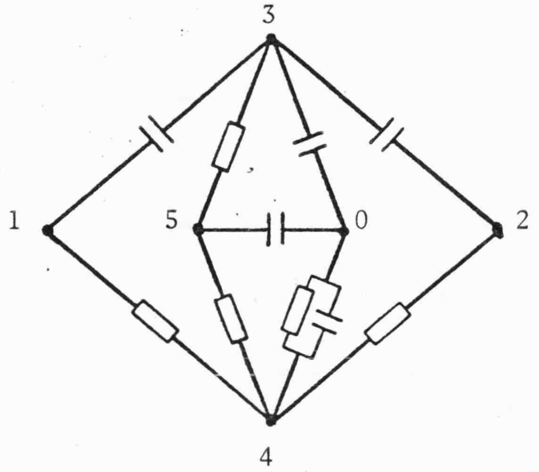
Method 3 - element included at indicated value, optimized in logarithmic domain.

S - indicates error reduced to limits of computer accuracy (approximately 10^{-16}).

TABLE 4.3. RESULTS ILLUSTRATING THE INFLUENCE OF THE LOGARITHMIC TRANSFORMATION AND THE INDICATED VALUES ON OPTIMIZATION SUBSEQUENT TO ELEMENT ADDITION.

	Minimum of Figure 4.3a			Minimum of Figure 4.3b			Minimum of Figure 4.3d
	Actual values	GN correction	Indicated values	Actual values	GN corrections	Indicated values	Actual values
C_{13}	21.1	2.72×10^{-2}		21.0	-9.34×10^{-3}		18.3
C_{34}	1.78	-6.10×10^{-2}		1.65	1.16×10^{-3}		1.66
C_{24}	2.57	8.46×10^{-2}		2.88	-8.51×10^{-4}		2.92
C_{05}	1.14	-3.85×10^{-1}		1.21	-5.22×10^{-1}		
C_{06}	1.96	-4.68×10^{-2}		1.72	5.27×10^{-2}		1.97
C_{56}	0.813	-3.28×10^{-1}		0.763	-9.20×10^{-2}		0.485
C_{07}	0.280	-3.83		0.216	-9.18×10^{-2}		2.72
C_{57}	4.61×10^{-6}	7.05×10^{4}	0.310	4.73×10^{-9}	4.20×10^{7}	0.198	0.263
G_{17}	0.518	-4.46		0.201	2.28		6.37
G_{67}	0.259	2.03		0.0617	0.732		0.0827
G_{26}	4.69	-0.103		3.92	3.46×10^{-3}		3.87
G_{03}	51.6	6.11×10^{-2}		52.0	1.49×10^{-3}		47.6
G_{04}	10.8	8.34×10^{-3}		11.2	-2.20×10^{-3}		11.0
G_{35}	2.48	-0.215		2.56	-1.85×10^{-1}		0.951
G_{16}	3.47×10^{-6}	7.04×10^{3}	0.0233	0.127	-4.08×10^{-1}		0.881
common	3.98	-8.47×10^{-2}		1.21	1.17		2.58
factors	2.38	-2.40×10^{-2}		2.35	2.47×10^{-3}		2.08

TABLE 4.4. VALUES AT MINIMA OF ERROR FOR NETWORKS OF FIGURE 4.3



$C_{13} = C_{23} = 1.50 \times 10^{-3}$

$C_{03} = 5.97 \times 10^{-1}$

$C_{05} = 4.54$

$C_{04} = 2.00 \times 10^{1}$

Common factor = 1.12

Error = 3.13×10^{-8}

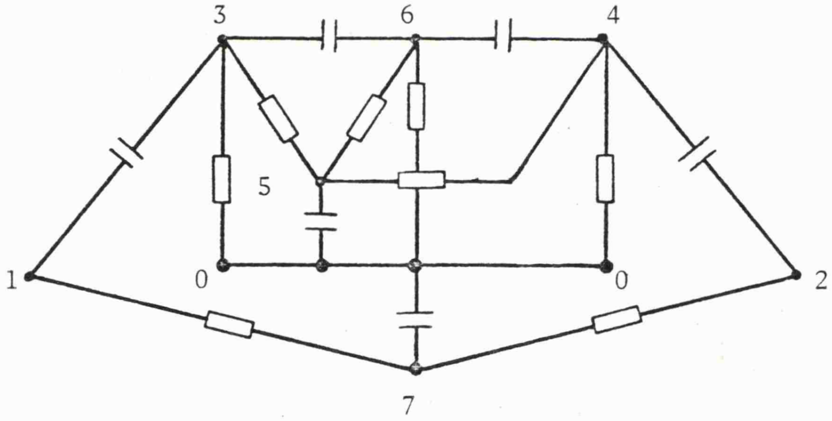
$G_{14} = G_{24} = 7.00 \times 10^{-2}$

$G_{35} = 5.34 \times 10^{-1}$

$G_{45} = 2.27$

$G_{04} = 1.02 \times 10^{-1}$

FIGURE 4.1: QUAZI REALIZATION OF FIALKOW'S FUNCTIONS



$C_{13} = 21.0$

$C_{36} = 3.79$

$C_{46} = 2.04$

$C_{24} = 5.09$

$C_{05} = 6.87$

$C_{07} = 0.568$

$G_{17} = 0.391$

$G_{27} = 0.291$

$G_{35} = 4.03$

$G_{45} = 3.22$

$G_{06} = 0.690$

$G_{04} = 9.41$

$G_{03} = 50.1$

$G_{56} = 1.86$

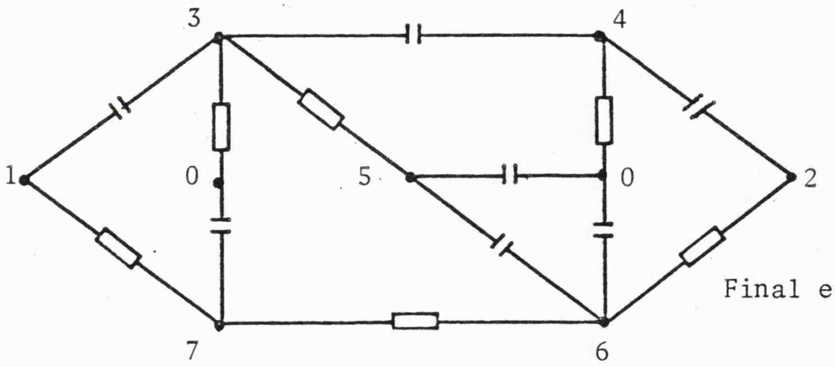
Common factors = 0.348, 1.18

Error = 8.80×10^{-7}

Possible RC element additions: C_{12}, C_{14}, C_{16}, C_{23}, C_{26}, C_{34}, C_{57}, G_{12}, G_{34}, G_{36}, G_{46}.

FIGURE 4.2: QUAZI REALIZATION OF LUCAL'S FUNCTIONS

(a)



Final error =
1.94 x 10^{-6}

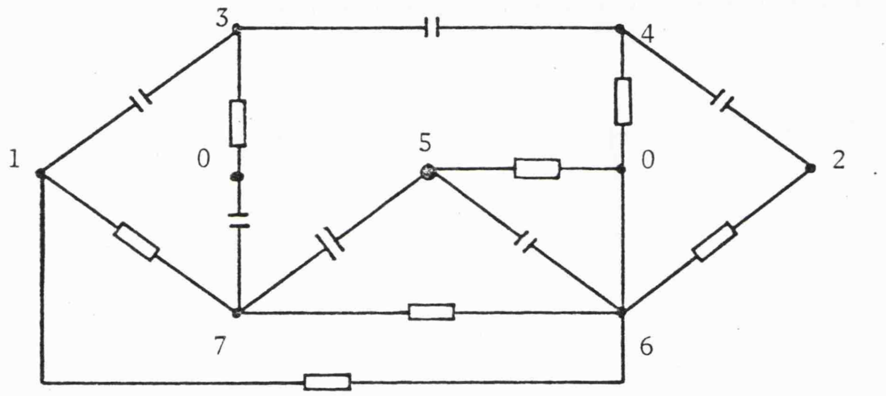
(b) ADD G_{16} Final error = 6.53 x 10^{-8}

(c) ADD C_{57}

(d) REMOVE C_{05} Final error = 9.64 x 10^{-9}

(e) ADD G_{05}

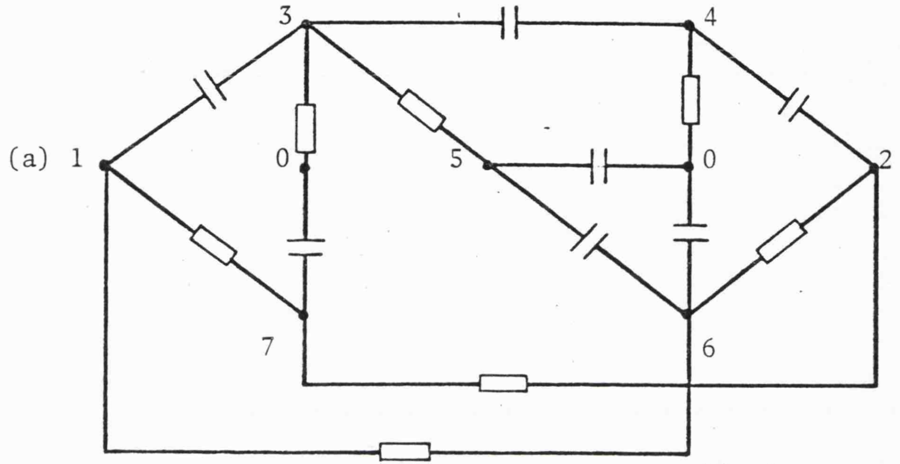
(f) REMOVE G_{35}



C_{13} = 14.0	G_{17} = 16.1
C_{34} = 1.68	G_{67} = 0.0927
C_{24} = 3.00	G_{26} = 3.81
C_{07} = 6.32	G_{03} = 39.2
C_{57} = 0.921	G_{04} = 10.8
C_{56} = 0.739	G_{05} = 2.10
C_{06} = 1.64	G_{16} = 0.0822

Common factors = 2.00 and 3.00

FIGURE 4.3: EVOLUTION OF A NETWORK REALIZING LUCAL'S FUNCTIONS



Final error = 6.92 x 10^{-8}

(b) ADD G_{67}

↓

(c) REMOVE G_{27} Final error = 6.53 x 10^{-8}

Network identical to that of Figure 4.3(b)

FIGURE 4.4: EVOLUTION OF A NETWORK REALIZING LUCAL'S FUNCTIONS

CHAPTER 5

NODE ADDITION

5.1. Introduction

It is improbable that the start topology chosen will prove capable of realizing the desired network response exactly and it is therefore necessary to develop methods for modifying the network topology. Criteria were developed to determine when topological modifications could best be made. These basically involved determining that either, the current network had attained the values which produce it's best approximation to the required network, or that the optimization was not proceeding quickly and efficiently. These criteria were detailed in section 3.10.

Later, an efficient method of element addition was developed (section 4.4). Element addition is to be preferred to node addition for two reasons. Firstly, it introduces fewer variables, which aids the optimization, and secondly, the nodal admittance matrix remains small, thus minimizing the computation required in the analysis. The introduction of elements confined within the existing topology may prove to be insufficient. The addition of new nodes may be required because, either, the network cannot be realized with the minimum number of nodes apparently possible, or, the start topology is far removed from a possible solution.

In this chapter, several possible methods for the addition of a node are reviewed and examined, some of which had been suggested previously and others devised by the author. As the research was restricted to idealized linear networks, the efficacy of any particular method was gauged very stringently. As exact realizations to the test problems were known to exist, it was postulated that a truly efficient method of node addition would produce an exact realization with, in most instances, the minimum number of nodes. Methods which produced a better approximation to the network but failed to provide realizations with the minimum number of nodes, were considered as inadequate.

The results obtained using the various methods are described in general and in relation to two specific examples. One of the methods devised by the author (section 5.8) provided realizations to these two examples and to many more, some of which are shown in Chapter 6.

5.2. Review of the Previous Methods for the Introduction of a New Node

It is perhaps worthwhile to reiterate that topological modifications are apparently accomplished more easily when a method of coefficient matching is employed. Using pole-zero and direct matching of frequency characteristics, Wright[12] was only able to successfully include elements to the simplest of networks and did not attempt the addition of nodes.

Di Mambro[13] suggested three possible methods for node addition. The first of these involved the substitution of equivalent networks as in Figure 5.1a. (A similar method to this is discussed in section 5.4.) Briefly, there are two main problems with this method. Firstly, how is the choice made as to which element to split? Furthermore, this is only a half-way stage. A further element has to be included to make this node electrically meaningful and this may prove difficult to achieve.

The equivalent circuit of Figure 5.1b also poses the question of which element is to be transformed. This transformation also necessitates the introduction of an inductor which may be undesirable because of the associated practical difficulties. One advantage of this method would appear to be that the elements will be included at approximately the required values, thus reducing the probability of problems with the optimization which arise when variables are introduced with extreme values.

The two further methods outlined by di Mambro involved the introduction of pairs of elements to the new node, but the criteria for determining which node to include were not ideal. The first was based on a steepest descent algorithm (which will tend to be very localized) and the second on an alternative error formulation which is not as efficient as that employed by the author. One of these methods was used by di Mambro in an example which succeeded in reducing the error by a factor of ten. However, the results were not conclusive as the run had to be curtailed due to a lack of research time.

Krzeczkowski[16] suggested two further methods which are discussed in sections 5.6 and 5.7. One of these methods, using a delta-wye transformation, was later adopted by Hegazi[14] who succeeded in producing a computer realization to one particular example. This was possibly the first occasion on which a node had been introduced entirely satisfactorily. Outside the group at Leicester, there seems to have been little achieved in terms of actual published results.

5.3. Two Specific Examples where the Introduction of a Further Node is Required

Using a design package similar to that at the commencement of this research, one particular attempt, by Cutteridge[17], to synthesize Fialkow's functions (eqns. 3.3), produced a quasi-realization with an error of 2.72×10^{-5}. This package used the same error formulation as the author and the element addition was based on the optimum values for virtual elements (section 4.2). The improved element addition technique developed by the author (section 4.4) introduced the element C_{03} to greater effect. After the removal of the capacitor C_{35}, an improved minimum of 4.99 x 10^{-6} was located (the network of Figure 5.2). At this stage there did not appear to be any further RC element additions, hence, this provides one example where node addition is required. (It is possible, however, to include an inductor at this stage and produce an RLC realization of the network without the introduction of a further node.) At this time realizations of this set of functions were only known to exist with five nodes and not with six as that the node addition would produce.

The second test example is similarly an RC network (synthesizing Lucal's functions) from which no further RC element additions are possible (Figure 5.3). Previously, the only solutions to this network had been produced by series-parallel decomposition[19, 30,31]. Similarly, computer realizations had only been produced by introducing no more than two alternative elements to these solutions. All these realizations had contained at least seven nodes, one more than the current network of this example. (A computer realization for this network with only six nodes is given in section 6.6.)

5.4. The Addition of a New Node by Replacing an Existing Element with an Equivalent Network

Di Mambro[13] outlined the possibility of introducing an extra node by splitting an existing element, as shown in Figure 5.1a. In theory, it should be possible to add further elements of the same type to the new node. In practice, the introduction of the node has little effect by itself and a third element of a different type must be introduced to the new node.

The author initially attempted to introduce this further element in a similar manner to the element addition method of section

4.4. Each existing element was split in turn and the GN corrections determined for each element addition made possible by the formation of the new node, with the new element and common factor set to arbitrary values of 10^{-6} and 1 , respectively. With the two specific examples described in section 5.3 (and with others), it proved impossible to locate an element which could in fact be included. A more intuitive approach was adopted after this.

The new approach was based on the assumption that, if the new network was very similar to the original network, then the starting error would be similar and could be decreased by the introduction of the extra variables. It was postulated that the network would be least disturbed if the smallest element was chosen for splitting as in Figure 5.1a. Similarly, least disturbance would result if the third element was connected to the reference node which is, in any case, the most general option and, possibly, most likely to produce the largest effect. The values were chosen such that they approximated to the introduction of a new common factor of value unity.

New nodes were introduced which did produce a reduction in the error but not always ideally. Initial results were encouraging with example 2 reaching a solution. (The route to the solution from this minimum is traced in Figure 4.4.) However, this proved to be an isolated success. Nodes tended to be repeatedly introduced into the same region, producing little reduction in the error. A typical example is shown in Figure 5.4. The method also failed to introduce a node to the network of example 1.

There are other difficulties associated with this method. Occasionally a new node would be introduced successfully only to be removed when the original element was re-introduced. Furthermore, as the GN algorithm was not involved in determining whether a node should be introduced, problems with the optimization did arise with particular examples.

It was explained in section 3.3 that certain networks (such as that defined by Lucal's functions) cannot be realized exactly when particular elements are included. It would seem unwise to include nodes in the positions of these small valued, undesirable elements if this, or a similar method, were to be used to introduce nodes.

5.5. Nodal Addition Employing a Modified Element Addition Algorithm

A method has been developed which is capable of introducing more than one element simultaneously, provided that, if the additions were performed individually, an element removal would not be required between additions. It is electrically meaningless for one element only to be connected to a node. It would seem, therefore, that if it were not for the extra common factor introduced, the introduction of two elements comprising a new node would be straightforward. (The introduction of one element alone could not possibly result in the removal of another element.)

As the optimum values for the three new variablestended to small values, the elements were set to small values of 10^{-6} and the common factor set to unity. The GN corrections were derived and, provided that criteria to establish the independence of the variables were not violated, the pairs of elements which had positive corrections were considered to be possible additions. One interesting feature was that the corrections for the elements forming the new node were identical in value. This feature perhaps calls into question the validity of the method as the minima do not occur with the elements at identical values. Priority of addition was given to the node which registered the lowest error value when included at the arbitrary values. The elements were introduced at their indicated values, namely, 10^{-6} times their GN corrections.

To summarize the results briefly, the method was successful in so far as it did introduce nodes which produced a reduction in the error, but not to the extent that computer realizations, not just quasi-realizations, were produced. A typical example of an attempted synthesis of Lucal's functions is given in Figure 5.5. The choice of which node to include was far from ideal and greatly favoured connections to nodes 1 and 2 . When the choice of nodes was determined by a user with knowledge of a solution, computer realizations could be produced (see Figure 5.6). As the error reduced, so the number of possible introductions decreased. In fact, it was not possible to introduce a further node to the networks of examples 1 and 2 . Furthermore, nodes indicated as possible additions were often removed, though it was not apparent whether this was simply difficulty with the optimization or if the method had failed.

The two new nodes shown in Figure 5.7 are identical, hence, if both are possible additions, the choice of which node to introduce is determined merely by computer round-off errors. However, it is quite possible that the introduction of one node will produce a realization whilst the other will not. If the new node was introduced in the incorrect position and no further additions occurred, it would be possible to reverse the node. However, it is normal to find that further additions will be possible.

To alleviate these difficulties, the effects of several modifications to the strategy were investigated:

(i) Renewed, but unsuccessful, attempts were made to calculate optimum values for the elements by optimizing these values only.

(ii) Priority was given to connections to the reference node (zero) via the internal nodes (not one or two). This was not always possible, particularly at lower errors, and so,

(iii) nodes were introduced, not to the optimum topology for the fixed number of nodes, but from the original topologies having the same number of nodes which had sparse numbers of elements and a higher error value.

(iv) Nodes were also included in groups of three elements rather than two, thus removing the ambiguity about reversals.

The results remained disappointing. It was still not possible to devise an automatic method capable of producing computer realizations.

One piece of information which did shed some light on which node to add was the optimum values for negative elements, as described in section 5.8. Such a method would prove difficult to employ automatically but would, perhaps, be of benefit to an interactive user. However, this would still not remove the difficulties caused by the absence of possible new nodes at low errors.

5.6. The Introduction of a Node by the Duplication of an Existing T-network

It was Krzeczkowski[16] who indicated that the networks of Figure 5.8 were equivalent and, without actually investigating its efficacy, suggested this as a possible method for introducing a node. Conceptually this method seems ideal in that it produces an equivalent network with extra variables, intimating that the error can only be reduced. Also, the new elements are introduced at reasonable values. The one apparent drawback is that there is no fundamental reason why the method should introduce nodes in the optimum position (i.e. so as to produce a solution with the minimum number of nodes).

In practice, the duplication often produces very large GN corrections which are unable to reduce the error, even when the maximum modification to each variable is limited. These large corrections also make it impossible to introduce further elements to by-pass this problem, as can sometimes occur. These large corrections remained when the values of one of the T-networks were altered so the two were no longer identical.

With example 1 the addition failed as described above. With example 2 there are two possible nodes to duplicate. When node 6 was duplicated the same problem appeared. When node 4 was duplicated two elements were removed, thus removing the new node.

5.7. The Delta-Wye Transformation as a Method of Introducing a Node

One obvious problem with this method was that often, when the introduction of a node was required, there was not a delta configuration to transform (see Figure 5.9). This would seem to suggest the necessity of a supplementary method, either of an alternative method of node addition or, of a restart facility to be used to locate alternative networks whose topologies do contain such a configuration. Both alternatives would appear to be far from ideal.

The transformation did not, on any occasion, enable an element of the same type as those forming the delta to be introduced to the new node. The introduction of an element of a different type, with an accompanying common factor, was required. Although Hegazi[14] used this method and succeeded in producing a realization, several points arise from his work. Only the one example was described, which gives little indication as to the general efficacy of the method. When the node was introduced, there were two possible delta configurations which could have been transformed. There was no indication as to why one had been preferred to the other. Furthermore, the choice of the accompanying element appeared to be arbitrary and its addition did in fact produce an increase in the error from a value of 1.74×10^{-4} to 2.12×10^{-2}.

The calculation of the optimum values for the new element and the common factor again proved to be of little benefit, as did the calculation of the GN corrections, except in formulating an approximate value for the new element. Consequently, the new element was introduced between the new and the reference nodes at a value of unity. The new common factor was also set to unity.

Generally, the results obtained were poor. On no occasion did the transformation produce a realization. When attempts were made to synthesize the network described by Lucal's functions, nodes were repeatedly introduced to the same regions of the network, producing little reduction in error.

The method failed with both test examples. Example 1 did not contain a delta configuration, whilst using the strategy outlined on example 2 merely resulted in the new node shorting out.

5.8. Node Introduction by Substituting a T-network in Place of a Negative Virtual Element

The order in which possible element additions are performed is governed by the error value obtained with the virtual element set to its optimum value (section 4.4). This method has proved to be almost totally successful in determining which element should be introduced, namely, that which will produce the greatest reduction in the error. Consequently, it seems reasonable to postulate that these optimum values will also give the optimum position for the introduction of a negative element. As one of the major defects with the previously discussed methods of node addition was that there seemed to be no apparent method of determining the optimum position for the introduction of the node, this information was thought to provide a firm basis on which to build an alternative method. (Nodes are only introduced when there are no available element additions, i.e. if any further elements were introduced they would attain negative values.)

The author chose to replace a prospective negative element by a T-network, as shown in Figure 5.10. This was chosen primarily on the basis of an inspection of the requirements to produce the known solutions, but also because of its simplicity and generality. It was thought that the four extra variables would provide a high probability of locating a better minimum. T-networks containing only two types of elements were chosen for two reasons. Firstly, it may be preferred that inductors are not included in the design. Secondly, the optimization algorithms often had difficulty in removing elements which had an associated common factor, hence, useful networks can be masked by superfluous additions.

Initially, the four new variables were included at values of unity, the mid value on the logarithmic scale. This enabled solutions to evolve for both test examples, thus producing the first six node realization to Fialkow's functions (Figure 6.24) and a seven node realization to Lucal's functions (Figure 4.3). Several attempts had previously been made, by the author and others, to produce such a six node realization by commencing from a topology with six nodes, but all had resulted in the removal of one node. It is possible to produce the solution topology originally given by Fialkow^3 from this realization by shorting nodes 4 and 5 and removing one of the pair of resistive

R_{04} elements produced. This would seem to confirm the theory that, if there is a solution to a problem with n nodes, there will be similar solutions with n + 1 nodes. This, in turn, validates the strategies of including a node when no element additions are possible and of ignoring any node removals indicated.

As could be expected, setting the new values arbitrarily did not always prove ideal. Examples arose where the value of unity was significantly larger, or smaller, than the required values. To reduce the occurrences of these faults, the values were chosen as shown in Figure 5.11, where the value G, for example, represents the modulus of the indicated value of the original negative element. The values are arranged to produce a new common factor with an approximate value of unity. Further problems arose when this indicated value was an underestimate, producing large GN corrections for the new elements and thus impairing the progress. Further empirical criteria were developed to combat this fault. If the initial error was below a value of five, the values of the elements constituting the new node were repeatedly multiplied by a factor of ten until the error had increased above this value.

This method was used successfully with many examples, producing many networks of a type which had not been realized before using any other methods of synthesis (see Chapter 6). However, the method did not succeed in introducing a node to the network from which Hegazi[14] successfully produced a realization. Furthermore, the procedures involved in introducing the new nodes were not ideal. Slight variations in the procedures can result in failure to locate a minimum following a feasible addition. This is perhaps one position from which extended optimization in the CG section, instead of switching to the GN when the error has reduced below 10^{-2}, would prove advantageous.

At one stage, extra additions were considered when the introduction of T-networks between the external nodes had failed. If a complete chain of elements (of the same type as the negative element) formed a link between nodes 1 and 2 , then new nodes were introduced, in turn, parallel to the elements in this chain. This innovation proved to be of little value with the examples considered.

5.9. Summary

Various methods have previously been suggested (and some examined) for introducing a new node to a network. Several of these methods, and others devised by the author, were examined. The one method[14] which had produced an exact realization, was found to be of limited value with the vast majority of examples. A method, based on the introduction of T-networks in place of negative valued elements, was devised by the author and used successfully on many occasions (section 5.8).

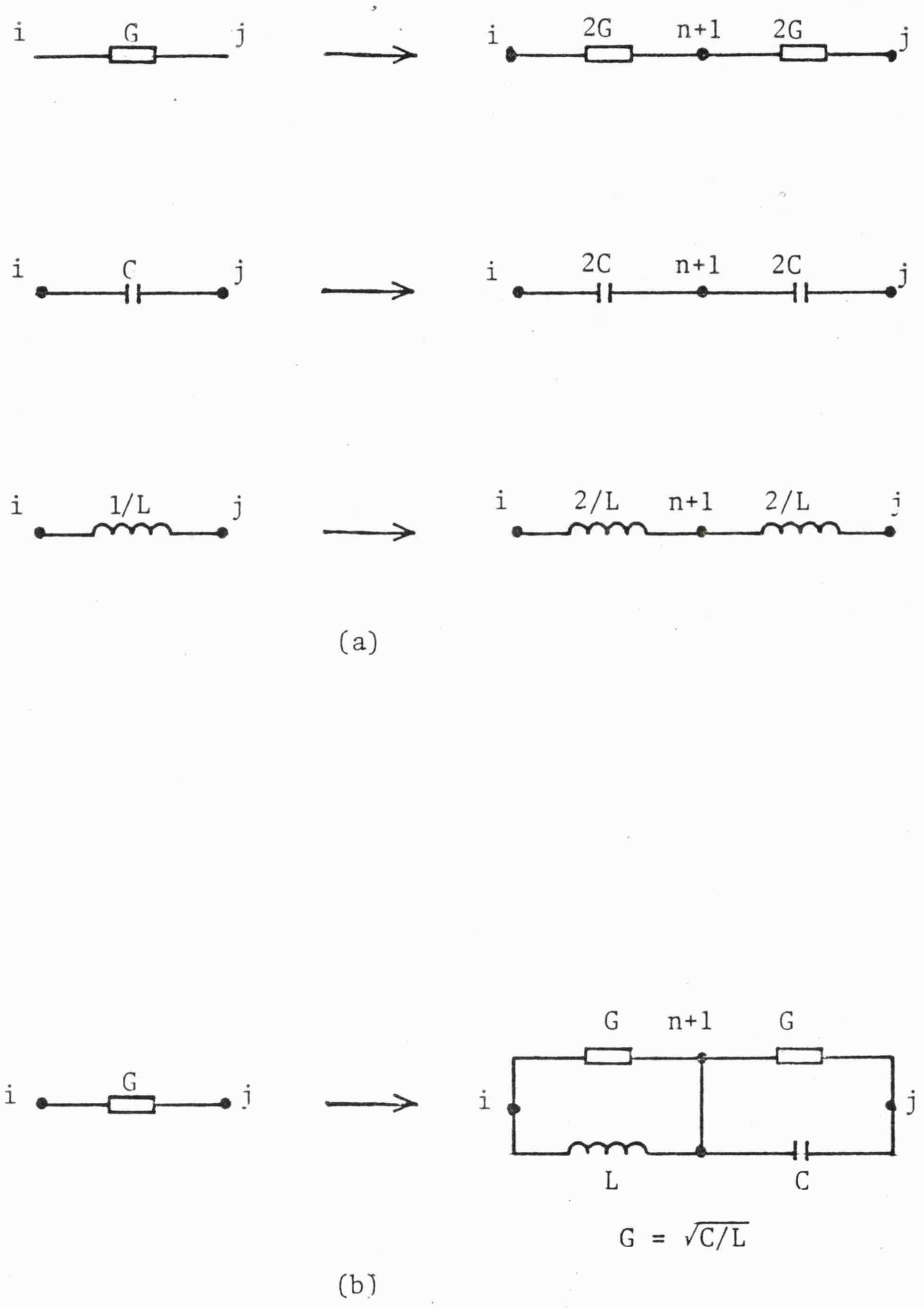
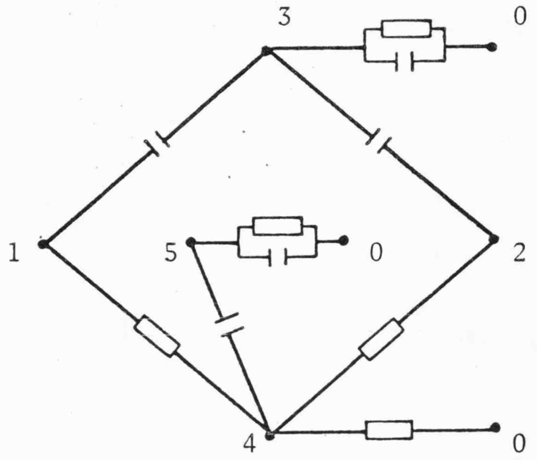


FIGURE 5.1 : NODE ADDITION BY SPLITTING ELEMENTS



$C_{13} = C_{23} = 1.50 \times 10^{-3}$

$C_{03} = 0.597$

$C_{45} = 25.1$

$C_{05} = 97.0$

Common factor = 0.888

Final error = 4.99×10^{-6}

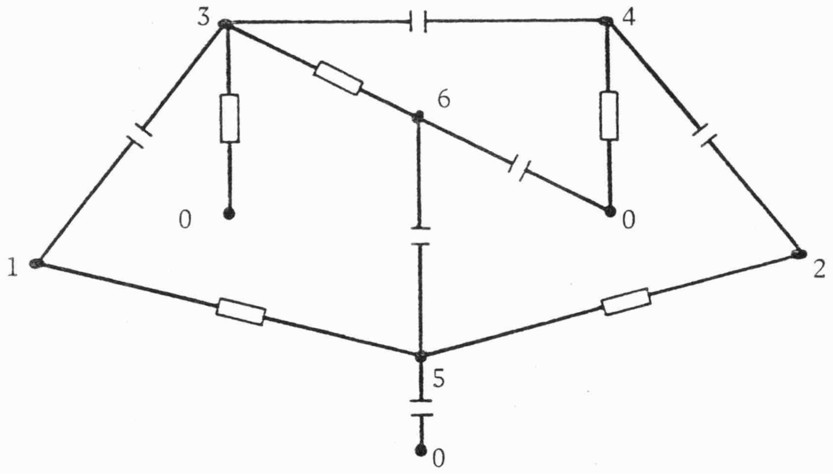
$G_{14} = G_{24} = 0.0700$

$G_{03} = 0.535$

$G_{05} = 48.1$

$G_{04} = 0.102$

FIGURE 5.2: RC NETWORK REQUIRING THE ADDITION OF A NODE TO REALIZE FIALKOW'S FUNCTIONS



$C_{13} = 21.3$

$C_{34} = 1.54$

$C_{24} = 3.38$

$C_{06} = 1.69$

$C_{56} = 0.756$

$C_{05} = 1.37$

Common factor = 2.50

Final error = 2.44×10^{-4}

$G_{15} = 0.180$

$G_{25} = 2.36$

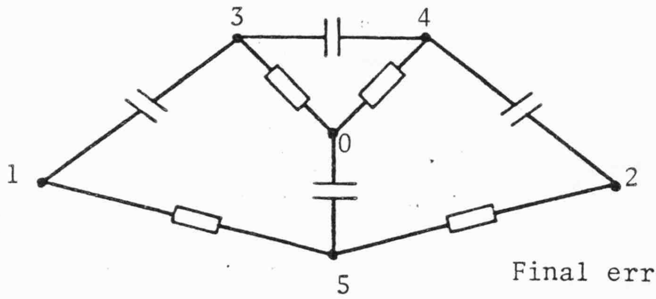
$G_{03} = 50.5$

$G_{04} = 12.7$

$G_{36} = 4.69$

FIGURE 5.3: RC NETWORK REQUIRING THE ADDITION OF A NODE TO REALIZE LUCAL'S FUNCTIONS

(a)

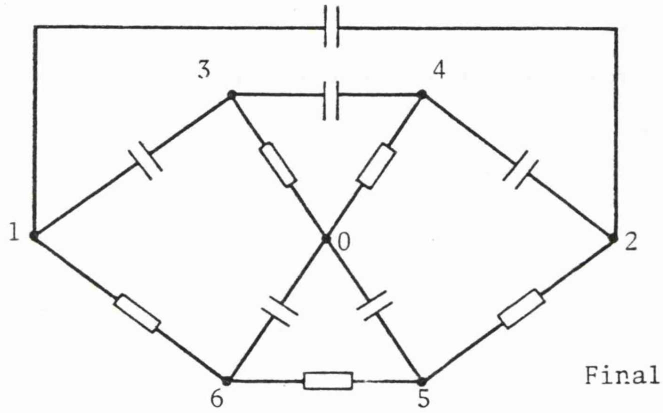


Final error = 0.819

(b) ADD C_{12}

Final error = 0.257

(c) ADD NODE 6



Final error = 4.48×10^{-2}

(d) ADD C_{56}

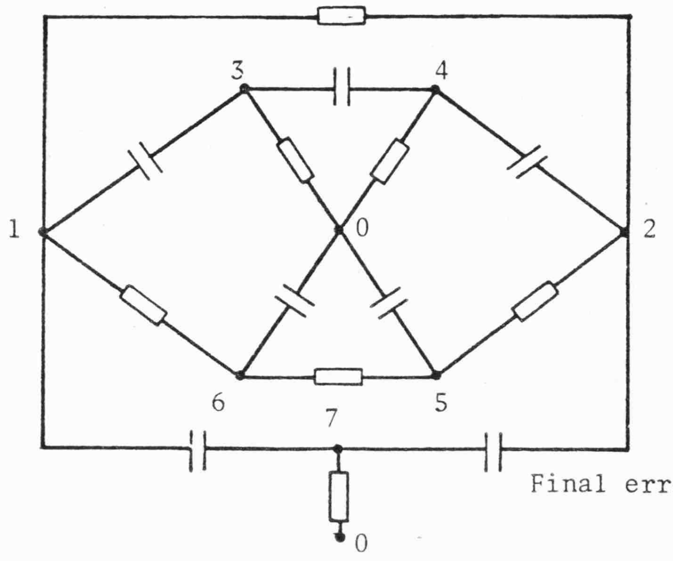
Final error = 3.97×0^{-2}

(e) ADD G_{12}

(f) REMOVE C_{56}

Final error = 3.25×10^{-2}

(g) ADD NODE 7



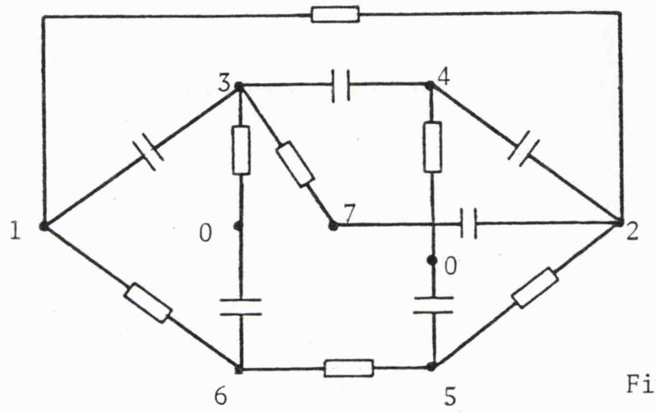
Final error* = 1.0×10^{-2}

FIGURE 5.4: (continued over)

(h) ADD G_{37}

(i) REMOVE G_{07}

(j) REMOVE C_{17}



Final error = 4.30×10^{-3}

$C_{13} = 18.7$ $G_{16} = 2.67$ $G_{37} = 0.455$

$C_{34} = 1.74$ $G_{56} = 7.99 \times 10^{-2}$

$C_{24} = 2.78$ $G_{25} = 2.65$

$C_{05} = 1.99$ $G_{03} = 51.8$

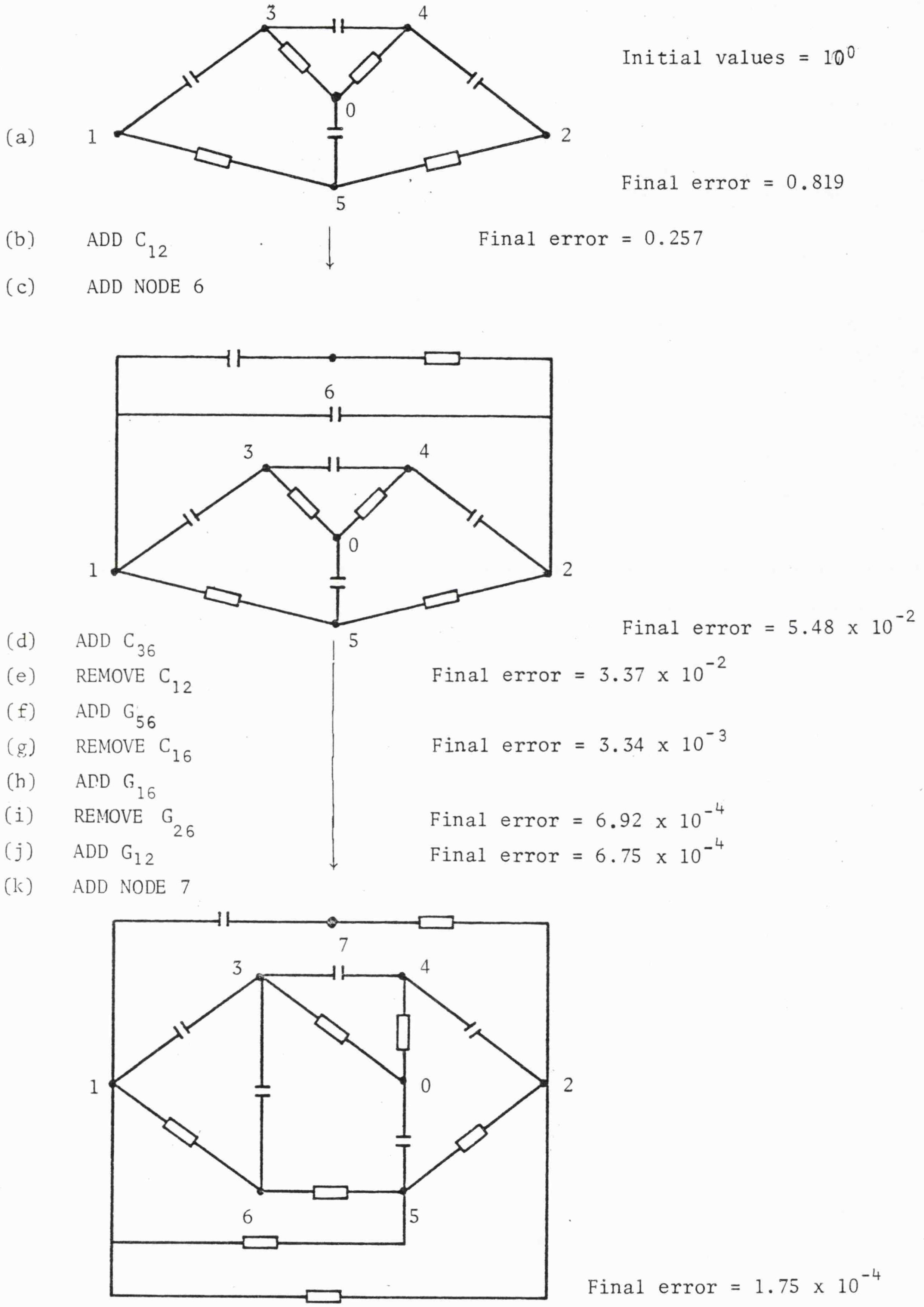
$C_{27} = 0.609$ $G_{04} = 13.6$

$C_{06} = 2.85$ $G_{12} = 9.20 \times 10^{-2}$

common factors 1.94 and 0.665

FIGURE 5.4: (CONTD.) EVOLUTION OF AN RC QUAZI REALIZATION OF LUCAL'S FUNCTIONS ADDING NODES BY THE METHOD OF SECTION 5.4

* indicates that a minimum of the overall error function was not located



(a) Initial values = 10^{0}

Final error = 0.819

(b) ADD C_{12} Final error = 0.257

(c) ADD NODE 6

Final error = 5.48×10^{-2}

(d) ADD C_{36}

(e) REMOVE C_{12} Final error = 3.37×10^{-2}

(f) ADD G_{56}

(g) REMOVE C_{16} Final error = 3.34×10^{-3}

(h) ADD G_{16}

(i) REMOVE G_{26} Final error = 6.92×10^{-4}

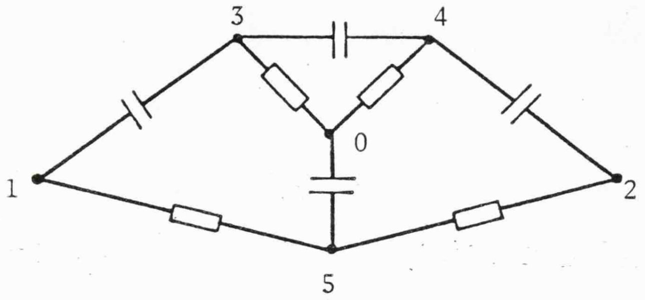
(j) ADD G_{12} Final error = 6.75×10^{-4}

(k) ADD NODE 7

Final error = 1.75×10^{-4}

FIGURE 5.5: EVOLUTION OF AN RC QUAZI REALIZATION OF LUCAL'S FUNCTIONS ADDING NODES BY THE METHOD OF SECTION 5.5

(a)



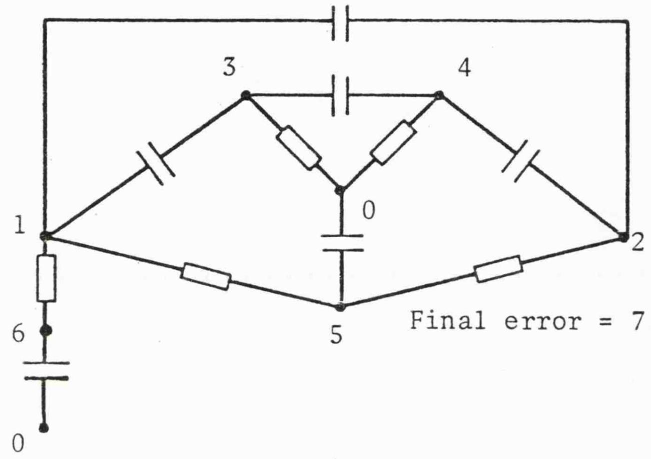
Initial values = 10^0

Initial error = 8.66

Final error = 0.819

(b) ADD C_{12} Final error = 0.257

(c) ADD NODE 6



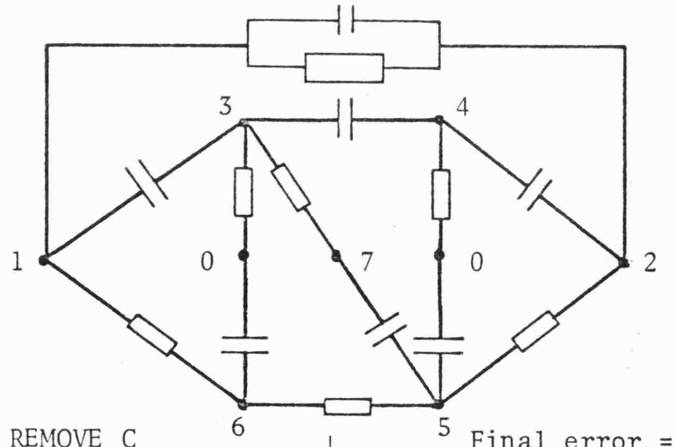
Final error = 7.04×10^{-2}

(d) ADD G_{56} Final error = 3.97×10^{-2}

(e) ADD G_{12}

(f) REMOVE G_{15} Final error = 3.25×10^{-2}

(g) ADD NODE 7



(h) REMOVE C_{12} Final error = 4.80×10^{-4}

(i) ADD C_{67} Final error* = 2.25×10^{-7}

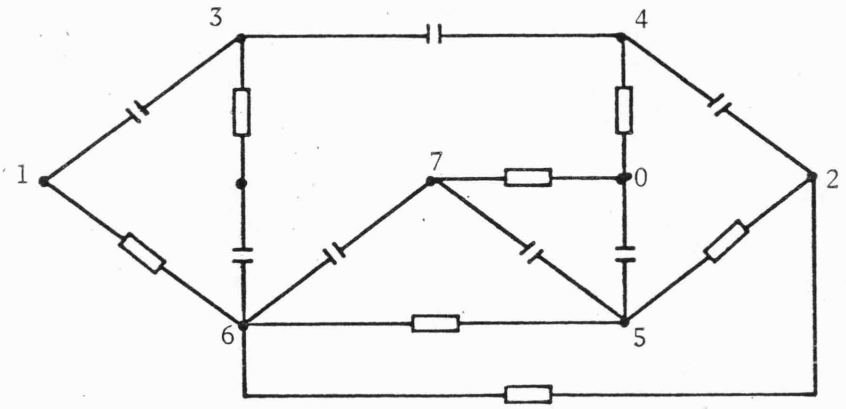
(j) ADD G_{26}

(k) REMOVE G_{12} Final error = 9.64×10^{-9}

(l) ADD G_{07}

FIGURE 5.6: REALIZATION OF LUCAL'S FUNCTIONS OBTAINED BY ADDING NODES IN THE MANNER OF SECTION 5.5 IN A POSITION DETERMINED BY THE AUTHOR (contd.)

(j) REMOVE G_{37} Final error = 3.50×10^{-15}



$C_{13} = 14.0$	$C_{06} = 6.38$	$G_{25} = 3.75$	$G_{56} = 0.112$
$C_{34} = 1.68$	$C_{57} = 0.722$	$G_{03} = 39.2$	$G_{26} = 5.96 \times 10^{-2}$
$C_{24} = 3.00$	$C_{67} = 0.937$	$G_{04} = 10.8$	$G_{07} = 2.10$
$C_{05} = 1.58$		$G_{16} = 16.2$	

Common factors = 2.00 and 3.00

FIGURE 5.6: CONTINUED

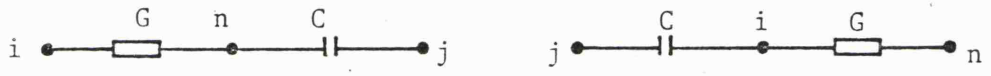


FIGURE 5.7: TWO EQUIVALENT NODE ADDITIONS

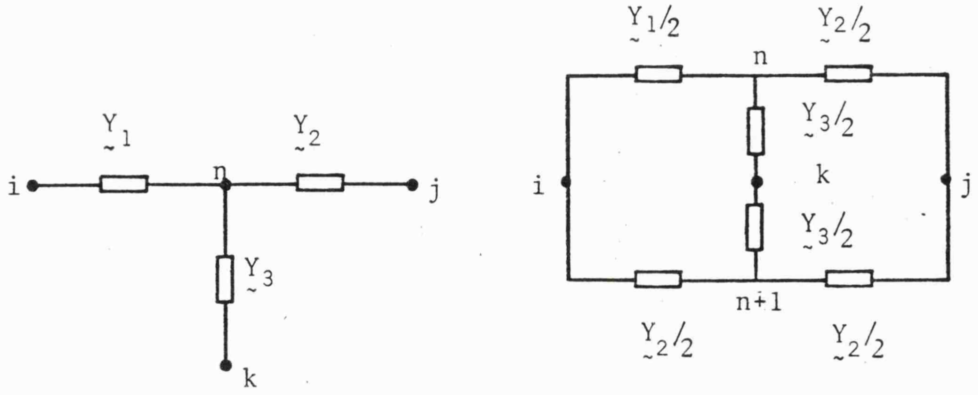
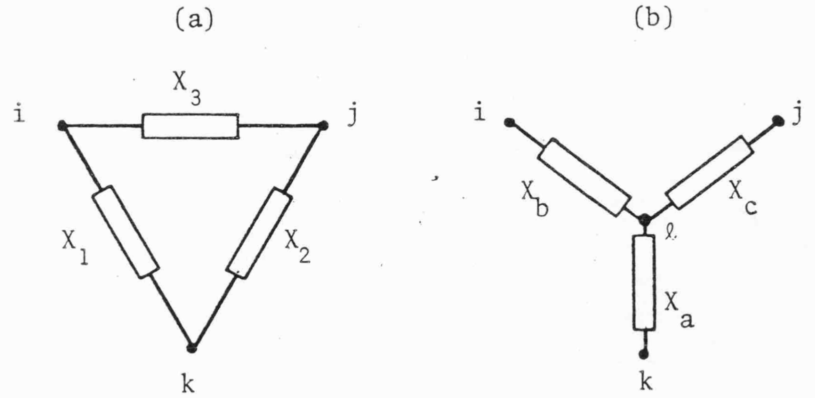


FIGURE 5.8: NODE ADDITION BY PARALLEL DUPLICATION



if $X^T = X_1 X_2 + X_2 X_3 + X_3 X_1$

then $X_a = X^T/X_3$, $X_b = X^T/X_1$ and $X_c = X^T/X_2$

FIGURE 5.9: DELTA-STAR TRANSFORMATION

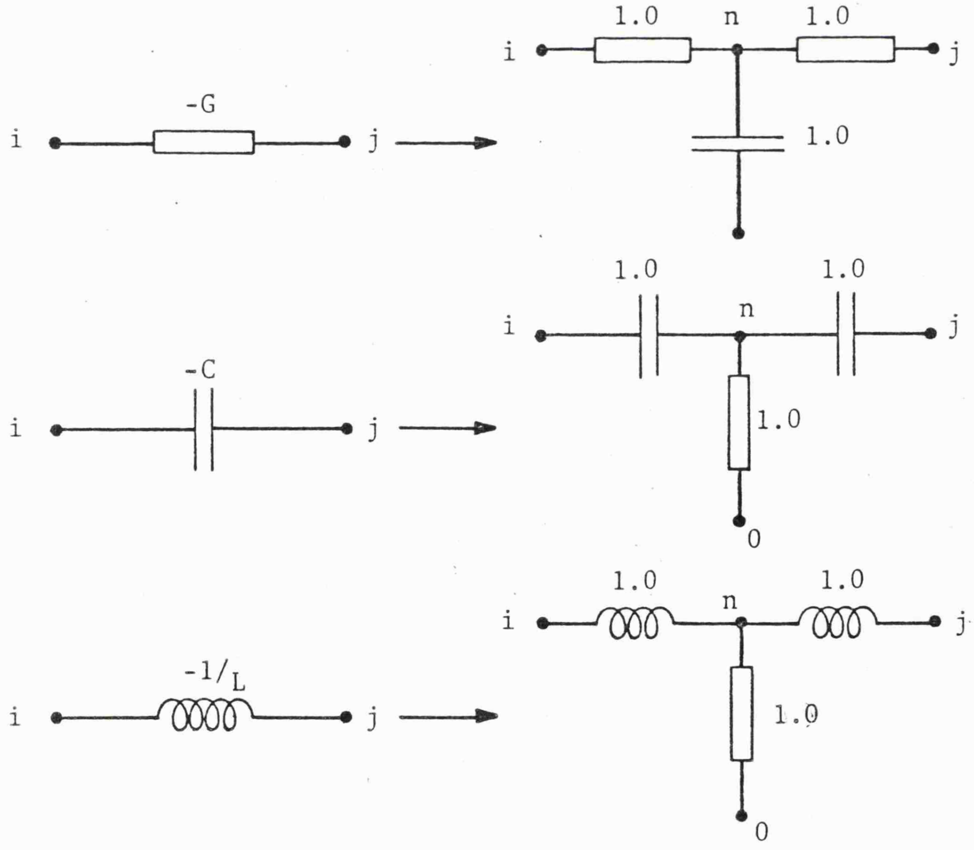


FIGURE 5.10: NODE ADDITION BY THE ORIGINAL METHOD OF SECTION 5.8

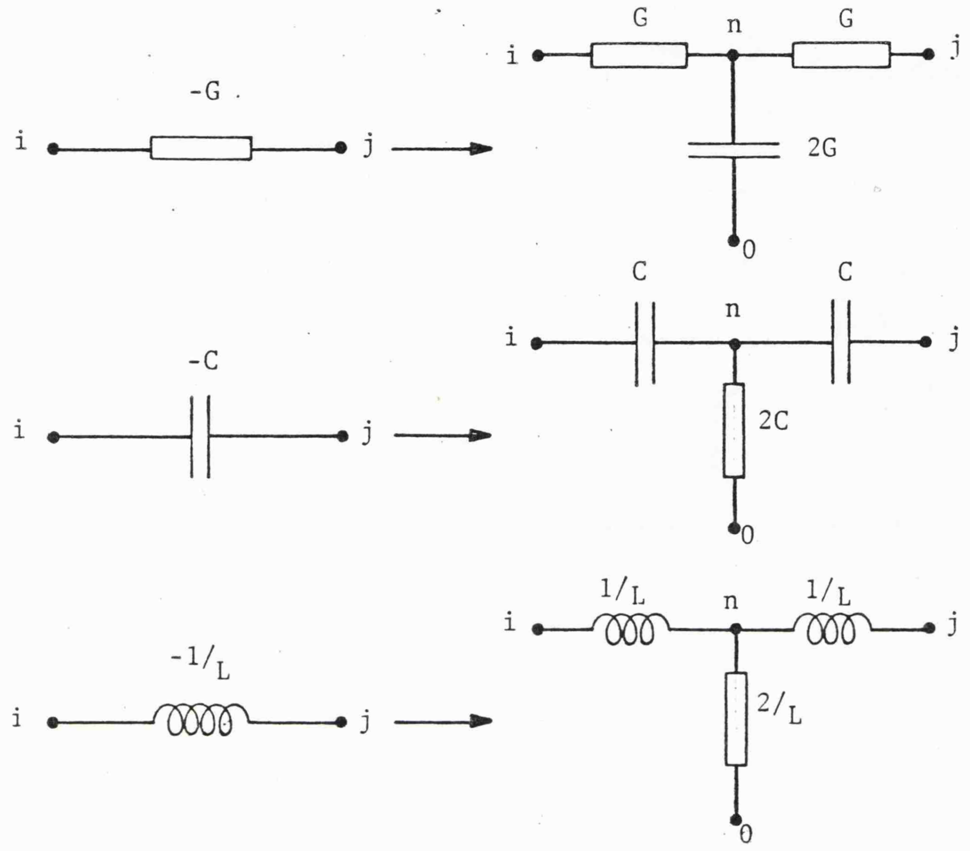


FIGURE 5.11: NODE ADDITION BY THE METHOD OF SECTION 5.8

CHAPTER 6

SELECTED RESULTS PRODUCED BY THE AUTOMATED DESIGN PACKAGE DEVELOPED BY THE AUTHOR

6.1. Introduction

The techniques which, when combined, form the automated design package, have been described in the previous chapters. Unless otherwise indicated, these results were obtained using the methods and strategies which were stated as being the most efficient. Designs with both RC and RLC networks were attempted.

To help clarify how a design evolves, an example is included which shows each topological modification of the design and describes any unsuccessful modifications (section 6.3). The space and time required precludes such a detailed description for other examples. Similarly, only an informative cross-section of all the results obtained could be included.

To facilitate an evaluation of the general efficacy of the package, classes of results are given when:

(i) the same start topology is used to synthesize several sets of slightly different network functions (section 6.4),

(ii) the synthesis of the same network (that defined by Lucal's functions) is attempted from various start topologies (sections 6.3 and 6.5).

Also, when appropriate, the results are compared to those obtained previously with similar methods applied by other researchers. With results from outside Leicester the design procedures will be totally different. Hence, it is only possible to compare the rudiments of what has been achieved, say, has node addition been successfully performed.

During the course of this research, several computer realizations were obtained which are of interest to the network theoretician, namely,

(i) a realization to Lucal's functions requiring only six nodes plus the reference node (section 6.6),

(ii) a series-parallel RLC realization of Fialkow's functions (section 6.8).

Several other networks of interest are also included.

The results produced by the automated design package are evaluated in Chapter 7. Also, difficulties with, and deficiencies of the program are outlined and future possibilities for research discussed.

6.2. The Network Functions Used as Test Examples

Lucal[19] published two RC realizations for the network described by the functions

$$y_{11} = \frac{\Delta_{22}}{\Delta_{1122}} = p + \frac{1}{6} + \frac{p}{p+1} + \frac{15p}{p+2} + \frac{35p}{p+3} .$$

$$-y_{12} = \frac{\Delta_{21}}{\Delta_{1122}} = p + \frac{1}{6} - \frac{p}{p+1} + \frac{15p/2}{p+2} - \frac{35p/3}{p+3}$$

$$y_{22} = \frac{\Delta_{11}}{\Delta_{1122}} = p + \frac{1}{6} + \frac{p}{p+1} + \frac{15p/4}{p+2} + \frac{35p/9}{p+3} \qquad (6.1)$$

The only realizations of these functions prior to this research had been produced by series-parallel decomposition of two element kind networks. The smaller of Lucal's realizations contained eight nodes (plus the reference) and three common factors of the form $(p + \alpha)$. Later, Hansen and Wanet[31] produced a realization with only seven nodes, thirteen elements and one common factor.

The functions can be written in the more general form

$$y_{11} = p + \frac{1}{6} + \frac{\alpha_{11}p}{p+1} + \frac{\beta_{11}p}{p+2} + \frac{\gamma_{11}p}{p+C}$$

$$-y_{12} = p + \frac{1}{6} + \frac{\alpha_{12}p}{p+1} + \frac{\beta_{12}p}{p+2} + \frac{\gamma_{12}p}{p+C}$$

$$y_{22} = p + \frac{1}{6} + \frac{\alpha_{22}p}{p+1} + \frac{\beta_{22}p}{p+2} + \frac{\gamma_{22}p}{p+C} \qquad (6.2)$$

where, $\alpha_{11} = 1$, $\alpha_{12} = -1$, $\alpha_{22} = 1$

$\beta_{11} = 15$, $\beta_{12} = 15/2$, $\beta_{22} = 15/4$

$\gamma_{11} = 35$, $\gamma_{12} = -35/3$, $\gamma_{22} = 35/9$

and $C = 3$ (6.3)

Eight similar sets of functions were obtained by modifying these parameters. (All examples retained the compactness of the poles.) The new values are as in equations 6.3 unless otherwise stated. The actual values of the coefficients are shown in Table 6.1. (Again, the values are not given if they are identical to those in Lucal's original set of functions.)

Example 1

$$\gamma_{11} = \gamma_{22} = 35/3 \qquad (6.4)$$

Example 2

$$\gamma_{11} = 35 \times 3 \ ; \ \gamma_{22} = 35/27 \qquad (6.5)$$

Example 3

$$\beta_{11} = \beta_{22} = 15/2 \qquad (6.6)$$

Example 4

$$\beta_{11} = 15 \times 2 \ ; \ \beta_{22} = 15/8 \qquad (6.7)$$

Example 5

$$\alpha_{11} = 1/2 \quad ; \quad \alpha_{22} = 2 \tag{6.8}$$

Example 6

$$\alpha_{11} = 2 \quad ; \quad \alpha_{12} = 1/2 \tag{6.9}$$

Example 7

$$C \;=\; 2.1$$

$$\beta_{12} = 15 \quad ; \quad \beta_{22} = 15 \tag{6.10}$$

note: if $B_{12} = 15/2$ then the cofactor Δ_{12} has two negative valued coefficients. This is avoided if β_{12} is set to a value of fifteen. However, without further modification this network would contravene Cauer's residue condition, namely,

$$\beta_{11}\,\beta_{22} - \beta_{12}^{2} \geq 0 \; .$$

Hence, if β_{22} is increased to a value of fifteen Cauer's condition will be satisfied and, furthermore, the poles will be compact.

Example 8

$$C = 10 \; . \tag{6.11}$$

If a computer realization had not already been obtained, runs from a minimal structure of five nodes were terminated prior to the addition of an eighth node. There usually proved to be insufficient time on the Cyber 72 to fully investigate the effects of such a third node addition. (The maximum job-time was 3500 octal seconds.)

For passive networks (as considered in this thesis), the coefficients of the polynomials produced must be positive for positive valued elements[56]. Consequently, at least one common factor of the form $(p + \alpha)$ is required to mask the effect of the negative coefficient of Fialkow's set of functions (equations 6.12),

$$y_{11} = y_{22} = \frac{1197p^3+56613.14p^2+28368.584p+191.184}{800000p^2+408000p+3840}$$

$$-y_{12} = \frac{3p^3-1.14p^2+197.176p+77.616}{800000p^2+408000p+3840} \qquad (6.12)$$

if they are to be realized by a passive network. $(0.38 < \alpha < \frac{197.176}{1.14})$ Furthermore, for a two element kind realization, the topology cannot be series-parallel[57].

Fialkow's functions can be written in the more general form

$$y_{11} = y_{22} = \frac{1}{800000}\left[\alpha_{11}p + \beta_{11} + \frac{Ap^2 + Bp}{p^2 + Cp + D}\right]$$

$$-y_{12} = \frac{1}{800000}\left[\alpha_{12}p + \beta_{12} - \frac{Ap^2 + Bp}{p^2 + Cp + D}\right] \qquad (6.13)$$

where,

$\alpha_{11} = 1197$; $\alpha_{12} = 3$

$\beta_{11} = 39839$; $\beta_{12} = 16,170$

$A = 16,172.67$; $B = 8,049.5384$

$C = 0.51$; $D = 0.0048$ (6.14)

The compact finite poles and the residues of these poles have irrational values[58].

Seven other sets of functions with a similar negative valued coefficient were obtained by modifying the values of the parameters in 6.14. The values of the new sets of parameters are as in Fialkow's original functions unless otherwise stated. The actual coefficient values are given in Table 6.2.

Example 1

$$\beta_{11} = 16170 \qquad (6.15)$$

This produces a further compact pole, that at zero.

Example 2

$$\alpha_{11} = 3 \qquad (6.16)$$

Produces a further compact pole, at infinity.

Example 3

$$\alpha_{11} = 3 \quad ; \quad \beta_{11} = 16,170 \qquad (6.17)$$

All four poles are now compact.

Example 4

$$\alpha_{11} = \alpha_{12} = 100$$
$$\beta_{11} = \beta_{12} = 16,121.17 \qquad (6.18)$$

All four poles are compact. This network was first suggested by Hegazi[14].

Example 5

$$\alpha_{12} = 783 \quad ; \quad \beta_{12} = 15770 \qquad (6.19)$$

These values produce two negative valued coefficients in the cofactor Δ_{12} .

Example 6

$$\alpha_{11} = 1977 \qquad \beta_{11} = 39430 \qquad (6.20)$$

Example 7

$$A = 16,172.167 \quad ; \quad B = 8,049.539$$
$$D = 0.005 \qquad (6.21)$$

6.3. A Detailed Study of the Evolution of a Network Which Realizes Lucal's Functions

This example (Figure 6.1) was chosen arbitrarily and not because it reflects particularly favourably on the techniques devised by the author. In fact, the evolution is a little strained in places. The author is of the opinion that a close study of such an example will provide a greater insight into the actual behaviour of the package. Several points worthy of consideration do arise.

(1) An examination of the topology at the realization (Figure 6.1x) indicates that five of the original eight elements have been retained. As two of the discarded elements were connected between the external nodes 1 and 2 , it is perhaps possible to say that five of the original six elements comprising the major features of the initial structure have been retained. This serves to illustrate the dependence on the original starting topology, even though this may appear to be dissimilar from the structure which evolves. It also underlines that realizations will not be produced with the minimum number of nodes from all start structures. Element additions are based on whether any one addition alone will provide an improved approximation, even when multiple additions are performed (as described in section 4.7). Variations are, therefore, essentially localized in character. However, it seems probable that the removal of nodes will allow for greater deviation from the start topology.

(2) The failure of the first element addition to the initial structure (Figure 6.1(a)) exemplifies the recognized drawback that, even when optimizing in the logarithmic domain, it is possible to map to minima with variables of negative value. The addition of the capacitor

C_{23} led to the removal of the elements C_{12} and R_{45} . Thus, the synthesized network coefficients could not be equated to all those required. With the initial structure at a minimum and the new element at the optimum value, the GN corrections were:

$$\begin{array}{ll} C_{13} = -0.374 & R_{25} = -8.58 \times 10^{-3} \\ C_{34} = 82.0 & R_{45} = 81.3 \\ C_{05} = -0.365 & R_{03} = 0.318 \\ C_{12} = -2.85 & R_{12} = 0.707 \\ C_{23} = 3.07 & \end{array} \qquad (6.22)$$

The correction for the element C_{12} is less than minus unity which, at this high error value, is probable evidence that mapping to a negative valued element has occurred. The correction for the resistor R_{45} was large and positive but the element was subsequently removed. The removal of the first element C_{12} (at which stage the value of the resistor R_{45} had increased by a factor of 60) produced a totally different minimum, thus necessitating the removal of a second element. (The variations of the values of elements are generally more marked at higher errors.) These large, genuinely required increases in the values indicate the difficulty in choosing criteria to discern between these situations and others where the variables are no longer independent.

This type of occurrence indicates the validity of the restart strategy. The introduction of the element C_{23} is subsequently performed.

(3) At one stage in the design the following sequence of events were performed by the package:

(a) the element C_{23} was included

(b) the element C_{12} was removed

(c) the element C_{24} was included

(d) the element C_{23} was removed

(e) the element R_{15} was included (3.4)

It would appear that the initial addition of the element C_{23} was superfluous and that the addition of the element C_{24} should have been preferred, particularly as the introduction of the removed element C_{12} was not possible at stage (e) . The location of the actual minima, with the capacitor C_{12} having negative values, confirmed that the original choice of the element C_{23} had been incorrect. The introduction of the element C_{23} realized a minimum of error 0.545 as compared to a minimum of 0.276 .

The values of the GN corrections when each element was introduced, in turn, at the optimum value were 5.26 for the element C_{23} and 1.81×10^3 for the element C_{24} . These values are typical of those which occur when an incorrect choice of element has been made, i.e. the element with the larger error on introduction at the optimum value, has greater scope for modification (signified by a larger GN correction) and produces a minimum with lower error than an element that is included at an optimum value which is high when compared to its value at the subsequent minimum. The determination of empirical criteria to extrapolate to an approximate, eventual error were not considered to be necessary. This feature occurs only rarely, always at relatively high error values, is easily rectified, and at lower errors in the vicinity of solutions the optimum element is invariably included.

A similar situation can occur when nodes are introduced. An investigation of such occurrences seemed to indicate that this would produce a solution with, say, one extra node and common factor. As alternative solutions are of interest for sensitivity considerations, etc.

this is not a totally unacceptable feature. Furthermore, an examination of several examples indicated that any surplus nodes would often be removed if such a facility was available.

(4) It was when the second node addition was required that the evolution became a little strained. The first attempt to introduce this second node was to substitute for the negative element R_{36} (section 5.8). Relatively large GN corrections were obtained after this introduction (SSQGNC of the order of 10^3). The reason for this was not evident. Possibly the elements were dependent or, perhaps, the values were far removed from those at a minimum and in a non-linear region of vector space. Although no minimum was located, the error was reduced below that prior to the node addition. The subsequent analysis did not locate any possible element additions. Nodes were not introduced from such positions, hence, the program restarted by including an alternative choice to form the seventh node. This also resulted in large GN corrections (SSQGNC of the order 10^5) because the new node was identical in structure to that of node 6 (Fig.6.1p). The first element addition (C_{27}) resulted in the removal of the elements R_{12} and C_{12} and the element C_{46} shorted out. The second addition (C_{26}) proceeded in a similar manner but an indication to remove the element C_{46} occurred simultaneously with an indication to short out an element. Of these two possibilities, element removal was preferred. The removal of this element produced an increase in the error (Figs. 6.1(s) to 6.1(t)) due to the location of an alternative minimum where the removal of a node was not required. The fact that the element C_{46} can be included from this minimum confirms that this is the case. (The element is re-introduced at a later stage and is present in the final topology.) Such an increase in error is acceptable because the error is only compared between adjacent element additions.

6.4. The Attempted Synthesis of Slightly Differing Network Functions from Those Given by Lucal Using the Same Starting Topology

The start network used was that described in the previous section (Figure 6.1(a)). Further network functions were obtained by modifying the values of the residues of one of the finite poles of Lucal's functions (functions 6.4 to 6.9). The two examples with modified pole values (functions 6.10 and 6.11) do not appear to have similar solutions and so cannot be meaningfully compared.

In the attempted synthesis of functions 6.5, the element C_{34} was removed, causing the job to terminate. Of the remaining six examples tested (Lucal's functions plus five modified sets of functions), five produced a realization. The other produced a good quasi solution with seven nodes (error 1.09×10^{-8}). The resulting topologies are shown in Figures 6.2 to 6.6.

The examples were very similar, in terms of topological modifications, until the addition of the second node (Figure 6.1(p)). The only differences were

(i) for functions 6.7 the element C_{34} was included in place of the element C_{12} ;

(ii) for functions 6.8 the element R_{12} was not present.

From this point onwards, all the successful examples with the modified functions proceeded by introducing a T-network in place of the negative element C_{23} . This did not introduce a node with the same structure as an existing node and the design proceeded smoothly from minimum to minimum. (This was also the third choice of addition, following the addition of the node identical in structure to an existing node, for the example synthesizing Lucal's functions.) It would seem that situations where existing nodes are duplicated in this manner necessitate the application of a different strategy (see Chapter 7).

Realizations to the other networks for which solutions were not obtained (functions 6.5. 6.7, 6.10 and 6.11) evolved from alternative minimal structures (Figures 6.7 to 6.10).

The T-network was thought to be introduced to greatest effect in the position corresponding to the negative element which produced the greatest reduction in the error value on inclusion at its optimum value. This is a very similar concept to the introduction of an element (section 4.4) about which it has been argued that the virtual element with the largest optimum value should be included[16]. Using this as an alternative location for the introduction of the T-network, realizations were produced to all seven examples previously discussed, from the minimal structure of Figure 6.11(a). The topological modifications were identical with all seven examples (Figure 6.11). The solution values are given in Table 6.3. In general, the introduction of a node in this position did not reduce the error by a factor as large as when the node was introduced in the normal position. The results with this example were more impressive using the alternative method because of the difficulties caused by a sequence of minima with low error values (section 6.5).

6.5. Further Attempts to Realize Lucal's Functions

Four further attempts to realize Lucal's functions from minimal structures are shown in Figures 6.12 to 6.15. Two were successful whilst the others were terminated before the eighth node was included. It can be seen that the example of Figure 6.13 requires the removal of the element R_{57} (value 2.64×10^{-3}) which will remove the common factor of value 8.45×10^{-6}. However, a realization was not obtained when the job was re-submitted with the element removed.

Three other starting structures produced realizations when simultaneous addition and removal of elements was employed (section 4.12). Using the normal method of addition, these examples can fail because of the difficulty associated with the location of minima at low errors. Typical realizations are shown in Figures 6.16 to 6.18. The example with the start structure of Figure 6.17 was attempted, unsuccessfully, by Hegazi[14] and di Mambro[13].

When these same three starting structures were used in the attempted synthesis of functions 6.4, 6.6 and 6.7, difficulty was again experienced in locating similar minima. However, it was not possible to locate the minima by applying simultaneous addition and removal, even though realizations were known to exist with the same topologies. The problems of locating minima were discussed in Chapter 4.

6.6. Computer Realizations of Lucal's Functions Having Only Six Nodes

As stated in the introduction, the realizations produced by series-parallel decomposition all contained at least seven nodes plus the reference. Similarly, all the computer realizations previously shown had required seven nodes. The fact that non-series parallel structures had failed to produce realizations with fewer numbers of elements had reinforced the opinion that solutions did not exist with fewer than seven nodes. However, a realization was discovered for the modified functions 6.5 which required only six nodes. This structure was used as a starting point for the attempted synthesis of Lucal's functions and a six node realization did evolve (Figure 6.19).

Later, an example with a previously used starting topology was re-run allowing element additions which destroyed the compactness of the poles at zero and infinity. This produced a further six node realization (Figure 6.20), of which the four smaller elements are redundant.

Low error values of 10^{-50} do not prove that a solution has been achieved, they merely indicate a good quazi solution with each coefficient matched to 25 figures. The element values of these two networks did not tend to rational values, hence, it is a difficult task to verify that these are in fact solutions. In an attempt to obtain realizations with rational values, jobs were run with series-parallel structures and with the common factor fixed at unity. No series-parallel realizations were obtained and the elements remained irrational with the common factor fixed. (One extra element was required.)

It would appear, from a comparison of the examples of Figures 6.16 and 6.20, that it is preferable to leave element additions unrestricted. The difference between the two examples arose because of the introduction of the element C_{35} in place of the element C_{05} . This type of element replacement occurred only very rarely. Generally, comparative runs from the same start network proceeded in a similar manner, with many more topological modifications required when the element additions were not restricted. With this example, the manoeuvre occurred at a high error value (0.25) and required ten topological modifications before it was accomplished (topologies 6.20c to 6.20m). It is practically impossible for such a substitution of elements to occur at lower errors with additions governed only by the effect of one element in isolation.

6.7. The Program Efficacy When Applied to the Synthesis of Fialkow's Functions

Many different RC realizations had been obtained for Fialkow's functions[13,14,15,16,17,58], both by element addition and node reduction. Provided a reasonable starting network was chosen, the methods of Cutteridge[17,33], di Mambro[13], Krzeczkowski[16] and Hegazi[14] were all able to produce realizations to this problem, presumably aided by the multiplicity of available solutions. The new algorithms developed by the author were able to emulate these results.

The example of Figure 6.21 produced a realization after three topological modifications whereas, using the same start network, Cutteridge[17] achieved a realization after five modifications. An alternative realization was obtained by the simultaneous addition of three elements (Figure 6.22). Although many other multiple additions could be successfully accomplished, this strategy was not normally adopted. One reason for this was exemplified by this particular example. Only two elements were indicated as possible third additions (C_{12} and R_{12}) whereas, in fact, there are a number of possible, alternative additions.

The author found that, in general, a design would evolve if the network was of the form shown in Figure 6.23(a), as opposed to that of Figures 6.23(b) and (c). With the alternative structures as shown in Figures (b) and (c), the corrections were large and unhelpful unless the network was unsymmetrical.

As the start networks become increasingly variant from the solution topologies, then the probability increases that a node addition will be required before a realization can be achieved. No RC realizations to this problem had been achieved after the inclusion of a node. The only successful node addition previous to this research had been accomplished, by Hegazi[14], with an RLC realization of a similar network. The method devised by the author was consistently able to include nodes to produce realizations with both RC and RLC networks. (No RLC realizations of Fialkow's functions could be found with only four nodes.)

A further example of Cutteridge[17], which had only produçed a quazi realization (2.72 x 10^{-5}) was examined. Under similar conditions, an extra element and node were included to produce the first six node realization obtained other than by a trivial transformation (Figure 6.24). This node addition can be avoided when inductors are included to produce RLC realizations. A re-run of the job, considering symmetrical pairs of elements for addition where necessary (which Cutteridge had not), produced an alternative six node realization (Figure 6.25), indicating the existence of a family of similar solutions, as with five nodes.

The evolution of a typical RLC realization which includes the addition of a node is described in Figure 6.26.

6.8. RLC Realizations of Fialkow's Functions Having Series-Parallel Topologies

The addition of the elements R_{13} and R_{23} produced the first such network obtained (Figure 6.27). There were several other possible additions. Of the possible RC additions, four others (R_{12}, C_{12}, C_{03}, C_{14} and C_{24}) would have retained a series-parallel structure. When these further alternative elements were introduced, only one succeeded in producing an alternative realization (Figure 6.28) whilst the others were immediately removed. However, this does not prove conclusively that these topologies are not solutions.

Unfortunately, the author was unable to fully investigate these realizations. However, several elementary points, pertinent to the network of Figure 6.28, are probably worthy of note.

It is not possible to determine, from the printout, the exact nature of the common factors. They appear to be equal in value but, whether they are exactly equal, and whether they are complex, is not discernable. However, the common factors are not equal to the values of the compact, finite poles of Fialkow's functions (i.e. $p + 0.255 \pm \sqrt{0.060225}$). This indicates that the finite poles are fully represented in one of the two subnetworks, and not decomposed.

The subnetwork containing node three gives a polynomial in Δ_{1122} which indicates the poles to be $(p + 0.255 + \sqrt{0.060225})$ and $(p + A)$, where A is the value of one common factor. Hence, the further poles must be produced by the other RC subnetwork. This eliminates the possibility of complex common factors.

An examination of Δ_{12} for the same subnetwork appears to indicate that a common factor of $(p + A)$ is not present. These initial observations would appear to indicate a situation where a pole is included into both subnetworks such that the net effect is zero, i.e. the residues are of equal magnitude but of opposite signs. In this case the common factors will be equal.

6.9. Attempted Synthesis of Network Functions Differing Slightly from Those Given by Fialkow

Krzeczkowski[16] has conjectured that it is not possible to produce a network having a negative coefficient in the cofactor Δ_{12}, as in Fialkow's functions, but with all of the poles compact. However, Krzeckowski had only examined the general case with five node, RC networks. Hegazi[14] was unable to find such a network by applying coefficient matching techniques. The finite poles of Fialkow's functions are already compact. The synthesis of similar networks (retaining compact, finite poles and a negative valued coefficient), produced by reducing the residues of the poles of y_{11} and y_{22} and having

(i) a compact pole at zero (functions 6.15)

(ii) a compact pole at infinity (functions (6.16)

(iii) compact poles at zero and infinity (functions 6.17),

was investigated. A fourth example (functions 6.18), again with all poles compact, devised (but not solved) by Hegazi, was also examined.

A solution to example 1 (functions 5.15) had already been produced by Cutteridge and Krzeczkowski[58] and many others could be similarly produced by the removal of one element, or a symmetrical pair, from a realization of Fialkow's functions.

Realizations of the three other sets of functions were not so readily achieved. Several of the elements connected to internal nodes were reduced to small values (of the order of 10^{-6}) and the jobs had to be run in double precision on the Cyber 72 (thirty decimal places). With RC networks, nodes frequently shorted out after the addition of an element and, although good quazi solutions were obtained, no realizations were located. (Topologies similar to the six node RC realizations of Fialkow's functions were utilized as starting topologies as well as five node RC structures.) However, RLC realizations were obtained (Figures 6.29 to 6.31) including examples with series-parallel topologies similar to those for Fialkow's functions

(Figures 6.32 and 6.33).*

Example five (functions 6.19) was constructed to produce two small negative valued coefficients in the cofactor Δ_{12} . Any realization of these functions with a passive network having positive valued elements would require at least two common factors. Starting topologies (both RC and RLC) which realized Fialkow's functions with two common factors were used, but the error value was not significantly reduced on any occasion. Furthermore, when extra nodes and inductors were included, thus increasing the numbers of common factors, the program still failed to make progress.

Example six (functions 6.20) shifted the values of the residues of y_{11} and y_{22} by the same amount as the previous example had shifted the corresponding residues of y_{12} . The program had no difficulty in realizing these functions (Figure 6.34).

Example seven (functions 6.21) is a network having finite poles of rational values (0.5 and 0.001). The residues of these finite poles were calculated such that the coefficients of the cofactor Δ_{12} remained unaltered. Realizations similar to those for Fialkow's functions were obtained (Figure 6.35).

If the formulation of a similar example is attempted (i.e. the values of the coefficients of the cofactor Δ_{12} are to remain unchanged) with finite poles of value one and two, then the functions are not

* Subsequent to the compilation of this thesis the network of Figure 6.33 was investigated by the author and found to be an exact realization (see Appendix 5). It it therefore possible to construct an exact RLC-series-parallel realization of Fialkow's functions by including further resistive and capacitive elements between nodes one(two) and zero.

$$R_{01} = R_{02} = \frac{113.568}{3840} = 0.029575 \text{ S.}$$

$$C_{01} = C_{02} = \frac{1194}{800,000} = 0.0014975 \text{ F.}$$

feasible for an RC network. (The residue of y_{11} at the pole p + 2 will have a negative value.)

6.10. Attempted Realization as RLC Networks of RC Realizable Sets of Functions

Whilst RLC realizations of Fialkow's functions have been given, and the existence of many others indicated, RLC realizations of Lucal's functions have not been referred to. As RC realizations have been located, it seems reasonable to assume that RLC realizations also exist. However, no exact RLC realizations were found, although good quazi realizations were obtained (error values of 10^{-8} were achieved).

The behaviour of the two networks was quite different. A surplus inductor was included at various positions into RC structures known to realize the functions and the network optimized from arbitrary values of unity. When attempting to synthesize Fialkow's network, one of the RC components was always removed to produce a quazi solution to the network. Presumably the removal of the surplus inductor was deferred by the existence of its accompanying common factor. Subsequent element addition produced a realization. However, when attempting to synthesize Lucal's functions, the inductor and its associated common factor were either removed or reduced to such low values that they should have been removed. Some inherent feature of Lucal's functions was apparently opposing the introduction of an inductor.

		Lucal's functions (6.1)	FUNCTIONS							
			6.4	6.5	6.6	6.7	6.8	6.9	6.10	6.11
Δ_{11}	p^4	36		108						
	p^3	533	813	1319	668	465.5	569	515	905.6	785
	p^2	1572	2412	3876	2112	1302	1752	1482	2571	3567
	p^1	1183	1743	2989	1588	980.5	1399	1075	1766.2	3262
	p^0	36		108						120
Δ_{12}	p^4	36		108						
	p^3	36		108					273.6	288
	p^2	72		216					595.8	2508
	p^1	36		108					343.8	2052
	p^0	36		108					25.2	120
Δ_{22}	p^4	36		108						
	p^3	2058	1218	13734	1788	2598	2040	2094	2025.6	2,310
	p^2	6552	4032	42,336	5472	8712	6462	6732	5931	11,382
	p^1	4638	2958	29034	3828	6258	4530	4854	4006.2	9552
	p^0	36		108					25.2	120
Δ_{1122}	p^3	36		108						
	p^2	216		648					183.6	468
	p^1	396		1188					298.8	1152
	p^0	216		648					151.2	720

Coefficient values as for Lucal's functions unless otherwise indicated.

TABLE 6.1. VALUES FOR THE COEFFICIENTS OF FUNCTIONS OBTAINED BY SLIGHTLY MODIFYING LUCAL'S FUNCTIONS

		Fialkow's functions (6.12)	FUNCTIONS						
			6.15	6.16	6.17	6.18	6.19	6.20	6.21
Δ_{11} and Δ_{22}	p^3	1197		3	3	100		1977	
	p^2	56,613.14	32,953.14	56,004.2	32,344.2	32344.84		56,610.94	
	p^1	28,368.584	16,301.984	28,362.8528	16,296.2528	16,271,8151		28,168.328	28,368.824
	p^0	191.184	77.616		77.616	77.381616		189.264	199.15
Δ_{12}	p^3	3				100	783		
	p^2	-1.14				-0.5	-3.34		
	p^1	197.176				172.7383	-3.08		
	p^0	77.616				77.381616	75.696		80.85
Δ_{1122}	p^2	800,000							
	p^1	408,000							
	p^0	348o							4000

Coefficient values as for Fialkow's functions unless otherwise indicated.

TABLE 6.2. VALUES FOR THE COEFFICIENTS OF FUNCTIONS OBTAINED BY SLIGHTLY MODIFYING FIALKOW'S FUNCTIONS

	Lucal's functions (6.1)	NETWORK FUNCTIONS					
		6.4	6.5	6.6	6.7	6.8	6.9
C_{13}	(14.0)	(8.00)	(32.0)	(12.0)	(18.0)	(14.0)	(14.0)
C_{04}	6.35	4.75	11.6	4.66	9.79	5.97	7.05
C_{05}	1.60	2.10	1.49	2.51	1.13	2.38	1.23
C_{46}	0.926	0.778	1.04	0.876	0.985	0.789	1.25
C_{56}	0.734	0.802	0.672	0.698	0.775	0.975	0.590
C_{45}	3.72×10^{-2}	3.17×10^{-2}	3.62×10^{-2}	2.87×10^{-2}	4.59×10^{-2}	4.89×10^{-2}	3.18×10^{-2}
C_{27}	(3.00)	(5.00)	$(2.\dot{3})$	(4.00)	(2.50)	(3.00)	(3.00)
C_{37}	(1.68)	$(\overline{1.48})$	1.85	(1.50)	1.84	(1.68)	(1.68)
G_{14}	16.2	10.8	32.2	$(12.\dot{6})$	23.2	$(15.\dot{6})$	1.72
G_{45}	0.176	0.175	0.177	0.174	0.178	0.175	0.177
G_{25}	3.81	5.58	3.21	$(5.5\dot{5})$	2.93	4.81	3.31
G_{03}	(39.2)	$(21.\dot{3})$	93.1	(36.0)	46.2	(39.2)	(39.2)
G_{06}	2.10	1.93	2.16	1.89	2.37	2.45	2.24
G_{07}	(10.8)	$(16.\dot{6})$	8.91	(12.0)	10.7	(10.8)	(10.8)

All have common factors $(p + 2)(p + 3)$. Brackets indicate exact values.

TABLE 6.3. VALUES OF THE ELEMENTS OF THE NETWORK OF FIGURE 6.11d AT REALIZATIONS OF SEVEN SETS OF FUNCTIONS

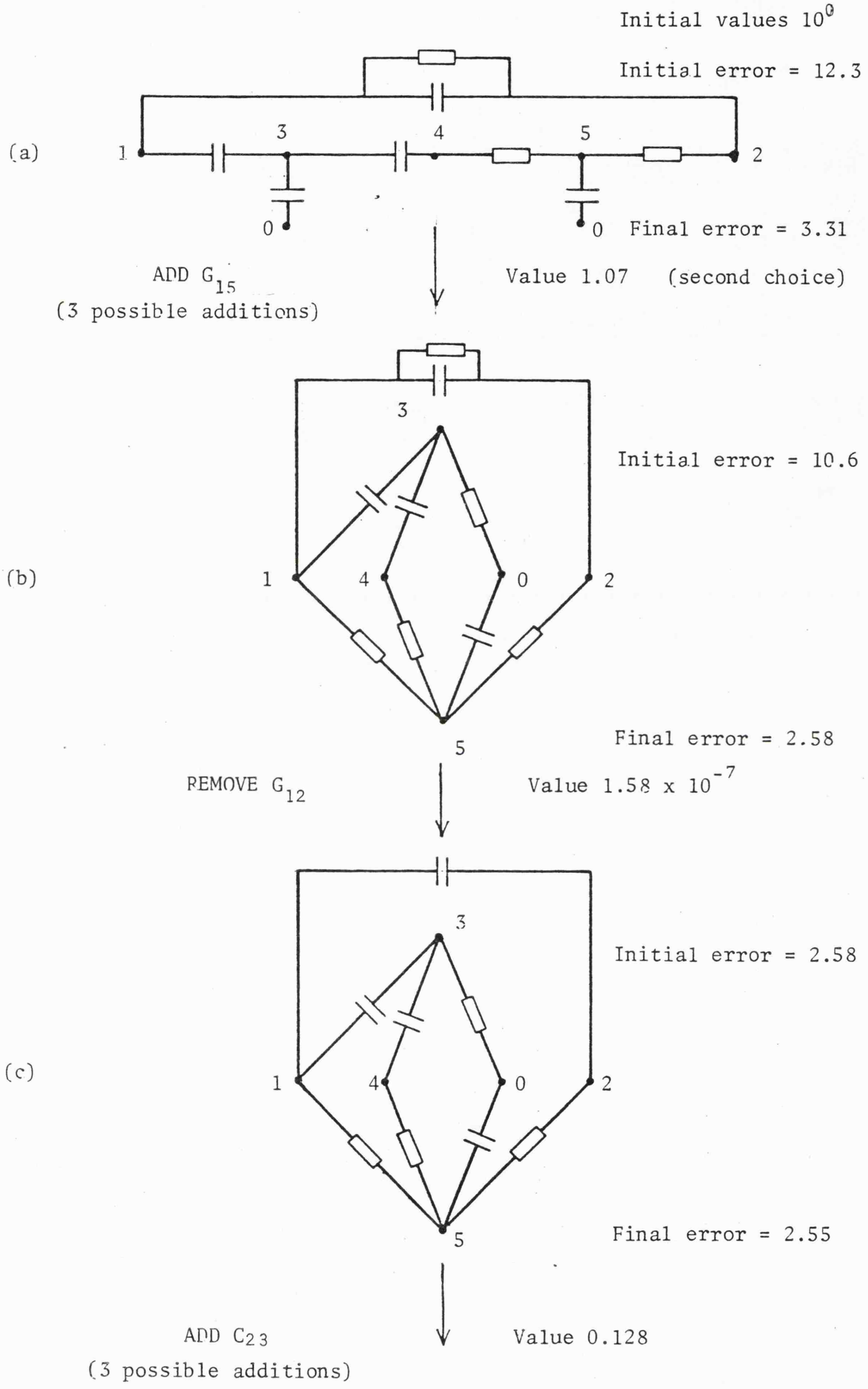
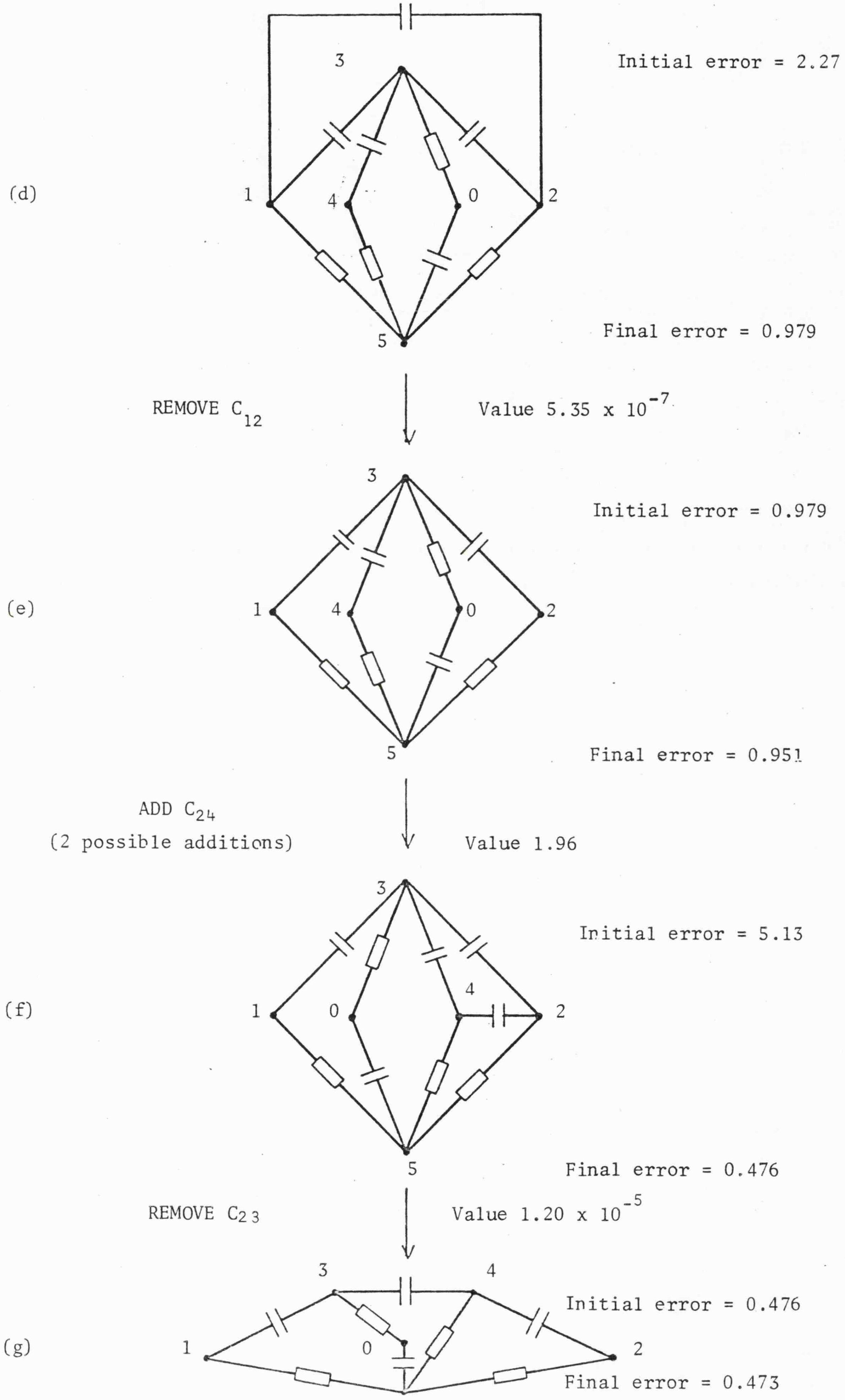


FIGURE 6.1: EVOLUTION IN DETAIL OF A NETWORK WHICH REALIZES LUCAL'S FUNCTIONS

(contd. over)



Initial error = 2.27

Final error = 0.979

REMOVE C_{12}

Value 5.35 x 10^{-7}

Initial error = 0.979

Final error = 0.951

ADD C_{24}

(2 possible additions)

Value 1.96

Initial error = 5.13

Final error = 0.476

REMOVE C_{23}

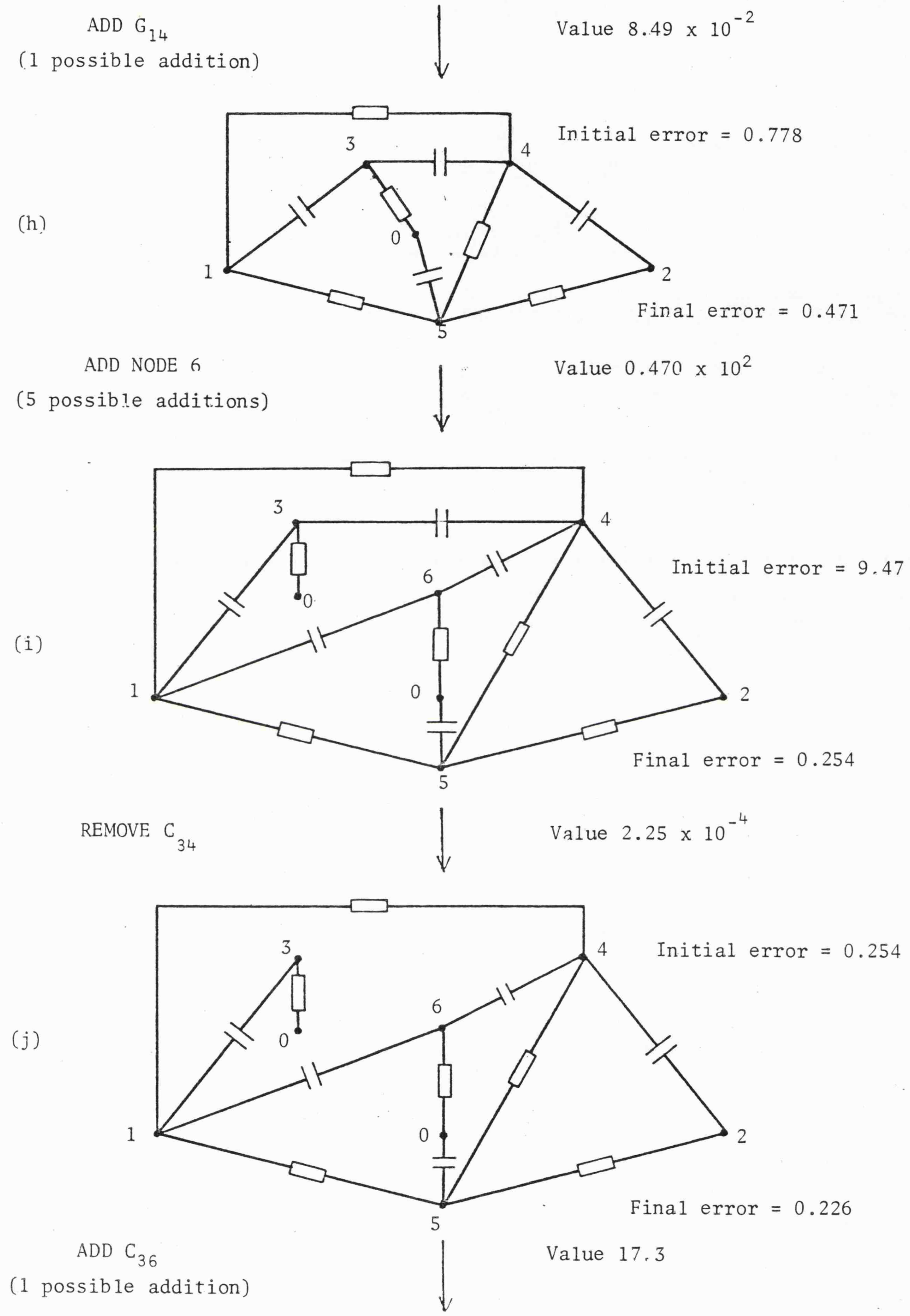
Value 1.20 x 10^{-5}

Initial error = 0.476

Final error = 0.473

FIGURE 6.1 CONTINUED

(contd.over)



ADD G_{14}
(1 possible addition)

Value 8.49×10^{-2}

(h)

Initial error = 0.778

Final error = 0.471

ADD NODE 6
(5 possible additions)

Value 0.470×10^{2}

(i)

Initial error = 9.47

Final error = 0.254

REMOVE C_{34}

Value 2.25×10^{-4}

(j)

Initial error = 0.254

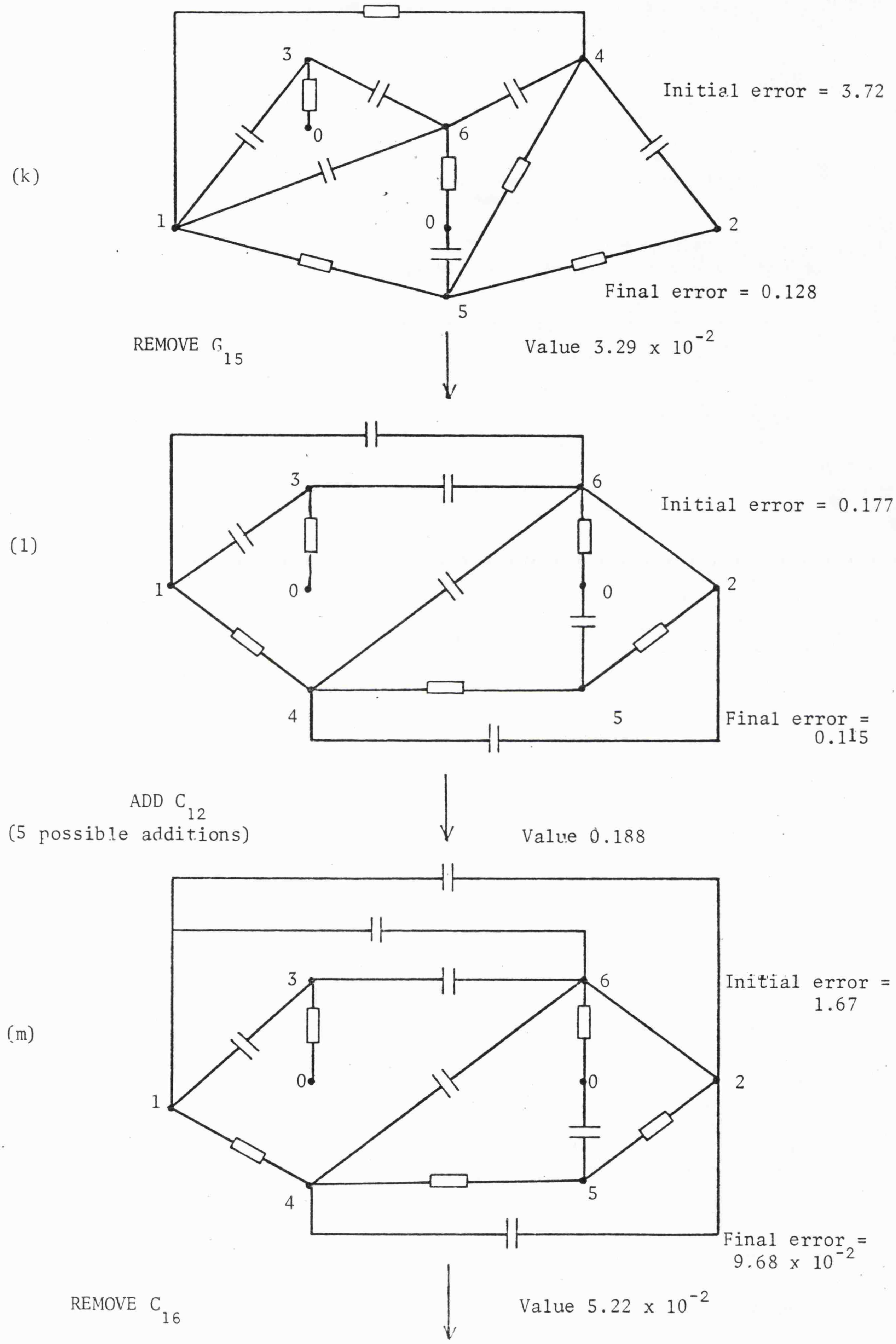
Final error = 0.226

ADD C_{36}
(1 possible addition)

Value 17.3

FIGURE 6.1: CONTINUED

(Contd.over)



(k) Initial error = 3.72

Final error = 0.128

REMOVE G_{15}

Value 3.29×10^{-2}

(1) Initial error = 0.177

Final error = 0.115

ADD C_{12}
(5 possible additions)

Value 0.188

(m) Initial error = 1.67

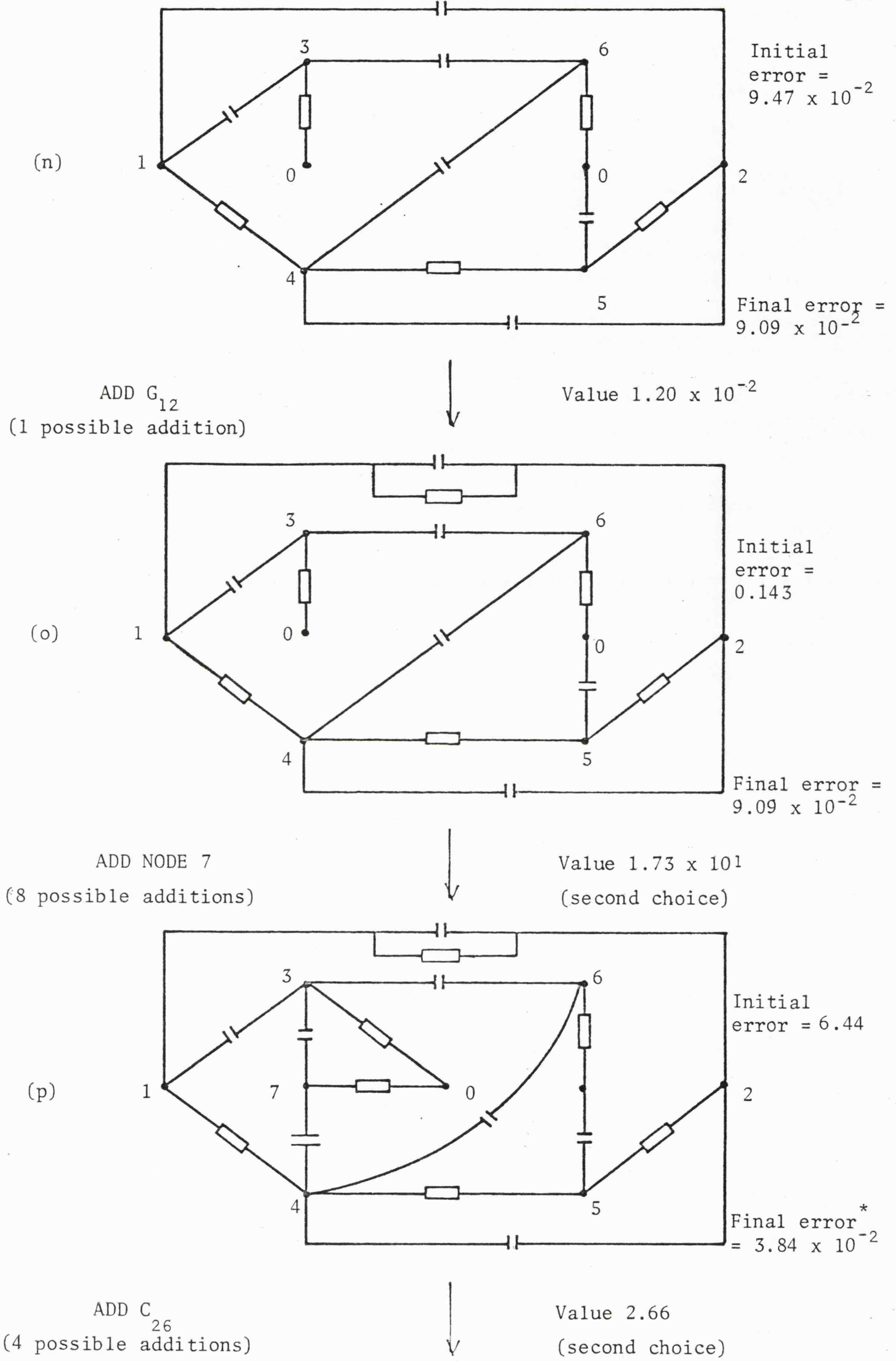
Final error = 9.68×10^{-2}

REMOVE C_{16}

Value 5.22×10^{-2}

FIGURE 6.1: CONTINUED

(Contd. over)



Initial error = 9.47×10^{-2}

Final error = 9.09×10^{-2}

ADD G_{12}
(1 possible addition)

Value 1.20×10^{-2}

Initial error = 0.143

Final error = 9.09×10^{-2}

ADD NODE 7
(8 possible additions)

Value 1.73×10^{1}
(second choice)

Initial error = 6.44

Final error* = 3.84×10^{-2}

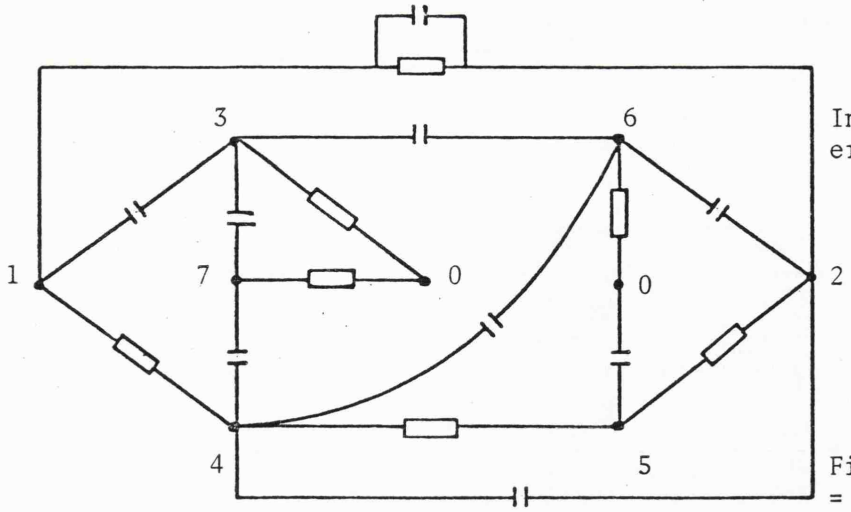
ADD C_{26}
(4 possible additions)

Value 2.66
(second choice)

FIGURE 6.1: CONTINUED

(Contd. over)

(q)

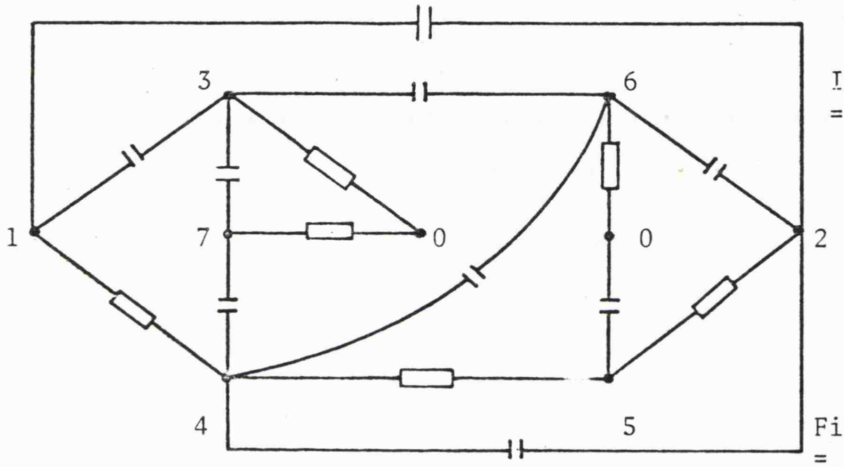


Initial error = 6.83

Final error = 2.21×10^{-2}

REMOVE G_{12} Value 8.28×10^{-8}

(r)

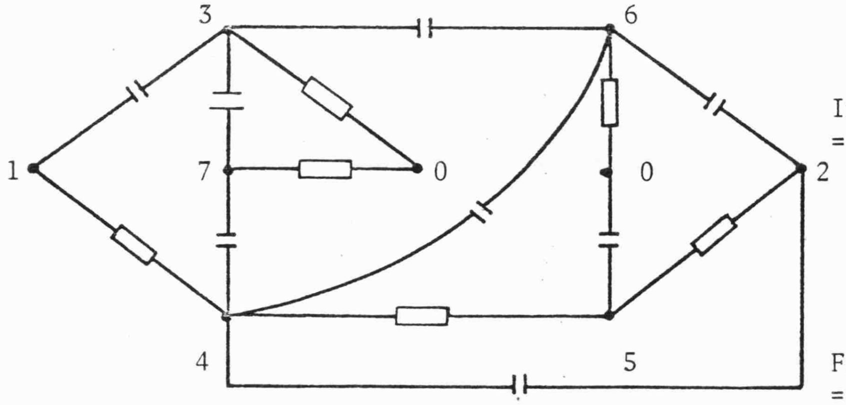


Initial error = 2.21×10^{-2}

Final error = 6.01×10^{-3}

REMOVE C_{12} Value 2.31×10^{-2}

(s)



Initial error = 3.60×10^{-2}

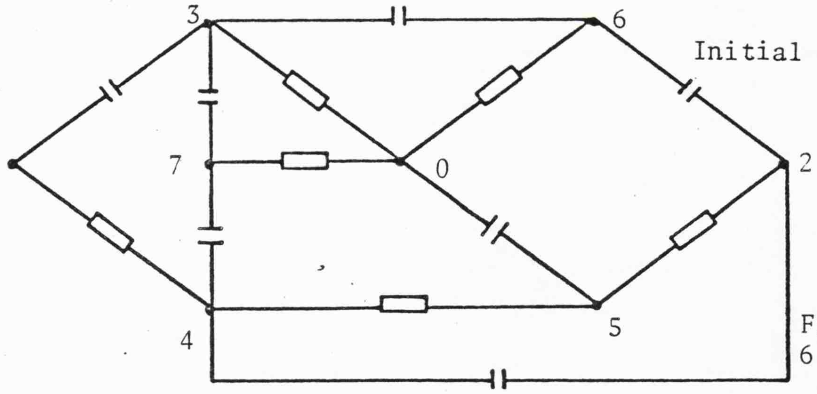
Final error = 2.26×10^{-3}

REMOVE C_{46} Value 0.555

FIGURE 6.1: CONTINUED

(Contd. over)

(t)



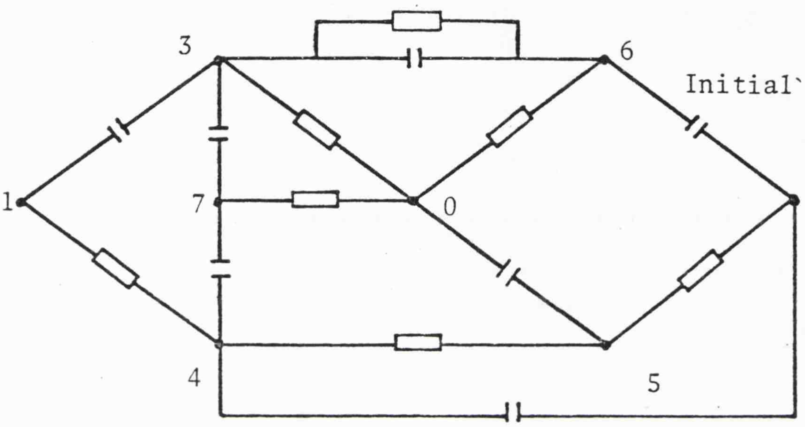
Initial error = 1.24

Final error = 6.98×10^{-3}

ADD G_{36}
(5 possible additions)

Value 0.872

(u)



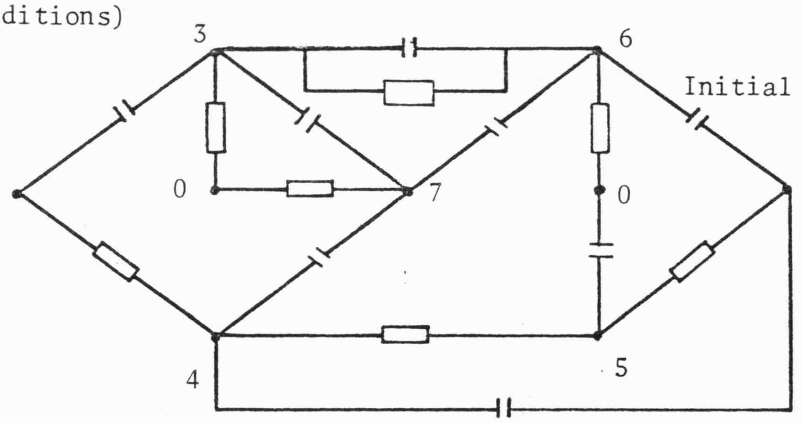
Initial` error = 0.184

Final error* $= 6.27 \times 10^{-3}$

ADD C_{67}
(6 possible additions)

Value 1.23

(v)



Initial error = 1.81

Final error $= 6.40 \times 10^{-4}$

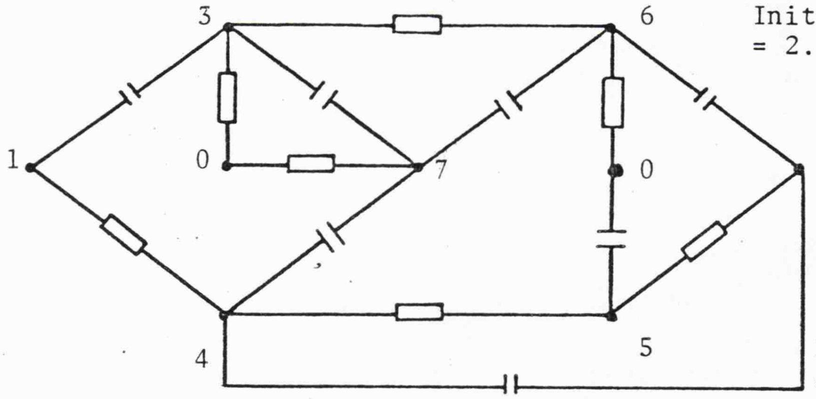
REMOVE C_{36}

Value 5.21×10^{-2}

FIGURE 6.1: CONTINUED

(Contd. over)

(w)



Initial error $= 2.71 \times 10^{-3}$

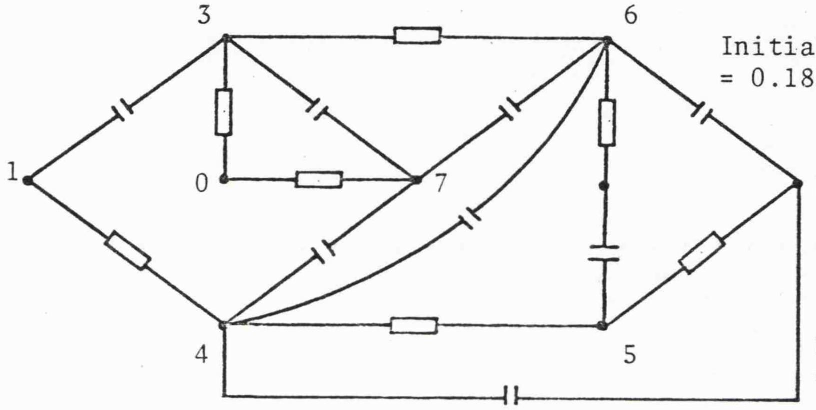
Final error $= 7.03 \times 10^{-6}$

ADD C_{46}

(2 possible additions)

Value 0.143

(x)



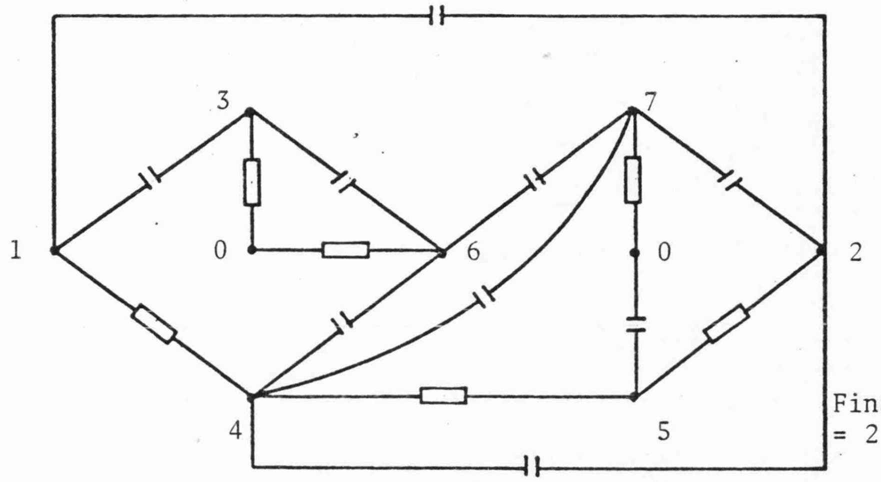
Initial error = 0.187

Final error $= 8.49 \times 10^{-22}$

$C_{13} = 20.6$	$G_{25} = 0.992$
$C_{03} = 0.911$	$G_{45} = 0.269$
$C_{24} = 0.175$	$G_{03} = 52.2$
$C_{37} = 3.46$	$G_{14} = 0.785$
$C_{47} = 0.406$	$G_{06} = 10.1$
$C_{26} = 4.40$	$G_{07} = 6.48$
$C_{67} = 1.96$	$G_{36} = 0.549$
$C_{46} = 0.314$	

Common factors 1.53 and 0.862

FIGURE 6.1: CONTINUED

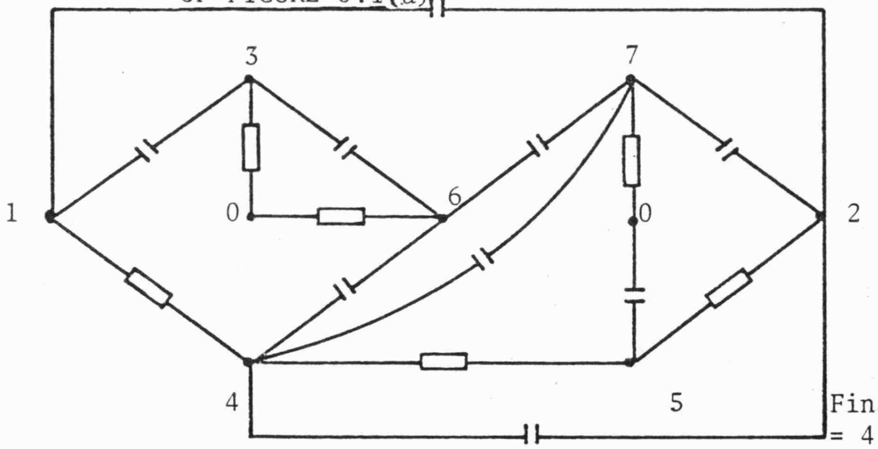


Final error = 2.96×10^{-20}

C_{13} = 13.7	C_{12} = 0.0534	G_{25} = 1.03
C_{05} = 0.815	C_{27} = 7.01	G_{03} = 29.9
C_{24} = 0.0991	C_{67} = 2.08	G_{06} = 5.02
C_{46} = 0.344	C_{47} = 0.502	G_{14} = 0.805
C_{36} = 2.46	G_{45} = 0.264	G_{07} = 18.2

Common factors 1.63 and 0.842

FIGURE 6.2: REALIZATION OBTAINED FOR FUNCTIONS 6.4 FROM THE STARTING TOPOLOGY OF FIGURE 6.1(a)

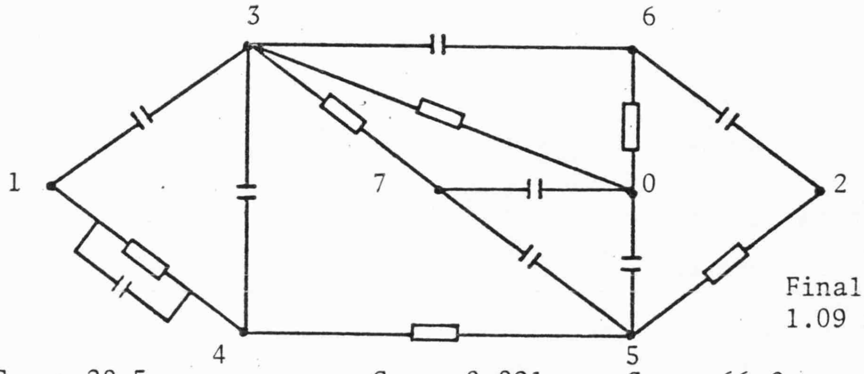


Final error = 4.83×10^{-20}

C_{13} = 16.7	C_{12} = 0.0602	G_{25} = 1.30
C_{05} = 1.00	C_{27} = 6.12	G_{03} = 45.9
C_{24} = 0.0969	C_{67} = 1.55	G_{06} = 4.68
C_{46} = 0.465	C_{47} = 0.405	G_{14} = 0.778
C_{36} = 3.42	G_{45} = 0.254	G_{07} = 14.0

Common factors 1.61 and 0.728

FIGURE 6.3: REALIZATION OBTAINED FOR FUNCTIONS 6.6 FROM THE STARTING TOPOLOGY FIGURE 6.1(a)

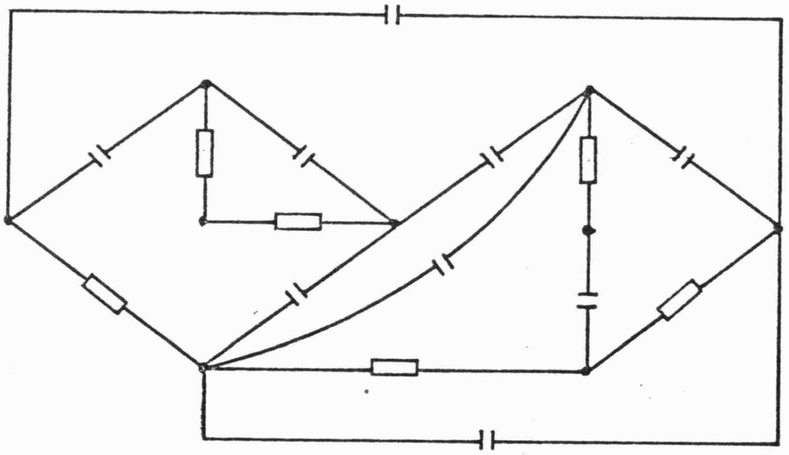


Final error = 1.09×10^{-8}

$C_{13} = 28.5$	$C_{07} = 0.821$	$G_{03} = 66.8$
$C_{05} = 1.32$	$C_{57} = 0.643$	$G_{14} = 0.678$
$C_{36} = 1.80$	$C_{14} = 0.392$	$G_{06} = 11.1$
$C_{26} = 2.44$	$G^{-}_{25} = 0.239$	$G_{37} = 2.02$
$C_{45} = 0.319$	$G_{25} = 2.98$	

Common factors 2.52 and 1.30

FIGURE 6.4: QUASI REALIZATION OBTAINED FOR FUNCTIONS 6.7 FROM THE STARTING TOPOLOGY OF FIGURE 6.1(a)

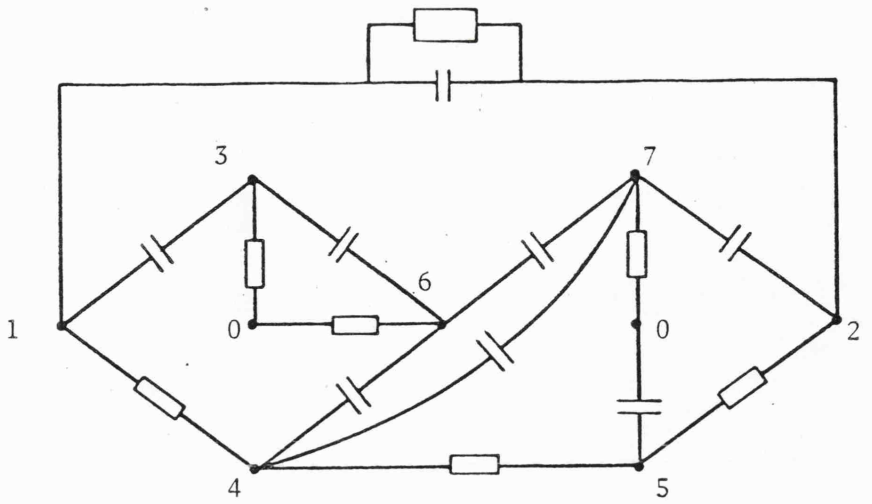


Final error = 4.74×10^{-21}

$C_{13} = 20.1$	$C_{27} = 4.92$	$G_{45} = 0.260$
$C_{05} = 1.38$	$C_{24} = 0.0819$	$G_{03} = 52.6$
$C_{46} = 0.365$	$C_{67} = 1.96$	$G_{06} = 5.78$
$C_{36} = 2.64$	$C_{47} = 0.314$	$G_{14} = 0.682$
$C_{12} = 0.0789$	$G_{25} = 1.47$	$G_{07} = 10.4$

Common factors 1.40 and 0.894

FIGURE 6.5: REALIZATION OBTAINED FOR FUNCTIONS 6.8 FROM THE STARTING TOPOLOGY OF FIGURE 6.1(a)



$C_{13} = 21.0$	$C_{27} = 4.35$	$G_{25} = 0.543$	$G_{06} = 5.53$
$C_{05} = 0.429$	$C_{67} = 1.43$	$G_{45} = 0.155$	$G_{07} = 11.6$
$C_{46} = 0.772$	$C_{47} = 0.541$	$G_{03} = 52.7$	$G_{12} = 0.0572$
$C_{36} = 5.20$		$G_{14} = 1.18$	
$C_{12} = 0.0403$			

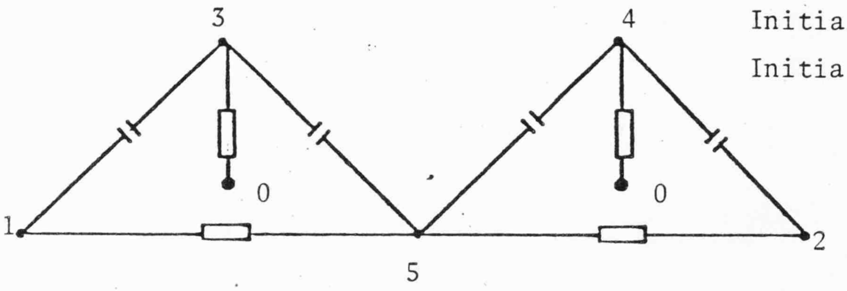
Common factors = 1.66 and 0.631

FIGURE 6.6: REALIZATION OBTAINED FOR FUNCTIONS 6.9 FROM THE STARTING TOPOLOGY OF FIGURE 6.1(a)

Initial values = 10^0

Initial error = 9.28

(a)



Final error = 0.475

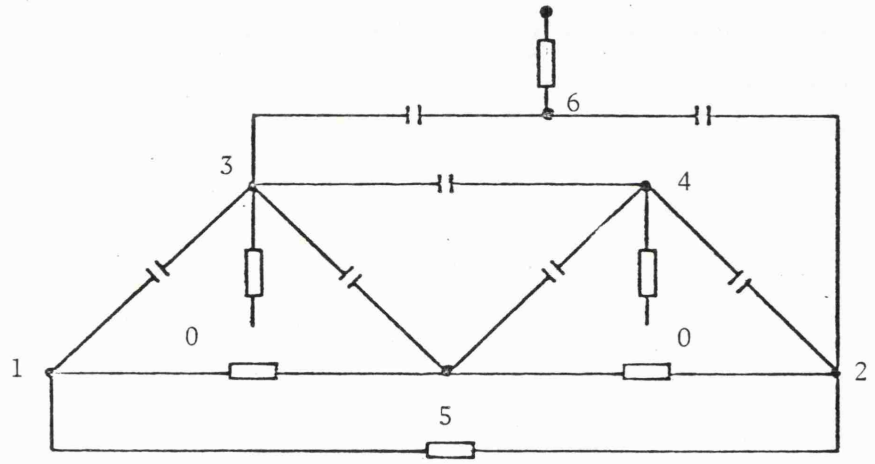
(b) ADD C_{34} Final error = 0.145

(c) ADD C_{15} Final error = 0.103

(d) ADD G_{12}

(e) REMOVE C_{15} Final error = 9.14×10^{-2}

(f) ADD NODE 6



(g) REMOVE C_{34} Final error = 1.87×10^{-2}

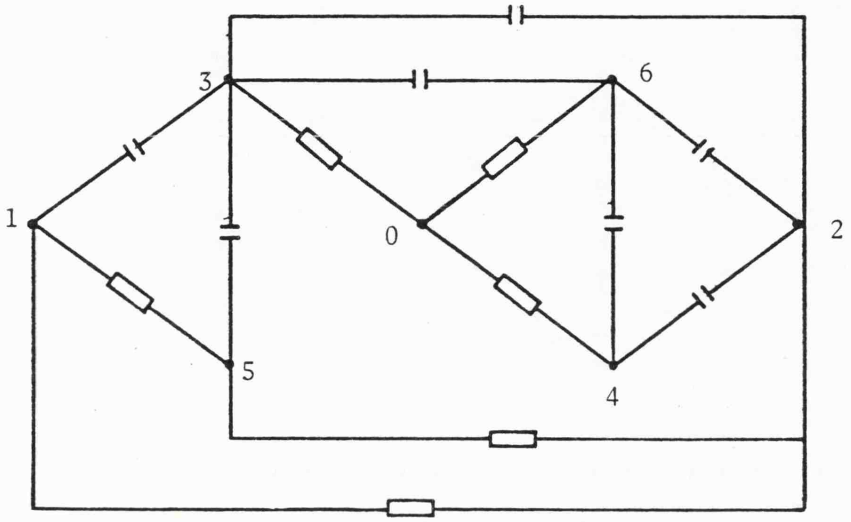
(h) ADD C_{46}

(i) REMOVE C_{45} Final error = 1.13×10^{-12}

(j) ADD C_{23}

FIGURE 6.7: REALIZATION OF FUNCTIONS 6.5

(Contd. over)



$C_{13} = 44.5$

$C_{35} = 0.925$

$C_{24} = 3.54$

$C_{26} = 0.129$

$C_{36} = 3.65$

$C_{46} = 1.92$

$C_{34} = 2.43 \times 10^{-2}$

Common factor = 1.03

$G_{15} = 0.175$

$G_{25} = 0.774$

$G_{03} = 122$

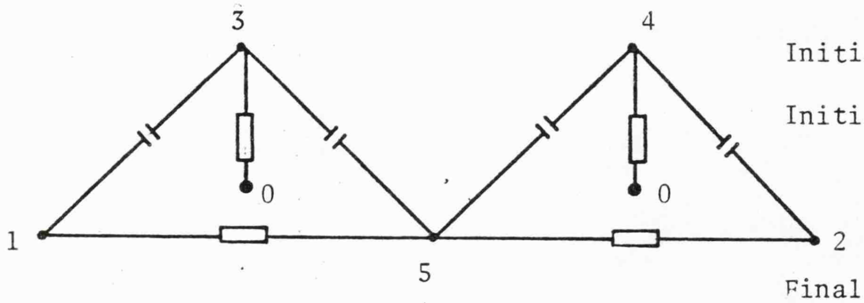
$G_{04} = 8.49$

$G_{12} = 2.42 \times 10^{-2}$

$G_{06} = 7.28$

FIGURE 6.7: CONTINUED

(a)



Initial values = 10^0

Initial error = 9.24

Final error = 0.522

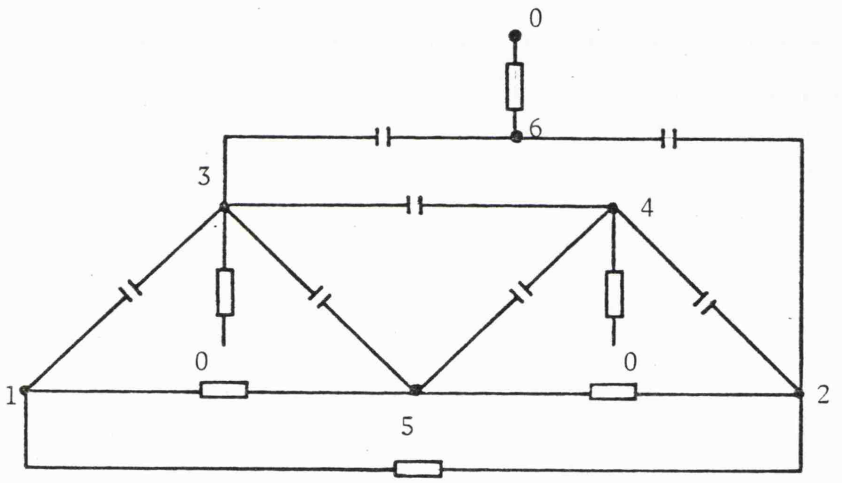
(b) ADD C_{34} Final error = 0.132

(c) ADD C_{15} Final error = 9.29×10^{-2}

(d) ADD G_{12}

(e) REMOVE C_{15} Final error = 8.39×10^{-2}

(f) ADD NODE 6

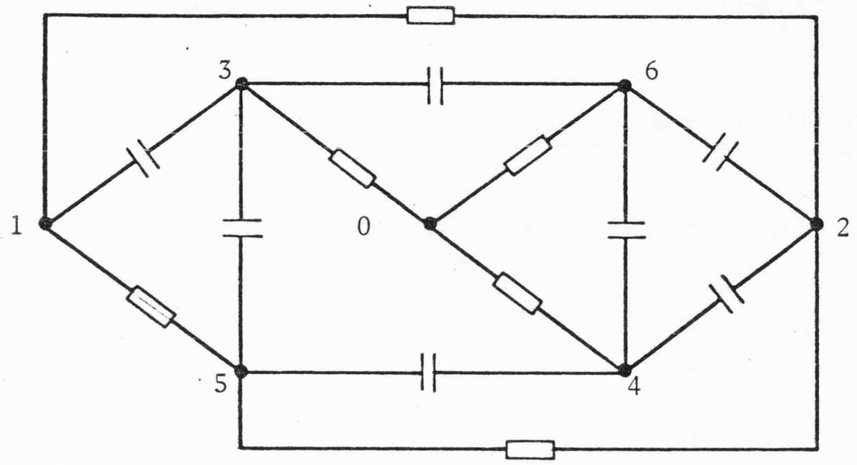


(g) REMOVE C_{34} Final error = 1.92×10^{-2}

(h) ADD C_{46} Final error = 2.71×10^{-23}

FIGURE 6.8: REALIZATION OF FUNCTIONS 6.7

(Contd. over)



$C_{13} = 28.6$

$C_{35} = 0.802$

$C_{45} = 0.124$

$C_{24} = 3.33$

$C_{26} = 0.289$

$C_{36} = 2.55$

$C_{46} = 2.11$

Common factor = 1.10

$G_{15} = 0.180$

$G_{25} = 0.798$

$G_{03} = 67.0$

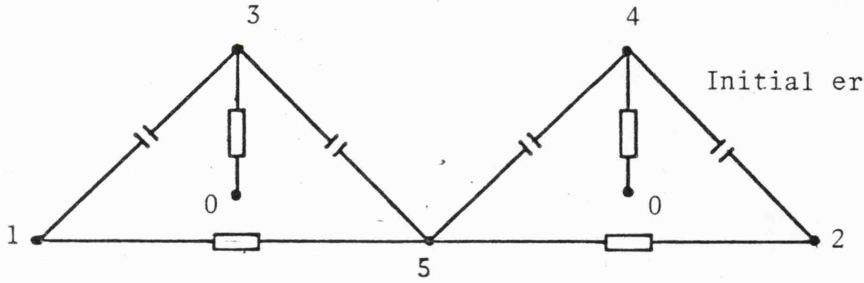
$G_{04} = 9.06$

$G_{12} = 2.00 \times 10^{-2}$

$G_{06} = 7.01$

FIGURE 6.8: CONTINUED

(a)



Initial error = 9.74

Final error = 0.836

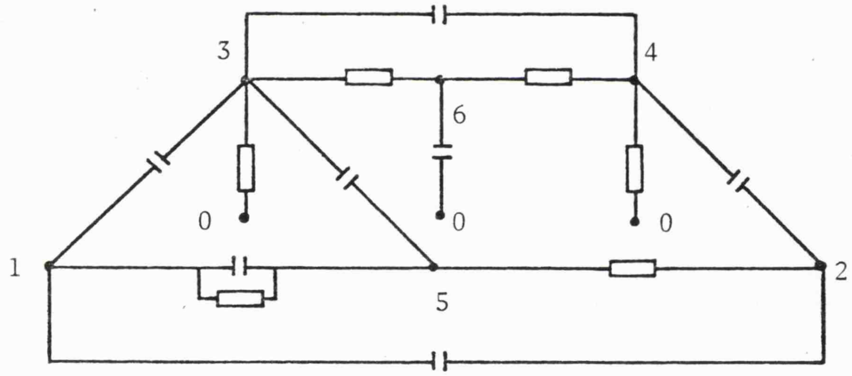
(b) ADD C_{15} Final error = 3.12×10^{-2}

(c) ADD C_{12} Final error = 9.04×10^{-4}

(d) ADD C_{34}

(e) REMOVE C_{45} Final error = 8.07×10^{-4}

(d) ADD NODE 6



(g) REMOVE C_{34} Final error = 2.24×10^{-7}

(h) ADD G_{34} Final error* = 5.58×10^{-15}

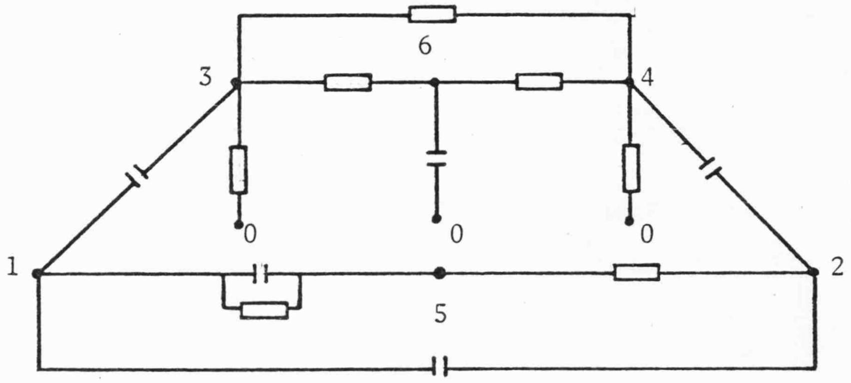
(i) ADD C_{14}

(j) REMOVE C_{35}

(k) REMOVE C_{14} Final error = 4.75×10^{-24}

FIGURE 6.9: REALIZATION OF FUNCTIONS 6.10

(Contd. over)



$C_{13} = (24.\dot{2})$

$C_{24} = (9.\overline{407})$

$C_{15} = 1.12$

$C_{12} = (1.00)$

$C_{06} = 3.07$

$G_{15} = 0.182$

Common factor 2.00

$G_{25} = (2.0\dot{5})$

$G_{03} = 46.8$

$G_{04} = 15.9$

$G_{36} = 1.86$

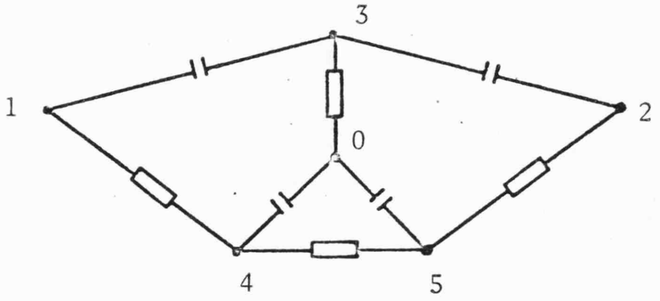
$G_{46} = 1.70$

$G_{34} = 0.444$

( ) - indicates exact values

FIGURE 6.9: CONTINUED

(a)



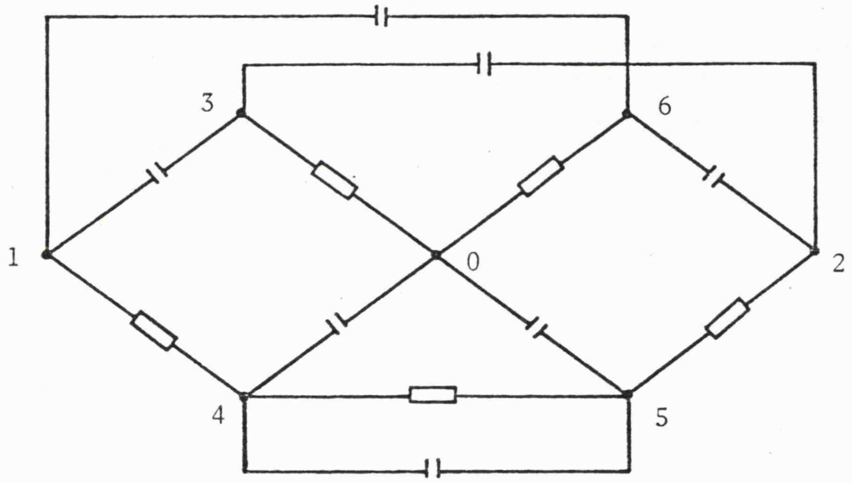
Initial values = 10^0

Initial error = 10.5

Final error = 1.35

(b) ADD C_{45} Final error = 5.04×10^{-2}

(c) ADD NODE 6



(d) REMOVE C_{05} Final error = 5.17×10^{-3}

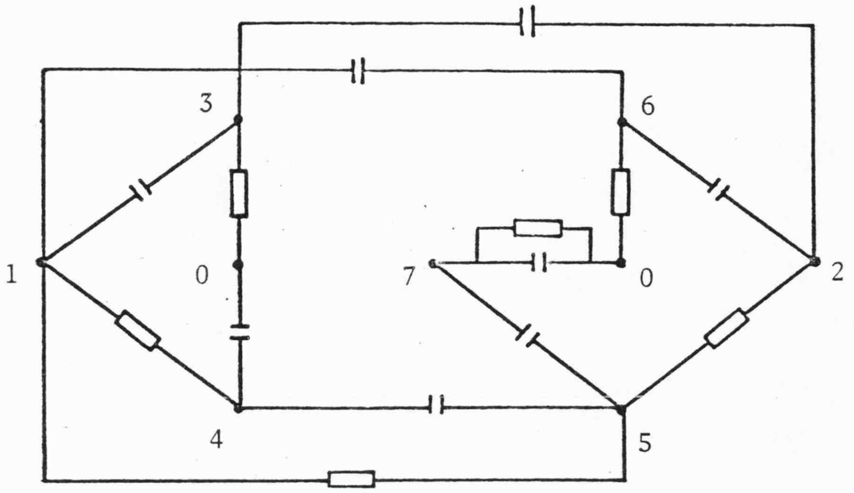
(e) ADD G_{15}

(f) REMOVE G_{45} Final error = 4.98×10^{-3}

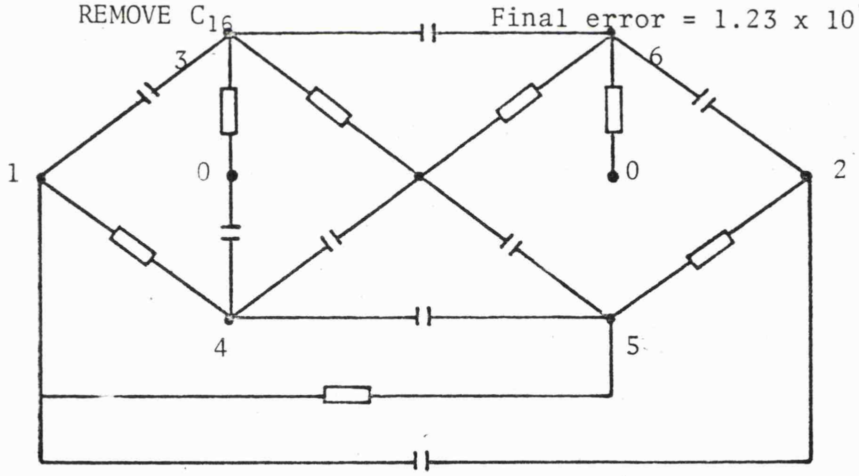
(g) ADD NODE 7

FIGURE 6.10: REALIZATIONS OF FUNCTIONS 6.11

(Contd. over)



(h)	REMOVE C_{07}	Final error = 1.04×10^{-3}
(i)	ADD C_{12}	Final error = 8.42×10^{-4}
(j)	ADD G_{37}	Final error = 7.44×10^{-4}
(k)	ADD C_{47}	
(l)	REMOVE G_{37}	
(m)	REMOVE C_{23}	Final error = 2.02×10^{-4}
(n)	ADD G_{67}	
(o)	REMOVE G_{07}	Final error = 7.30×10^{-5}
(p)	ADD C_{36}	
(q)	REMOVE C_{16}	Final error = 1.23×10^{-19}

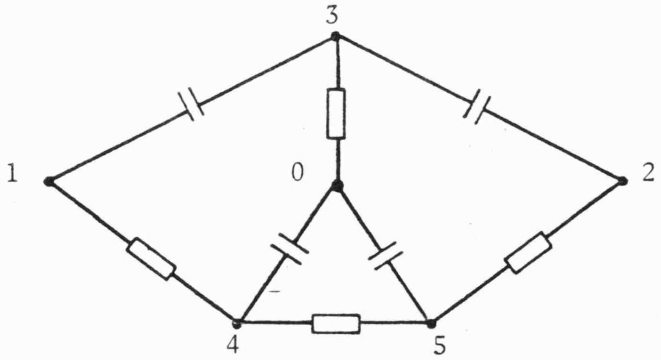


C_{13} = 4.77	G_{14} = 28.4
C_{04} = 4.72	G_{25} = 7.02
C_{45} = 2.14	G_{03} = 15.0
C_{26} = 0.627	G_{06} = 16.2
C_{57} = 1.08	G_{15} = 0.171
C_{12} = 0.573	G_{67} = 0.0443
C_{47} = 0.903	G_{37} = 3.08
C_{38} = 1.87	

Common factors = 8.82 and 2.75

FIGURE 6.10: CONTINUED

(a)



(b) ADD C_{45}

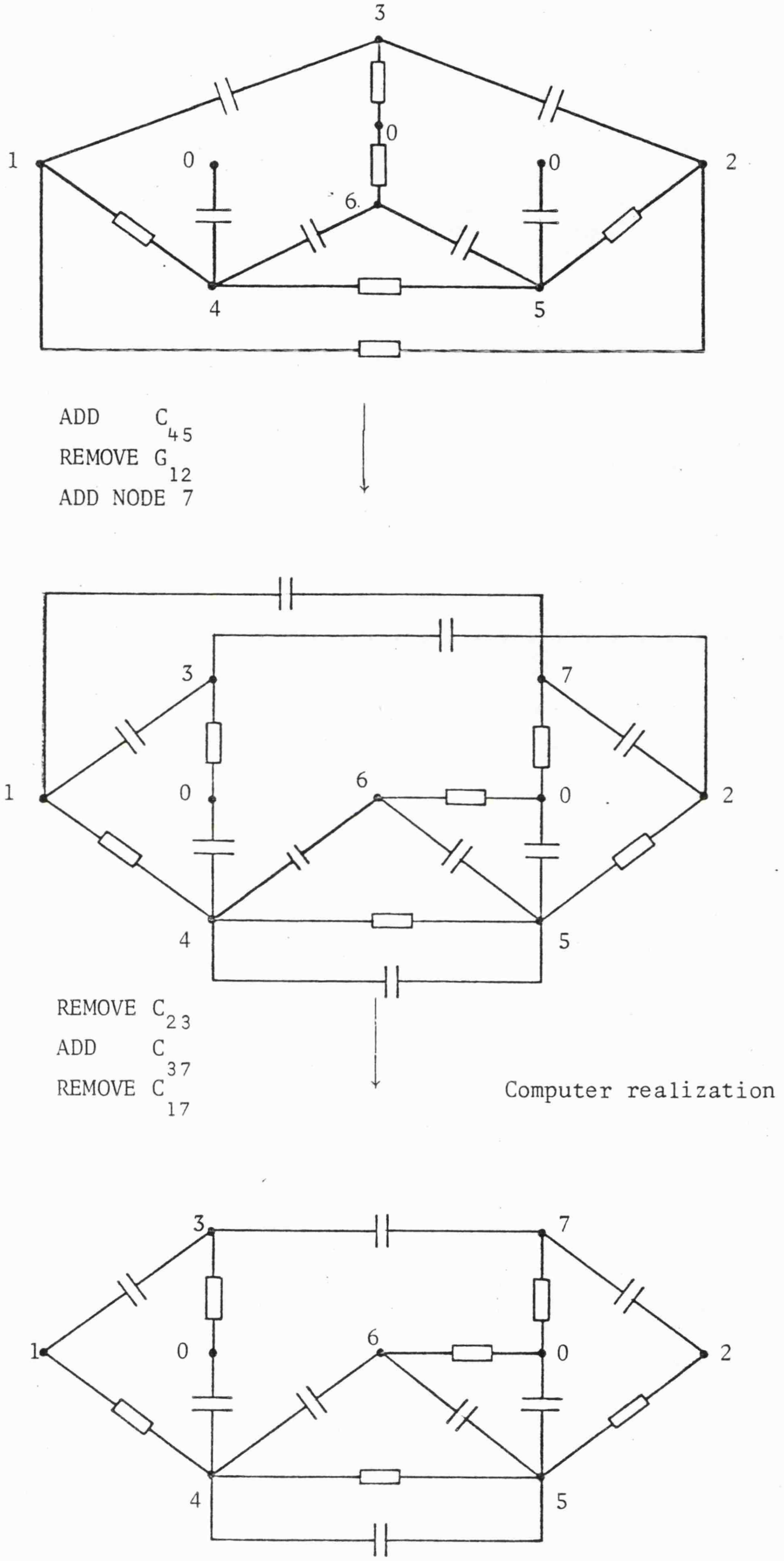
(c) ADD R_{12}

(d) REMOVE C_{45}

(e) ADD NODE 6

FIGURE 6.11: STRUCTURAL CHANGES WHEN THE NODE IS INCLUDED IN AN ALTERNATIVE POSITION

(contd. over)



(f) ADD C_{45}

(g) REMOVE G_{12}

(h) ADD NODE 7

(i) REMOVE C_{23}

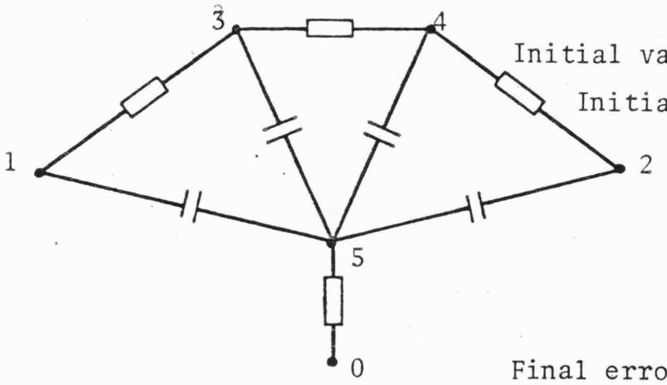
(j) ADD C_{37}

(k) REMOVE C_{17}

Computer realization

FIGURE 6.11: CONTINUED

(a)

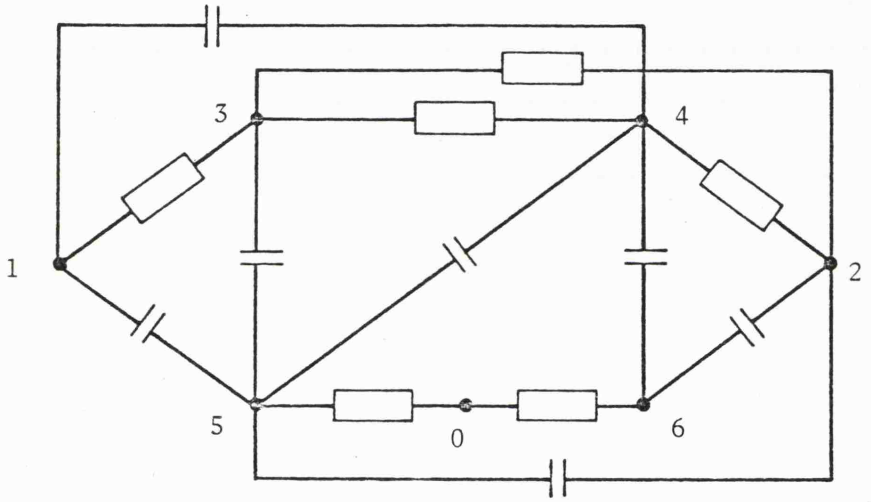


Initial values = 10^0

Initial error = 10.5

Final error = 4.38

(b)	ADD C_{14}	Final error = 1.61
(c)	ADD G_{23}	Final error = 1.61
(d)	ADD NODE 6	

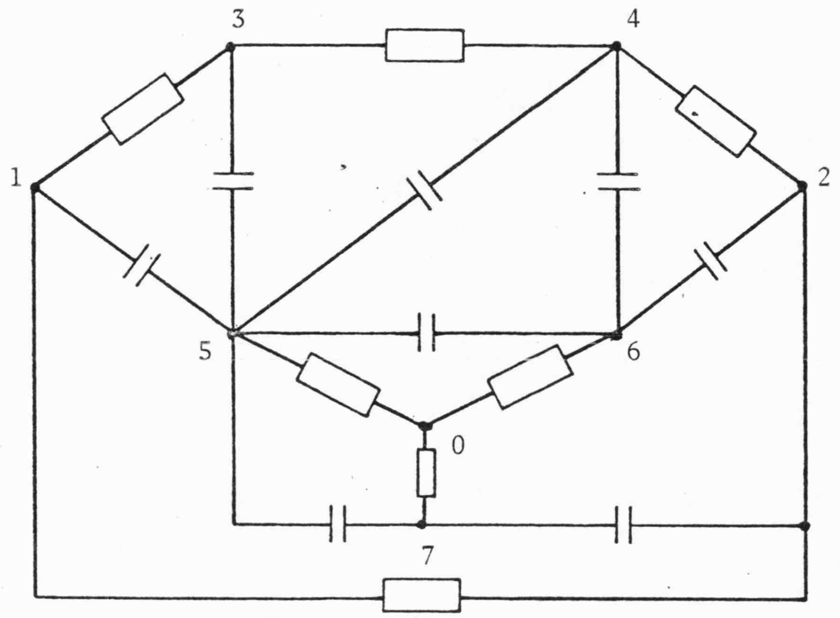


(e)	REMOVE C_{14}	Final error = 4.13×10^{-1}
(f)	ADD C_{56}	
(g)	REMOVE C_{25}	Final error = 3.73×10^{-2}
(h)	ADD G_{12}	
(i)	REMOVE C_{23}	Final error = 2.31×10^{-2}

(j) ADD NODE 7

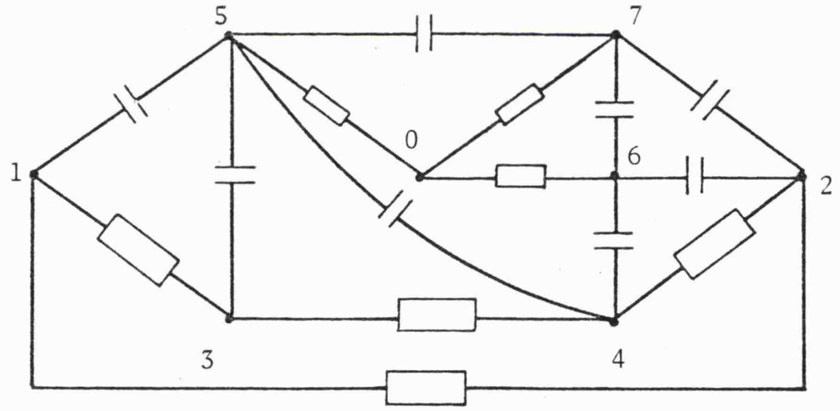
FIGURE 6.12: REALIZATION OF LUCAL'S FUNCTIONS

(Contd. over)



(k) REMOVE C_{56} ↓ Final error* = 2.19×10^{-3}

(1) ADD C_{67} Final error = 2.29×10^{-21}

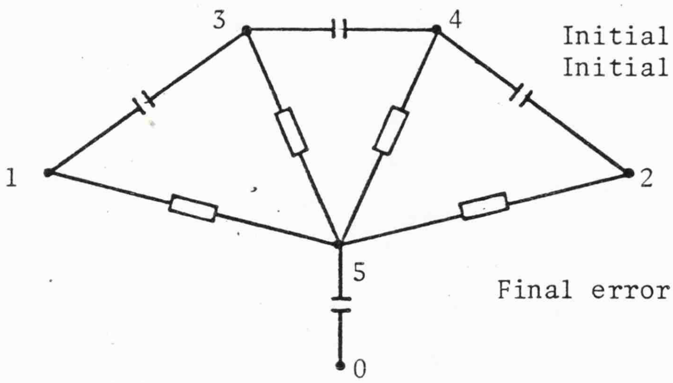


$C_{15} = 21.1$	$G_{13} = 0.226$
$C_{35} = 0.189$	$G_{34} = 0.289$
$C_{45} = 0.675$	$G_{24} = 0.817$
$C_{26} = 4.28$	$G_{05} = 52.7$
$C_{46} = 0.114$	$G_{06} = 10.9$
$C_{27} = 0.259$	$G_{12} = 0.0570$
$C_{57} = 2.57$	$G_{07} = 6.37$
$C_{67} = 1.96$	

Common factors = 3.06 and 1.10

FIGURE 6.12: CONTINUED

(a)



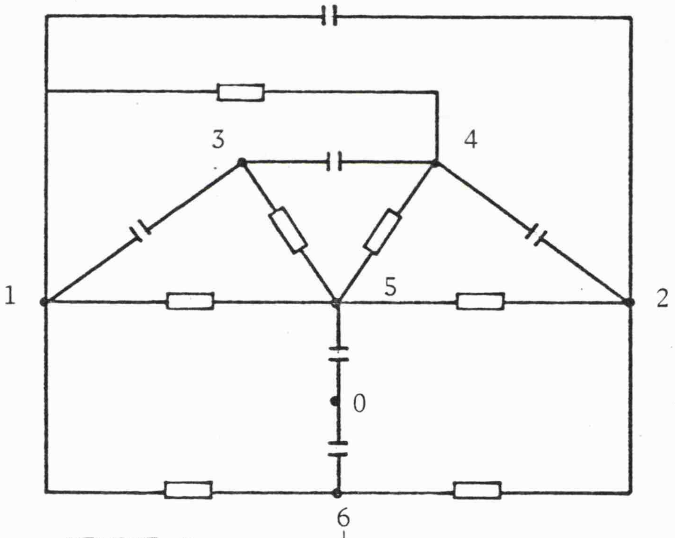
Initial values = 10^0
Initial error = 10.3

Final error = 2.38

(b) ADD C_{12} Final error = 1.22

(c) ADD G_{14} Final error = 1.22

(d) ADD NODE 6



(e) REMOVE G_{14}

(f) REMOVE G_{16} Final error = 1.13

(g) ADD G_{46}

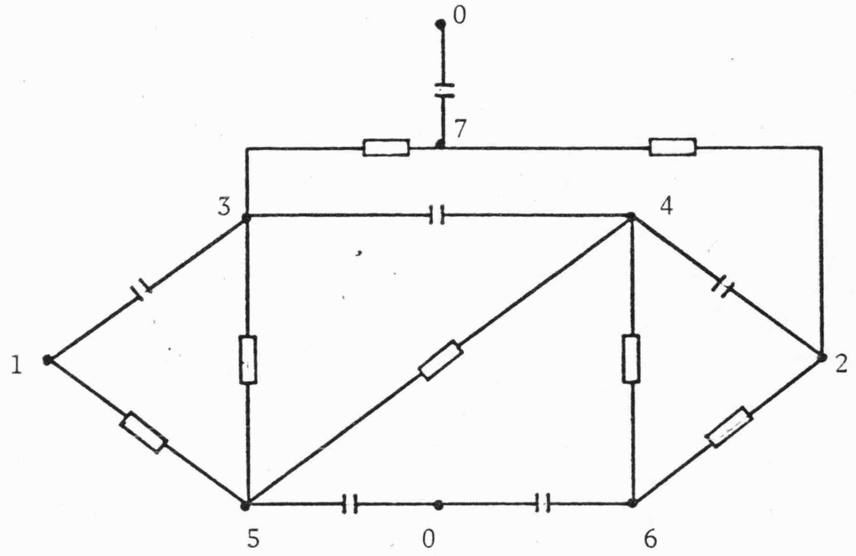
(h) REMOVE C_{12}

(i) REMOVE G_{25} Final error = 0.774

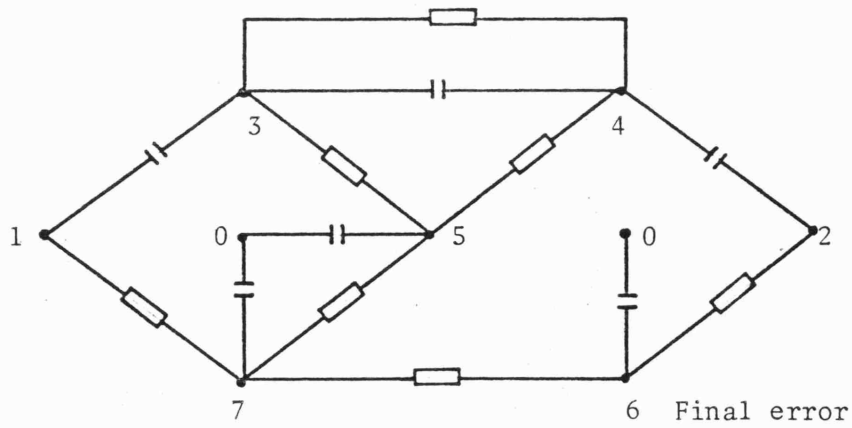
(j) ADD NODE 7

FIGURE 6.13: QUASI REALIZATION OF LUCAL'S FUNCTIONS

(Contd. over)



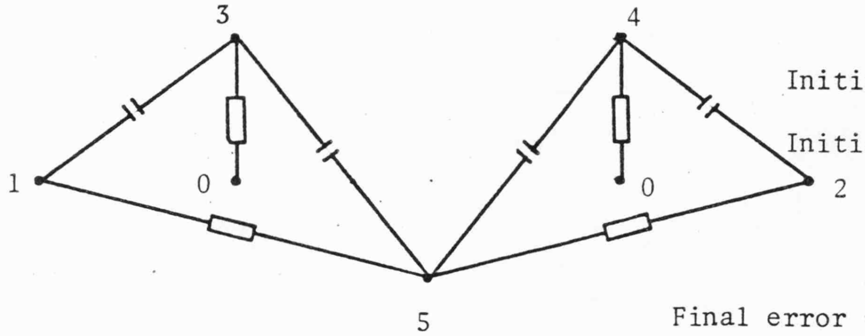
(k)	REMOVE G_{27}	Final error = 0.744
(l)	ADD G_{56}	
(m)	REMOVE G_{46}	Final error = 0.716
(n)	ADD G_{17}	Final error* = 0.455
(o)	ADD G_{67}	
(p)	REMOVE G_{15}	Final error = 1.18×10^{-2}
(q)	REMOVE G_{34}	Final error = 9.26×10^{-3}
(r)	ADD G_{46}	
(s)	REMOVE G_{56}	Final error = 9.25×10^{-3}
(t)	ADD G_{57}	
(u)	REMOVE G_{37}	
(v)	REMOVE G_{46}	



Final error = 9.27×10^{-3}

FIGURE 6.13: CONTINUED

(a)

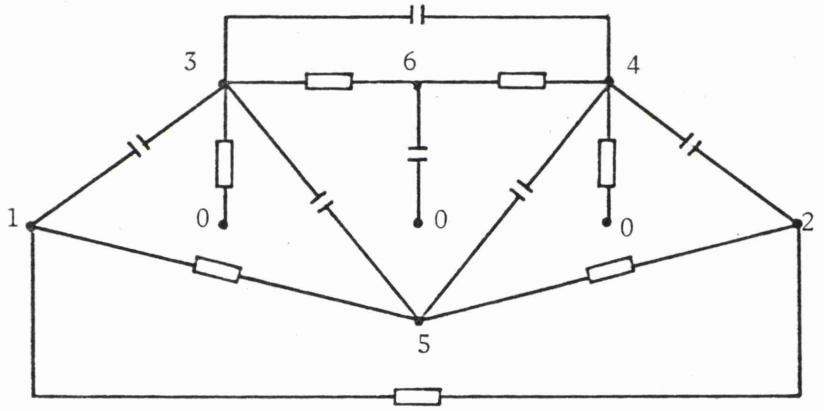


Initial values = 10^0

Initial error = 9.32

Final error = 0.443

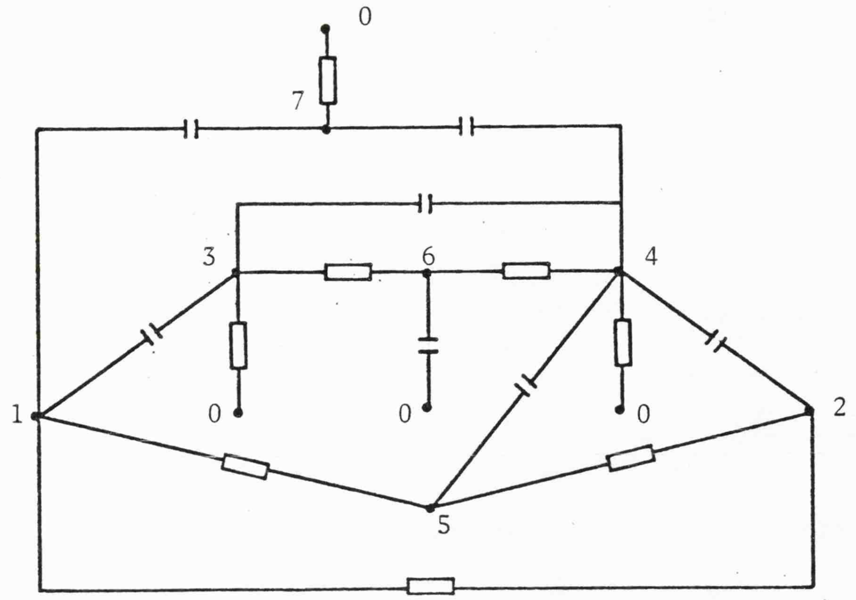
(b)	ADD C_{34}	Final error = 0.128
(c)	ADD C_{15}	Final error = 9.25×10^{-2}
(d)	ADD G_{12}	
(e)	REMOVE C_{15}	Final error = 8.09×10^{-2}
(f)	ADD NODE 6	



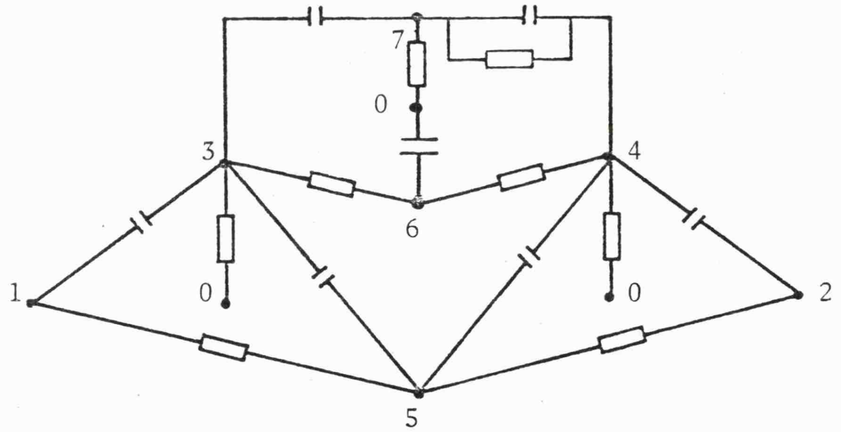
(g)	REMOVE C_{35}	Final error = 2.73×10^{-2}
(h)	ADD NODE 7	

FIGURE 6.14: REALIZATION OF LUCAL'S FUNCTIONS

(Contd.over)



(i)	REMOVE	C_{34}	Final error = 6.65×10^{-4}
(j)	ADD	C_{37}	
(k)	REMOVE	C_{12}	Final error = $4.2l \times 10^{-4}$
(l)	ADD	G_{47}	
(m)	REMOVE	C_{17}	Final error*= 2.48×10^{-4}
(n)	ADD	C_{35}	Final error = 1.65×10^{-22}



C_{13} = 20.8	C_{47} = 2.83	G_{15} = 0.560	G_{36} = 2.04
C_{45} = 0.402	C_{37} = 1.97	G_{25} = 0.237	G_{46} = 1.46
C_{24} = 5.11	C_{35} = 0,262	G_{03} = 52.2	G_{07} = 4.72
C_{06} = 2.51		G_{04} = 9.57	G_{47} = 1.98

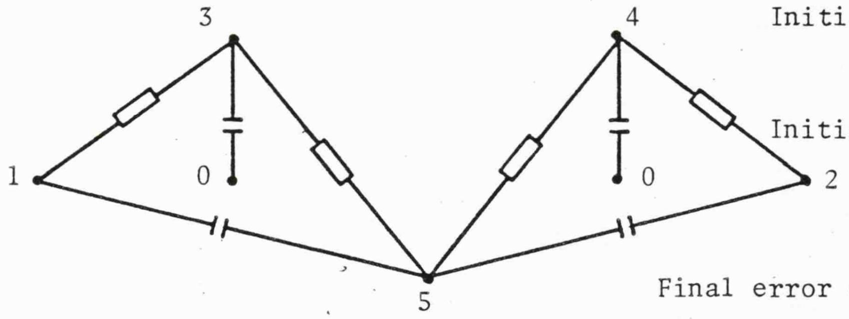
Common factors = 1.39 and 1.24

FIGURE 6.14: CONTINUED

Initial values = 10^{0}

Initial error = 8.74

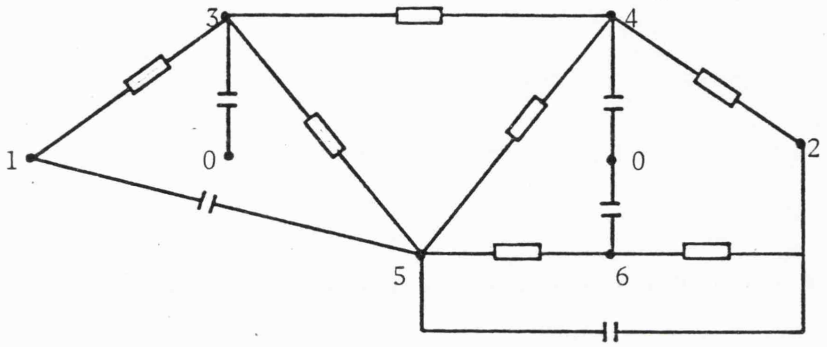
(a)



Final error = 1.33

(b) ADD G_{34} Final error = 1.33

(c) ADD NODE 6

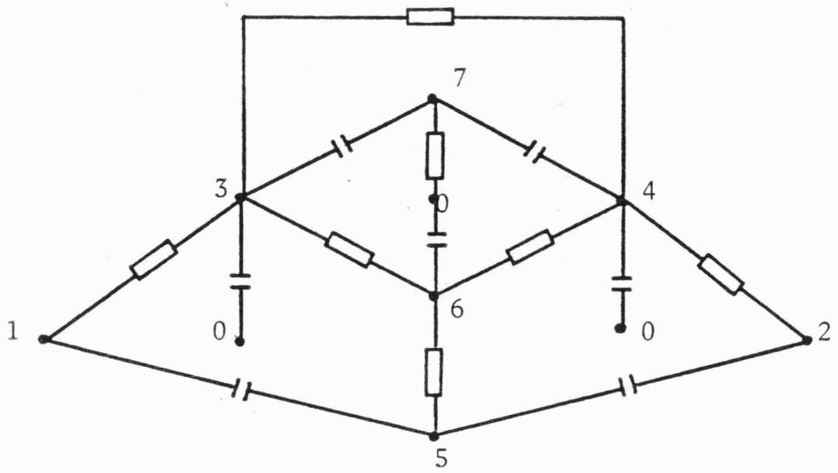


(d) REMOVE G_{34} Final error = 1.01

(e) ADD G_{36}

(f) REMOVE G_{34} Final error = 0.973

(g) ADD NODE 7



Lowest error produced was 0.452.

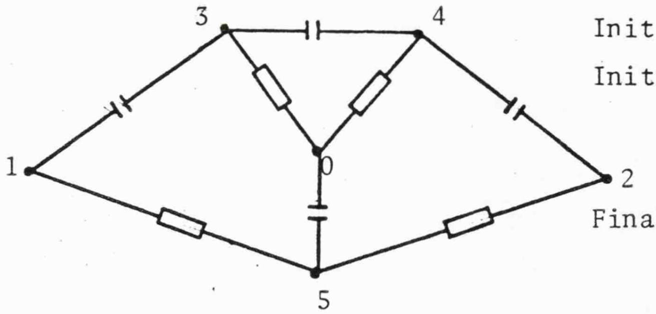
At this stage an alternative node addition was required.

FIGURE 6.15: FAILURE TO REALIZE LUCAL'S FUNCTIONS

Initial values = 10^0

Initial error = 8.65

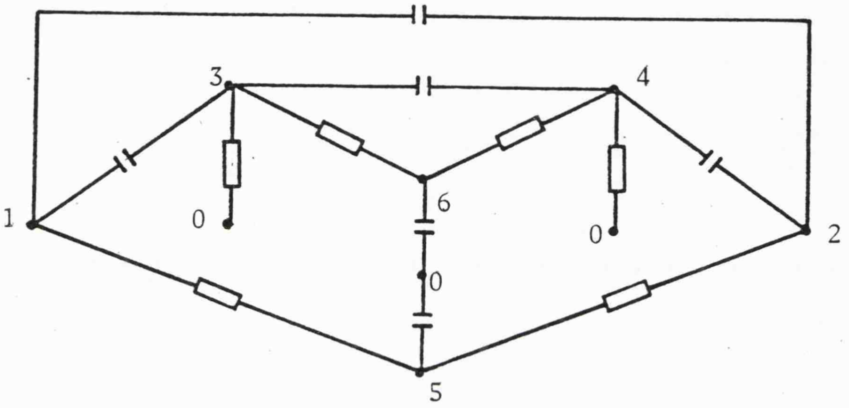
(a)



Final error = 0.819

(b) ADD C_{12} Final error = 0.257

(c) ADD NODE 6

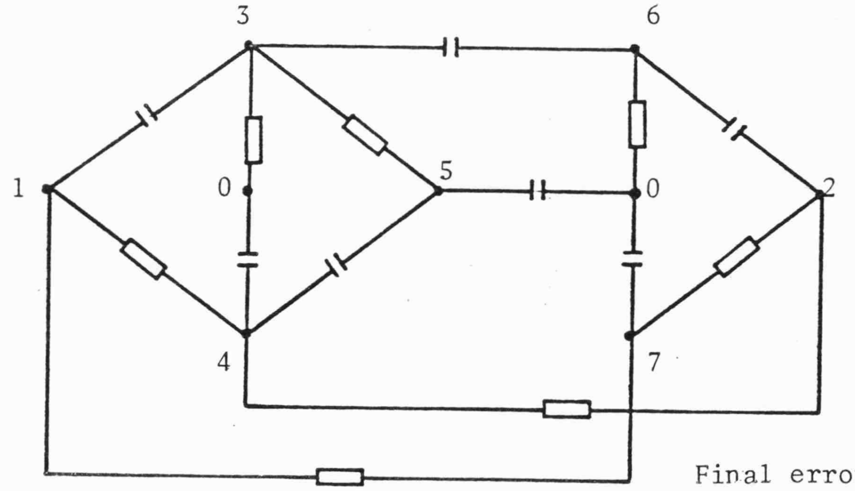


(d) REMOVE C_{12} Final error = 7.77×10^{-3}

(e) ADD C_{56} Final error = 2.45×10^{-4}

(f) REMOVE G_{46} Final error = 2.44×10^{-4}

(g) ADD NODE 7

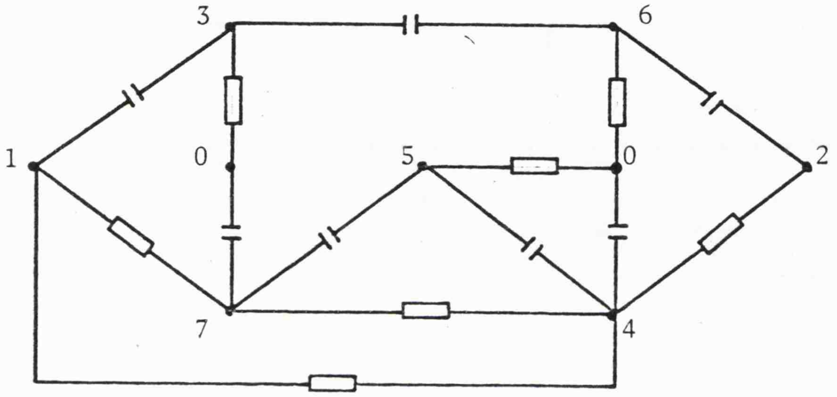


Final error = 6.92×10^{-8}

FIGURE 6.16: REALIZATION OF LUCAL'S FUNCTIONS REQUIRING SIMULTANEOUS ADDITION AND REMOVAL OF ELEMENTS (SECTION 4.12)

(Contd. over)

(h)	ADD G_{47}	
(i)	REMOVE G_{27}	Final error = 6.53×10^{-8}
(j)	ADD C_{57} and	
	REMOVE C_{05}	Final error = 9.85×10^{-9}
(k)	ADD G_{05}	
(l)	REMOVE G_{35}	Computer realization



$C_{13} = 14.0$	$R_{14} = 0.0822$
$C_{04} = 1.64$	$R_{24} = 3.81$
$C_{45} = 0.739$	$R_{03} = 39.2$
$C_{26} = 3.00$	$R_{06} = 10.8$
$C_{36} = 1.68$	$R_{17} = 16.1$
$C_{07} = 6.32$	$R_{47} = 0.0927$
$C_{57} = 0.921$	$R_{05} = 2.10$

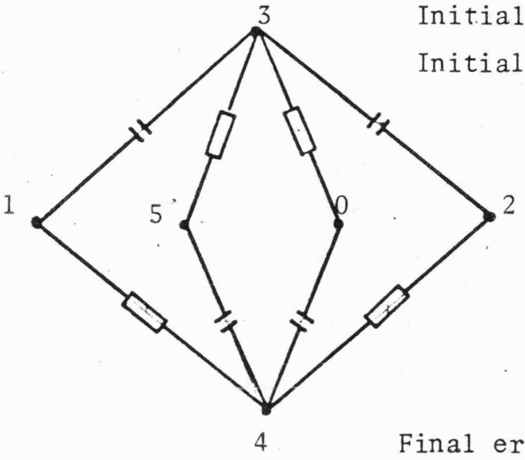
Common factors = 2.00 and 3.00

FIGURE 6.16: CONTINUED

Initial values = 10^0

Initial error = 9.17

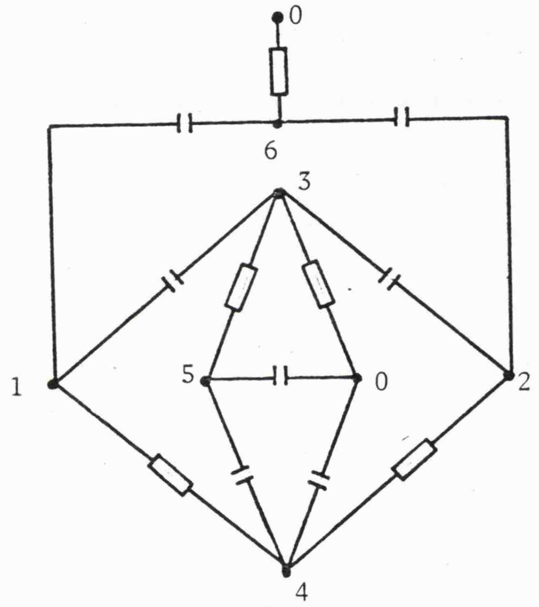
(a)



Final error = 0.951

(b) ADD C_{05} ↓ Final error = 0.912

(c) ADD NODE 6



(d) REMOVE C_{23} Final error* = 9.47×10^{-2}

(e) ADD C_{36}

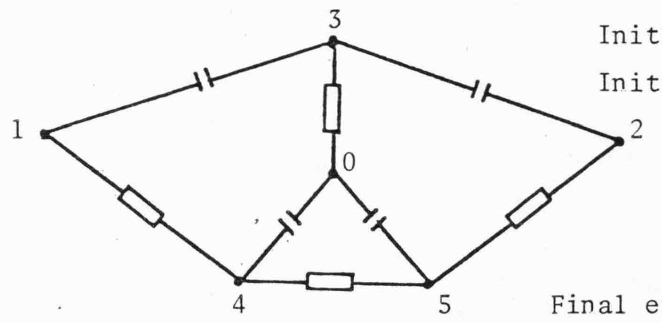
(f) REMOVE C_{16} Final error = 2.44×10^{-4}

(g) ADD NODE 7

Continues as in figure 6.16

FIGURE 6.17: REALIZATION OF LUCAL'S FUNCTIONS REQUIRING SIMULTANEOUS ADDITION AND REMOVAL OF ELEMENTS (SECTION 4.12)

(a)



Initial values = 10^0

Initial error = 8.21

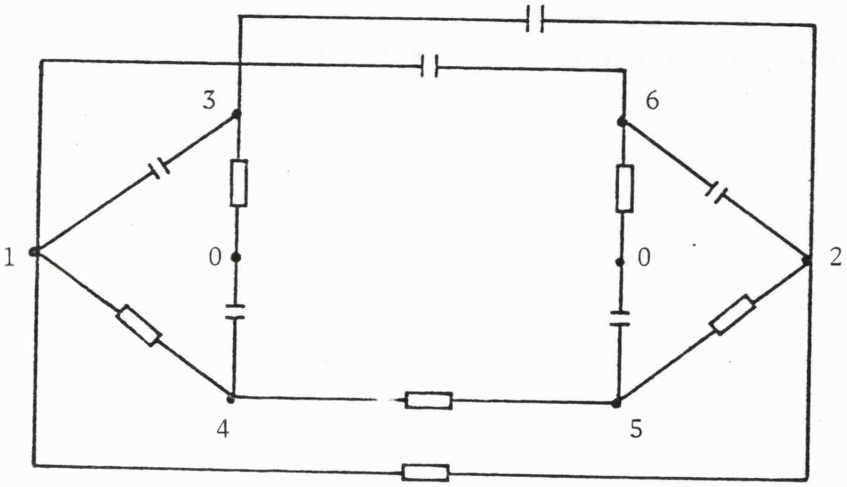
Final error = 1.46

(b) ADD C_{45} — Final error = 0.819

(c) ADD G_{12}

(d) REMOVE C_{45} — Final error = 0.659

(e) ADD NODE 6



(f) REMOVE C_{23} — Final error = 0.182

(g) ADD C_{36} — Final error = 9.72×10^{-2}

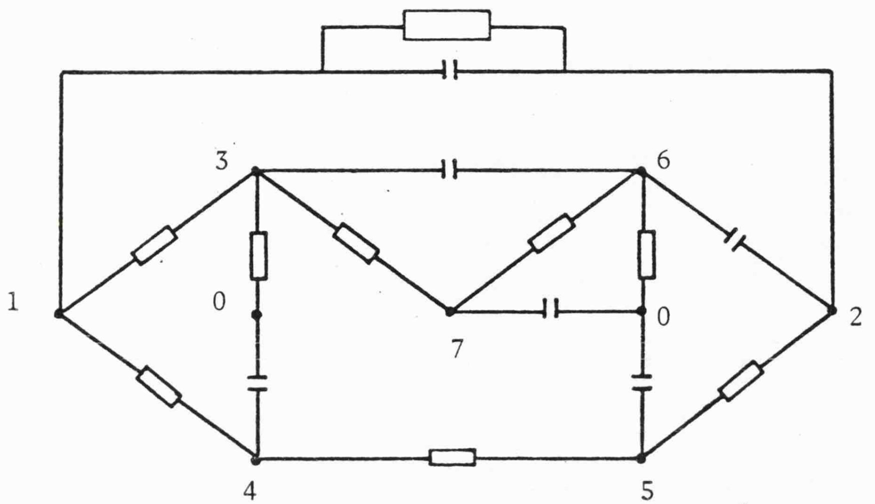
(h) ADD C_{12}

(i) REMOVE C_{16} — Final error = 3.25×10^{-2}

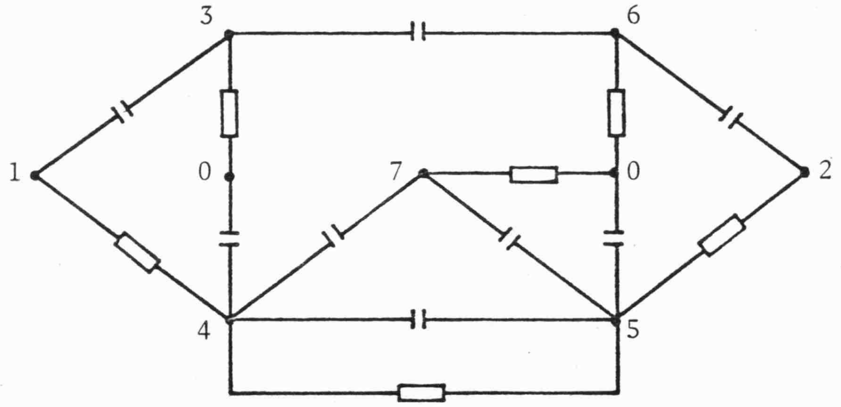
(j) ADD NODE 7

FIGURE 6.18: REALIZATION OF LUCAL'S FUNCTIONS REQUIRING SIMULTANEOUS ADDITION AND REMOVAL OF ELEMENTS (SECTION 4.12)

(Contd. over)

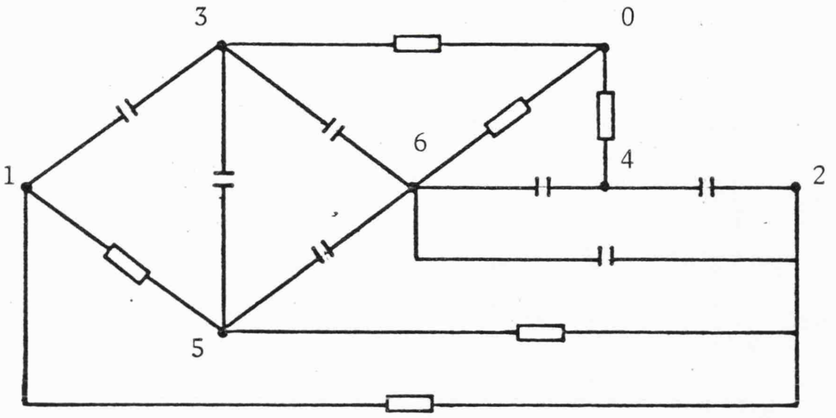


(k)	REMOVE C_{12}		Final error* = 1.05 x 10^{-3}
(l)	ADD C_{57}		
(m)	REMOVE G_{67}		Final error = 1.60 x 10^{-6}
(n)	ADD C_{45}		
(o)	REMOVE G_{12}		Final error = 6.53 x 10^{-8}
(p)	ADD C_{47}	and	
	REMOVE C_{07}		Final error* = 4.45 x 10^{-8}
(q)	ADD G_{07}		
(r)	REMOVE G_{37}		



This is an exact realization as given by Cutteridge[30].

FIGURE 6.18: CONTINUED

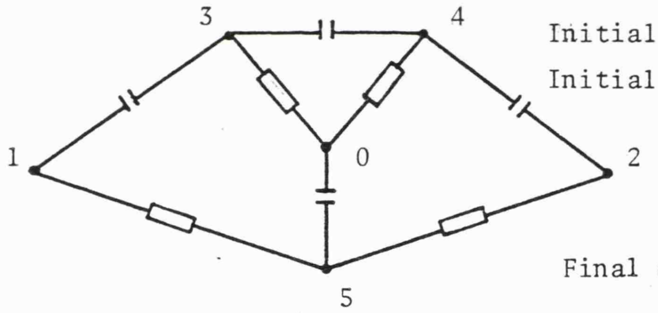


C_{13} = 21.1	C_{26} = 0.232	G_{15} = 0.172	G_{04} = 10.8
C_{35} = 0.854	C_{36} = 2.64	G_{25} = 0.804	G_{12} = 0.0247
C_{24} = 4.29	C_{46} = 2.24	G_{03} = 52.6	G06 = 6.57
	C_{56} = 0.101		

Common factor = 0.986

FIGURE 6.19: REALIZATION OF LUCAL'S FUNCTIONS WITH 6 NODES

(a)



Initial values = 10^0

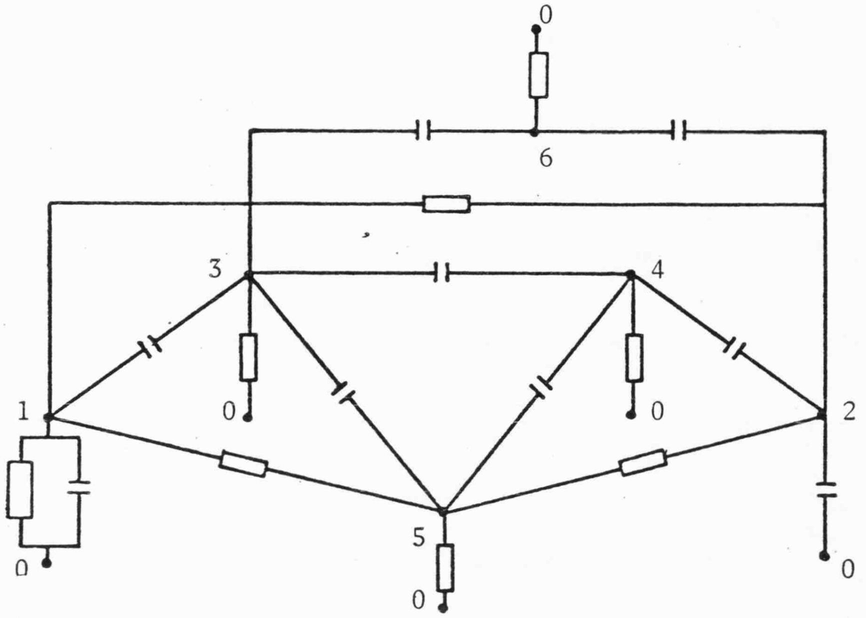
Initial error = 8.66

Final error= 0.819

(b)	ADD	C_{12}	Final error = 0.257
(c)	ADD	C_{35}	Final error = 0.168
(d)	ADD	C_{02}	Final error = 0.140
(e)	ADD	G_{23}	Final error = 0.139
(f)	ADD	G_{14}	Final error = 0.139
(g)	ADD	G_{12}	
(h)	REMOVE	C_{12}	
(i)	REMOVE	G_{14}	Final error = 0.100
(j)	ADD	C_{45}	
(k)	REMOVE	G_{23}	Final error = 7.45 x 10^{-2}
(l)	ADD	C_{01}	
(m)	REMOVE	C_{05}	Final error = 7.03 x 10^{-2}
(n)	ADD	G_{05}	Final error = 7.03 x 10^{-2}
(o)	ADD	G_{01}	Final error = 7.02 x 10^{-2}
(p)	ADD NODE 6		

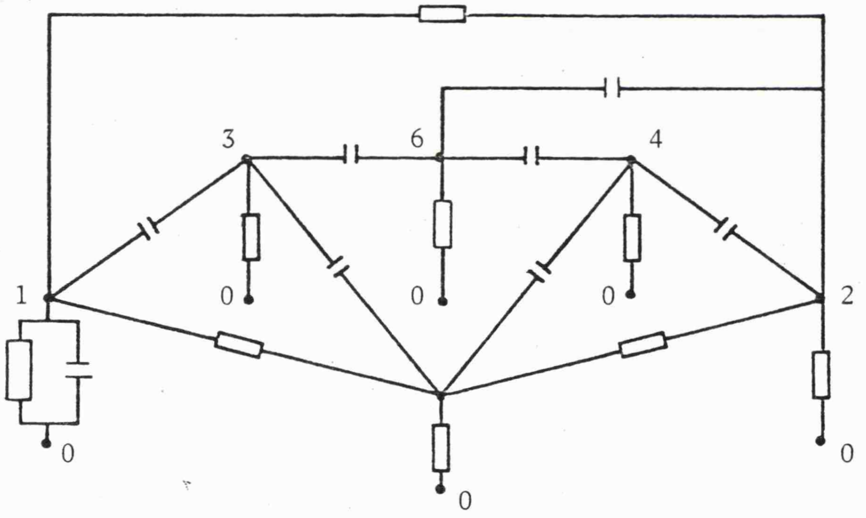
FIGURE 6.20: SIX-NODE REALIZATION OF LUCAL'S FUNCTIONS OBTAINED BY RELAXING THE CONSTRAINTS ON ELEMENT ADDITIONS

(Contd over)



(q) REMOVE C_{34} ↓ Final error = 1.73×10^{-2}

(r) ADD C_{46} computer realization.

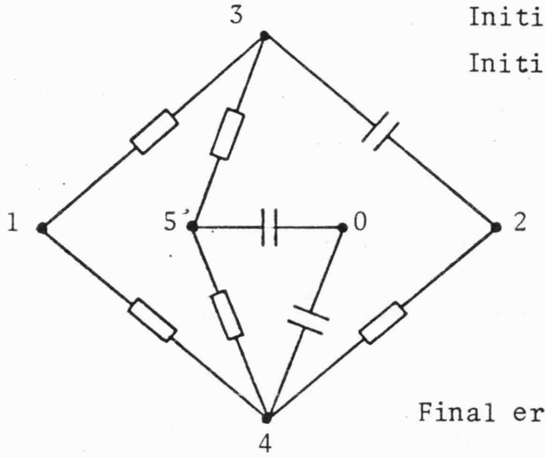


$C_{13} = 21.1$	$C_{45} = 8.68 \times 10^{-2}$	$G_{15} = 0.180$	$G_{12} = 0.0190$
$C_{24} = 4.29$	$C_{01} = 5.03 \times 10^{-11}$	$G_{25} = 0.823$	$G_{05} = 3.08 \times 10^{-12}$
$C_{35} = 0.857$	$C_{26} = 0.245$	$G_{03} = 52.6$	$G_{01} = 2.00 \times 10^{-12}$
$C_{02} = 3.92 \times 10^{-11}$	$C_{36} = 2.63$	$G_{04} = 10.9$	$G_{06} = 6.47$
	$C_{46} = 2.01$		

Common factor = 1.09

FIGURE 6.20: CONTINUED

(a)



Initial values = 10^0

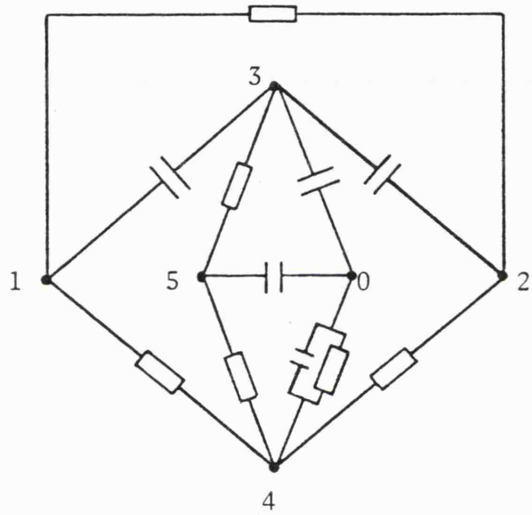
Initial error = 17.7

Final error = 3.56

(b) ADD G_{04} Final error = 2.99

(c) ADD C_{03} Final error = 1.29×10^{-2}

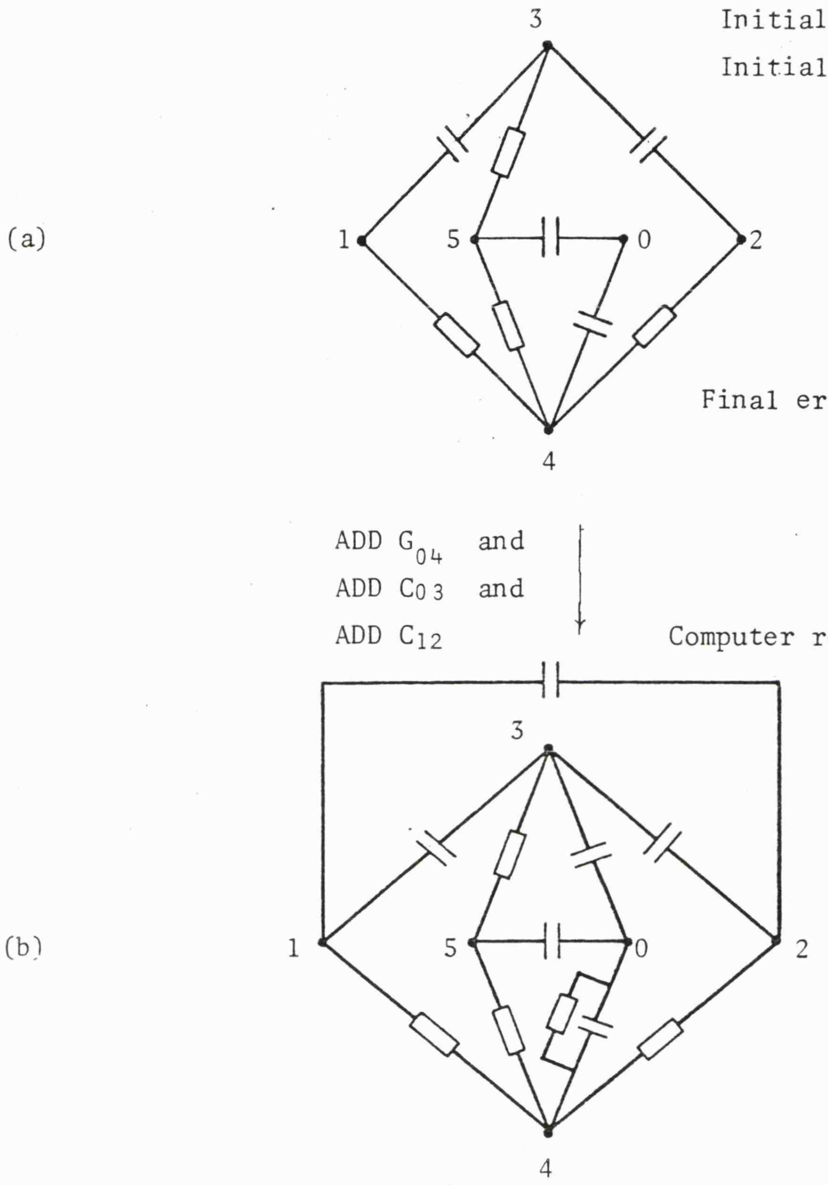
(d) ADD G_{12} computer realization



C_{13}, C_{23}	$= (1.5 \times 10^{-3})$	G_{14}, G_{24}	= 0.0700
C_{03}	= (0.597)	G_{35}	= 0.579
C_{04}	= 20.1	G_{45}	= 2.22
C_{05}	= 4.49	G_{04}	= 0.102
		G_{12}	$= 2.81 \times 10^{-7}$

Common factor = 1.20

FIGURE 6.21: REALIZATION OF FIALKOW'S FUNCTIONS



Initial values = 10^0

Initial error = 17.7

Final error = 3.56

ADD G_{04} and

ADD C_{03} and

ADD C_{12}

Computer realization

C_{13}, C_{23}	$= 1.50 \times 10^{-3}$	G_{14}, G_{24}	= (0.07)
C_{04}	= 20.1	G_{35} =	= 0.650
C_{05}	= 4.44	G_{45}	= 2.24
C_{03}	= 0.658	G_{04}	= 0.102
C_{12}	$= 3.52 \times 10^{-7}$		

Common factor = 1.24

FIGURE 6.22: REALIZATION OF FIALKOW'S FUNCTIONS BY THE SIMULTANEOUS ADDITION OF THREE ELEMENTS (SECTION 4.7)

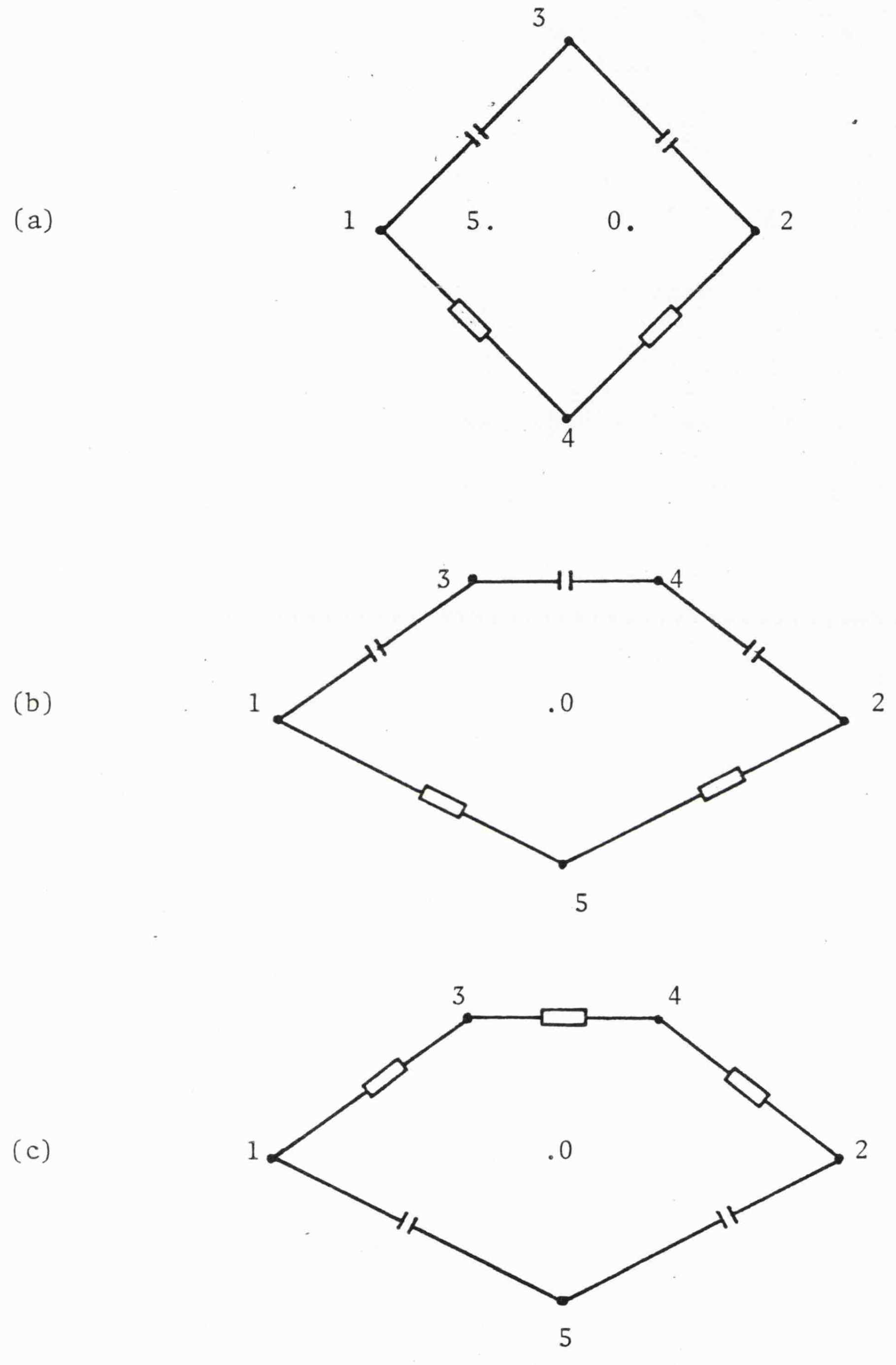
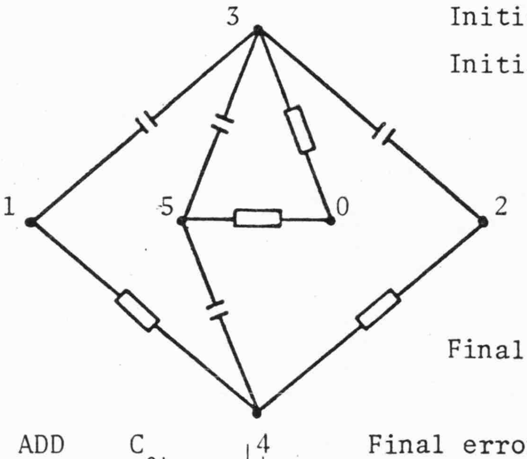


FIGURE 6.23: ALTERNATIVE NETWORK STRUCTURES

(a)

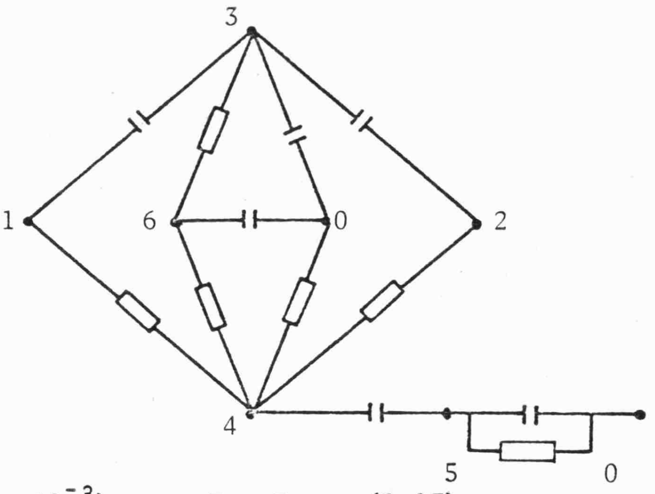


Initial values = 10^0

Initial error = 16.8

Final error = 3.62

(b)	ADD C_{04}	Final error = 0.590
(c)	ADD G_{04}	Final error = 1.03×10^{-3}
(d)	ADD C_{05}	
(e)	REMOVE C_{04}	Final error = 2.74×10^{-5}
(f)	ADD C_{03}	
(g)	REMOVE C_{35}	Final error = 4.99×10^{-6}
(h)	ADD NODE 6	
(i)	REMOVE G_{03}	Computer realization



C_{13}, C_{23}	= (1.5×10^{-3})	G_{14}, G_{24}	= (0.07)
C_{45}	= 22.3	G_{04}	= 0.102
C_{05}	= 198	G_{05}	= 87.6
C_{03}	= (0.597)	G_{36}	= 0.534
C_{06}	= 2.25	G_{46}	= 1.36

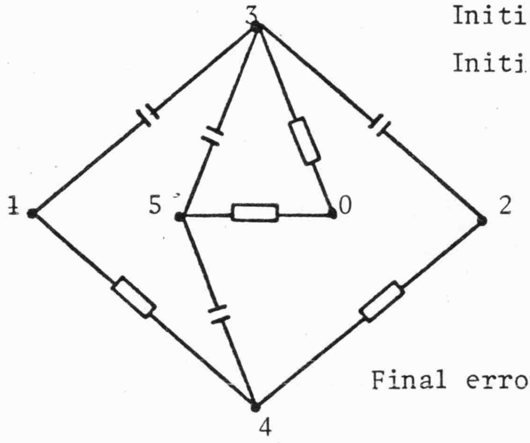
Common factors = 1.34 and 0.401

FIGURE 6.24: SIX-NODE RC REALIZATION OF FIALKOW'S FUNCTIONS

Initial values = 10^0

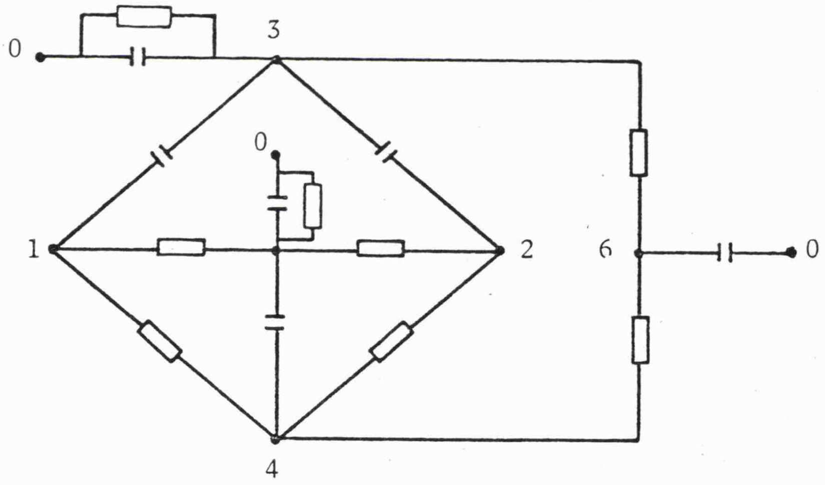
Initial error = 17.8

(a)



Final error = 3.62

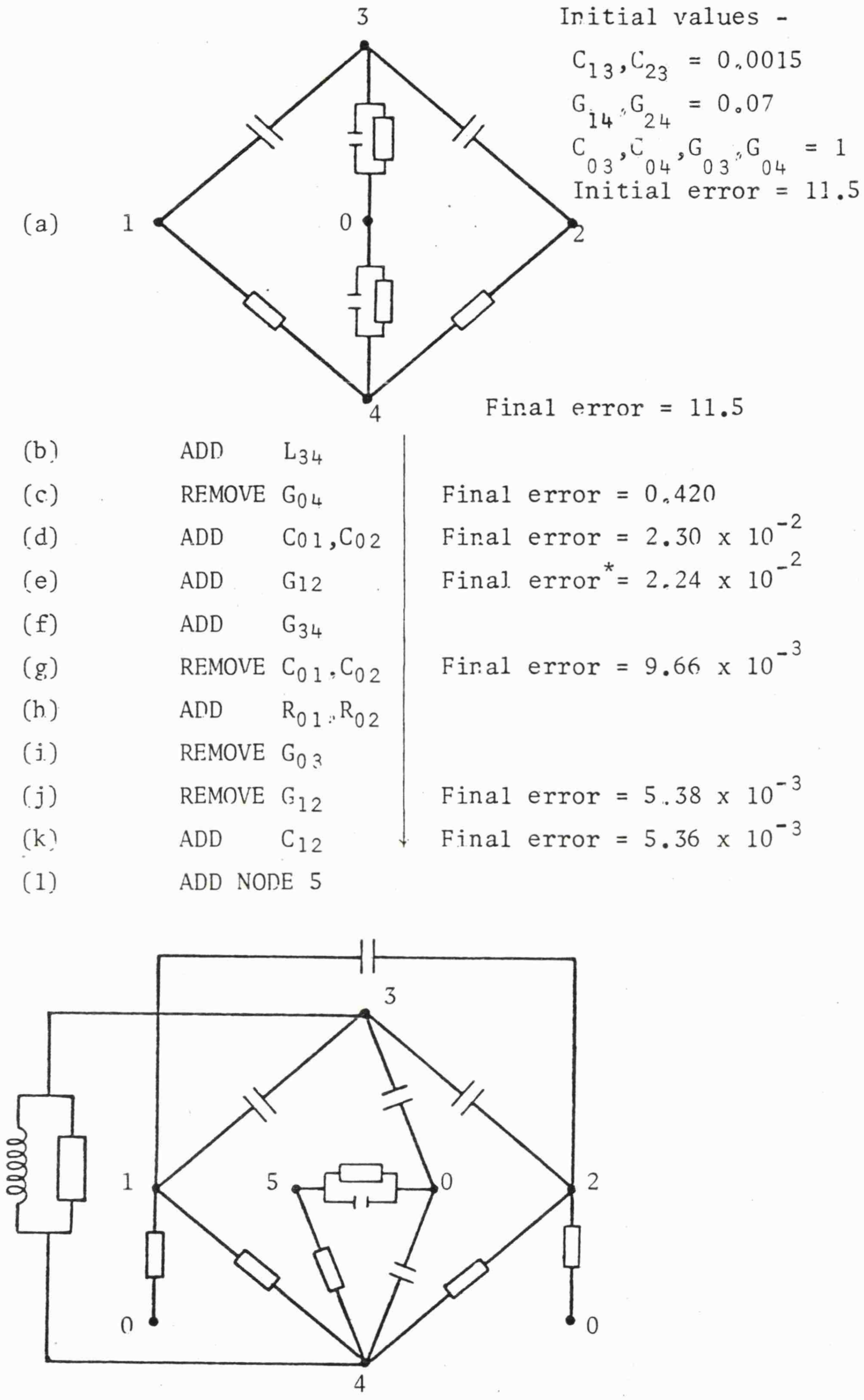
(b)	ADD C_{05}	Final error = 1.29
(c)	ADD R_{01}, R_{02}	Final error = 7.09×10^{-5}
(d)	ADD R_{15}, R_{25}	
(e)	REMOVE R_{01}, R_{02}	Final error = 2.82×10^{-5}
(f)	ADD C_{03}	
(g)	REMOVE C_{35}	Final error = 4.99×10^{-6}
(h)	ADD NODE 6	
(i)	REMOVE G_{03}	Computer realization



$C_{13} = C_{23}$	$= (1.5 \times 10^{-3})$	G_{14}, G_{24}	$= 4.04 \times 10^{-2}$
C_{45}	$= 7.02$	G_{05}	$= 215$
C_{05}	$= 477$	G_{15}, G_{25}	$= 2.96 \times 10^{-2}$
C_{03}	$= (0.597)$	G_{36}	$= (0.534)$
C_{06}	$= 0.772$	G_{46}	$= 0.788$

Common factors = 0.442 and 2.23

FIGURE 6.25: SIX-NODE RC REALIZATION OF FIALKOW'S FUNCTIONS



Initial values -

$C_{13}, C_{23} = 0.0015$

$G_{14}, G_{24} = 0.07$

$C_{03}, C_{04}, G_{03}, G_{04} = 1$

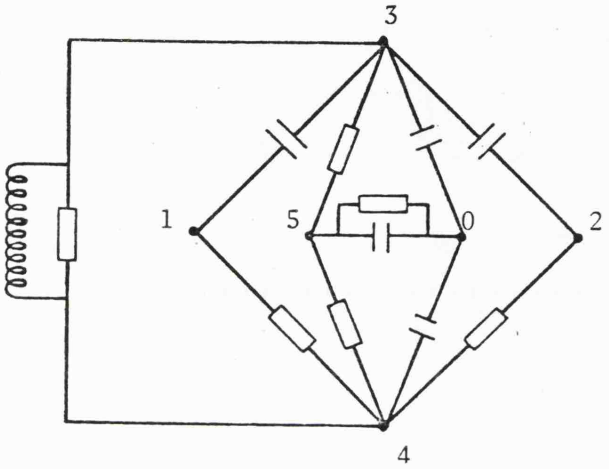
Initial error = 11.5

Final error = 11.5

(b)	ADD	L_{34}	
(c)	REMOVE	G_{04}	Final error = 0.420
(d)	ADD	C_{01}, C_{02}	Final error = 2.30×10^{-2}
(e)	ADD	G_{12}	Final error* = 2.24×10^{-2}
(f)	ADD	G_{34}	
(g)	REMOVE	C_{01}, C_{02}	Final error = 9.66×10^{-3}
(h)	ADD	R_{01}, R_{02}	
(i)	REMOVE	G_{03}	
(j)	REMOVE	G_{12}	Final error = 5.38×10^{-3}
(k)	ADD	C_{12}	Final error = 5.36×10^{-3}
(l)	ADD NODE 5		

FIGURE 6.26: RLC REALIZATION OF FIALKOW'S FUNCTIONS INCLUDING THE ADDITION OF A NODE

(m)	REMOVE C_{12}	
(n)	REMOVE G_{01},G_{02}	Final error = 1.19×10^{-5}
(o)	ADD G_{35}	Final error = 1.70×10^{-20}

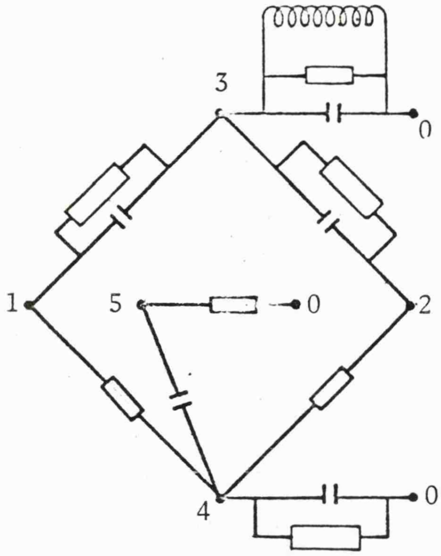


C_{13},C_{23}	= (1.5×10^{-3})	G_{14},G_{24}	= 0.070
C_{03}	= (0.597)	G_{34}	= 0.483
C_{04}	= 20.6	G_{05}	= 0.111
C_{05}	= 4.60	G_{45}	= 1.32
		G_{35}	= 5.15×10^{-2}

L_{34} = 0.121

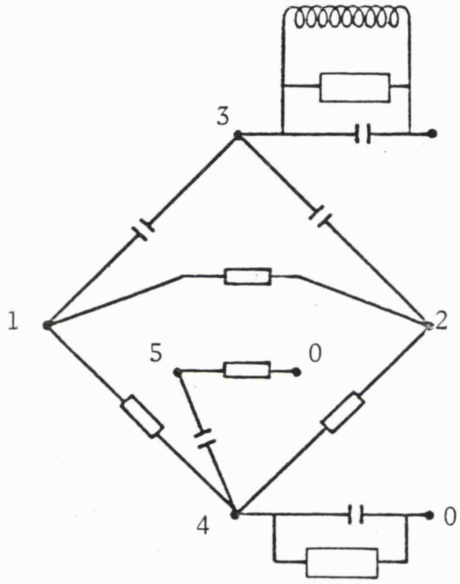
Common factor = 0.398 and 0.398

FIGURE 6.26: CONTINUED



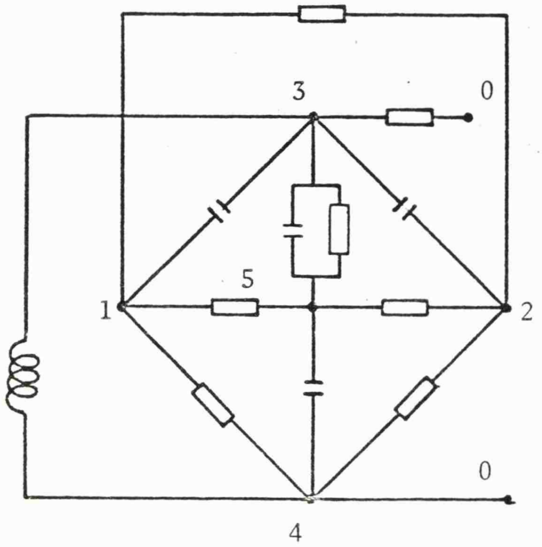
$C_{13}, C_{23} = (1.5 \times 10^{-3})$
$C_{04} = 19.9$
$C_{03} = (0.597)$
$C_{45} = 5.19$
$G_{14}, G_{24} = 0.0700$
$G_{03} = 0.596$
$G_{04} = 0.102$
$G_{05} = 2.02$
$G_{13}, G_{23} = 7.80 \times 10^{-5}$
$L_{03} = 0.148$
Common factors = 0.494 and 0.494

FIGURE 6.27: RLC REALIZATION OF FIALKOW'S FUNCTIONS HAVING A SERIES-PARALLEL STRUCTURE



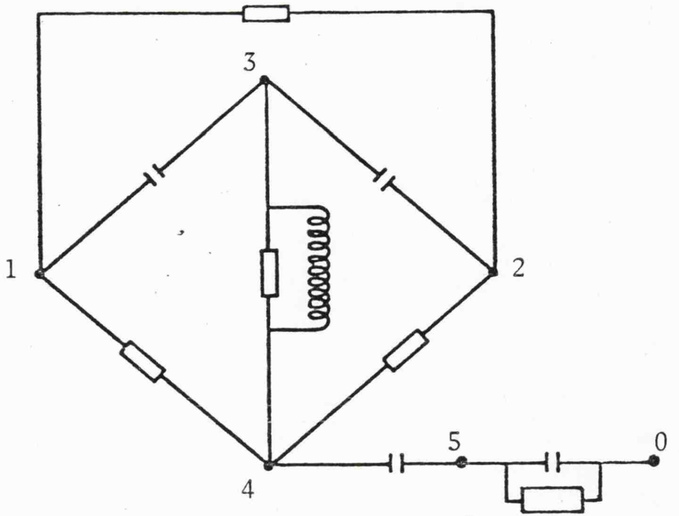
$C_{13}, C_{23} = (1.5 \times 10^{-3})$
$C_{04} = 20.0$
$C_{03} = (0.597)$
$C_{45} = 5.19$
$G_{14}, G_{24} = 0.0700$
$G_{03} = 0.595$
$G_{04} = 0.102$
$G_{05} = 2.01$
$G_{12} = 3.84 \times 10^{-7}$
$L_{03} = 0.148$
Common factors = 0.492 and 0.492

FIGURE 6.28: RLC REALIZATION OF FIALKOW'S FUNCTIONS HAVING A SERIES-PARALLEL STRUCTURE



$C_{13}, C_{23} = (7.5 \times 10^{-6})$
$C_{45} = 8.41$
$C_{05} = 95.5$
$G_{14}, G_{24} = 4.04 \times 10^{-2}$
$G_{05} = 46.8$
$G_{03} = 1.49 \times 10^{-5}$
$G_{15}, G_{25} = 2.96 \times 10^{-2}$
$G_{12} = 3.96 \times 10^{-7}$
$L_{34} = 3.72 \times 10^{-6}$
Common factors = 0.494 and 0.494

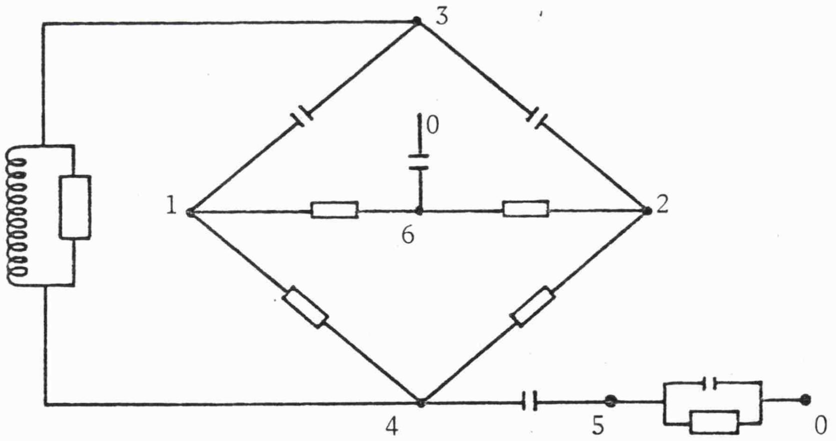
FIGURE 6.29: RLC REALIZATION OF FUNCTIONS 6.16



C_{13},C_{23}	$= (7.05 \times 10^{-6})$	G_{14},G_{24}	$= 4.04 \times 10^{-2}$	G_{12}	$= 3.63 \times 10^{-7}$
C_{45}	$= 8.38$	G_{05}	$= 15.8$	L_{34}	$= 3.65 \times 10^{-6}$
C_{05}	$= 32.4$	G_{34}	$= 1.48 \times 10^{-5}$		

Common factor = 0.489 and 0.489

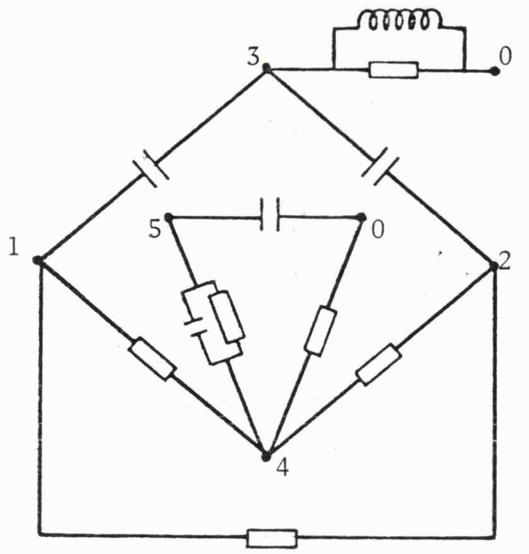
FIGURE 6.30: RLC REALIZATION OF FUNCTIONS 6.17



C_{13},C_{23}	$= (2.5 \times 10^{-4})$	G_{14},G_{24}	$= 4.03 \times 10^{-2}$	L_{34}	$= 3.92 \times 10^{-6}$
C_{45}	$= 8.38$	G_{34}	$= 2.575 \times 10^{-4}$		
C_{05}	$= 77.8$	G_{05}	$= 38.7$		
C_{06}	$= 1.07 \times 10^{-4}$	G_{16},G_{26}	$= 8.39 \times 10^{-7}$		

Common factor = $p^3 + 0.529\ p^2 + 0.0159\ p + 1.22 \times 10^{-4}$

FIGURE 6.31: RLC REALIZATION OF FUNCTIONS 6.18



$C_{13}, C_{23} = (7.5 \times 10^{-6})$

$C_{45} = 97.0$

$C_{05} = 25.1$

$G_{14}, G_{24} = 0.0700$

$G_{45} = 47.2$

$G_{03} = 1.49 \times 10^{-5}$

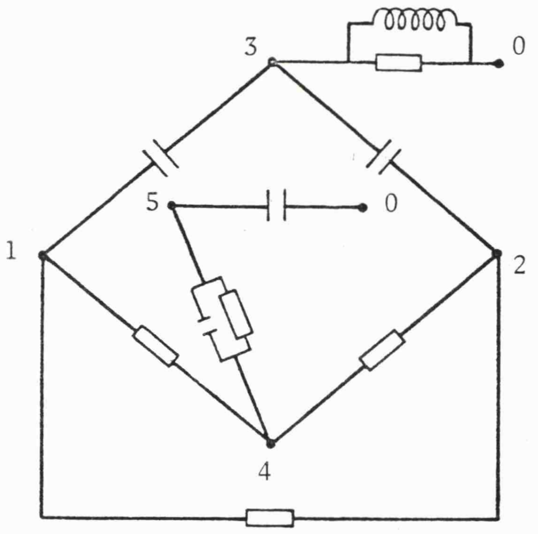
$G_{04} = 0.102$

$G_{12} = 3.84 \times 10^{-7}$

$L_{03} = 3.69 \times 10^{-6}$

Common factors = 0.492 and 0.492

FIGURE 6.32: RLC SERIES-PARALLEL REALIZATION OF FUNCTIONS 6.16



$C_{13}. C_{23} = (7.5 \times 10^{-6})$

$C_{45} = 32.4$

$C_{05} = 8.38$

$G_{14}; G_{24} = 4.04 \times 10^{-2}$

$G_{45} = 15.8$

$G_{03} = 1.49 \times 10^{-5}$

$G_{12} = 3.84 \times 10^{-7}$

$L_{03} = 3.69 \times 10^{-6}$

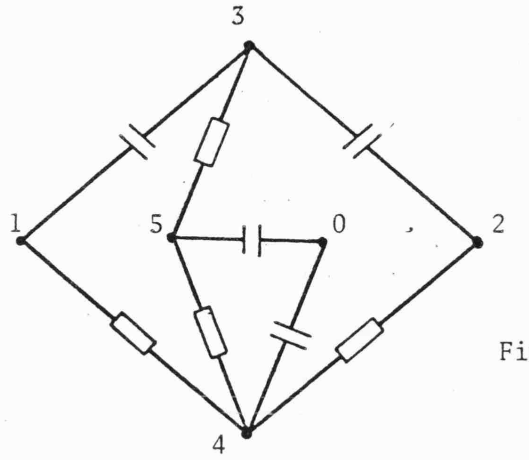
Common factors = 0.492 and 0.492

FIGURE 6.33: RLC SERIES-PARALLEL REALIZATION OF FUNCTIONS 6.17

Initial values = 10^0

Initial error = 17.7

(a)

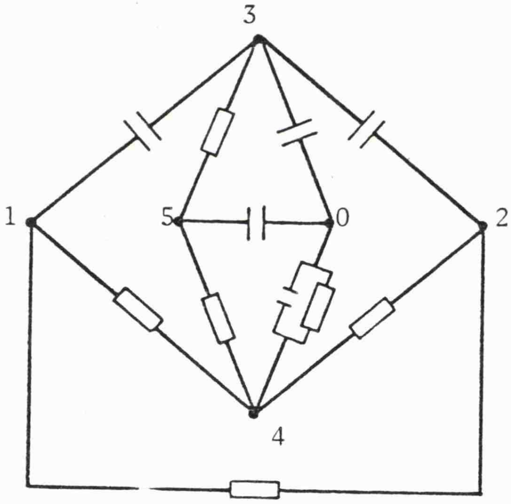


Final error = 3.57

(b) ADD G_{04} Final error = 2.99

(c) ADD C_{03} Final error = 7.65×10^{-3}

(d) ADD G_{12}

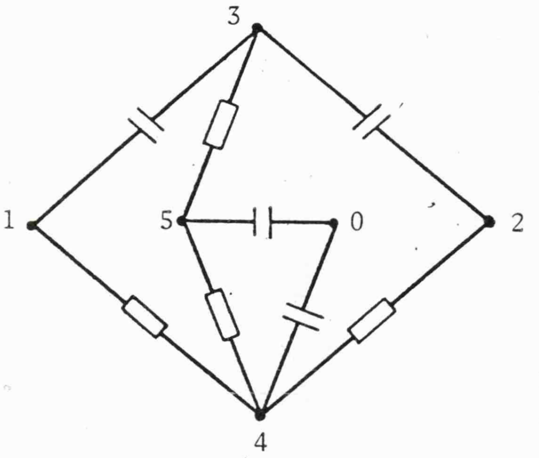


C_{13}, C_{23}	= (1.5×10^{-3})	G_{14}, G_{24}	= 0.0700
C_{04}	= 20.1	G_{35}	= 0.570
C_{05}	= 3.48	G_{45}	= 1.91
C_{03}	= (0.597)	G_{04}	= 0.102
		G_{12}	= 2.23×10^{-7}

Common factor = 1.26

FIGURE 6.34: REALIZATION OF FUNCTIONS 6.20

(a)



Initial values = 10^0

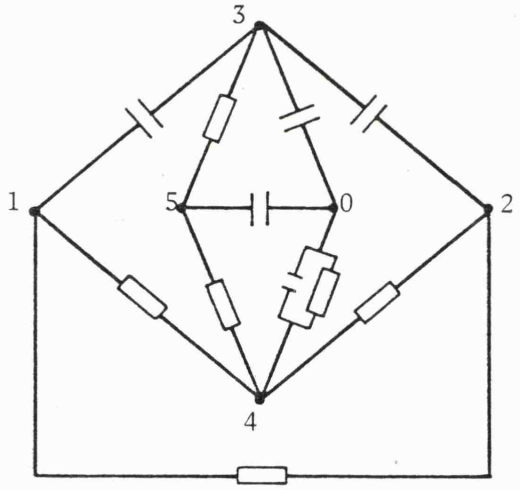
Initial error = 17.7

Final error = 3.58

(b) ADD G_{04} Final error = 2.99

(c) ADD C_{03} Final error = 5.46×10^{-3}

(d) ADD G_{12} Final error = 1.99×10^{-24}



C_{13}, C_{23}	= (2.475×10^{-3})	G_{14}, G_{24}	= 6.95×10^{-2}
C_{04}	= 19.9	G_{35}	= 1.59
C_{05}	= 3.28	G_{45}	= 2.48
C_{03}	= 1.63	G_{04}	= 0.100
		G_{12}	= 3.03×10^{-7}

Common factor = 1.84

FIGURE 6.35: REALIZATION OF FUNCTIONS 6.21

CHAPTER 7

CONCLUSIONS AND POSSIBILITIES FOR FURTHER RESEARCH

The major conclusion of this research is that it is possible, at least with networks of the order considered, to evolve realizations from a minimal starting topology. Realizations requiring the addition of one or two nodes were repeatedly obtained to several test problems. (Although element addition had previously been achieved by several researchers[13,14,15,16,17] on only one instance had a node been included to successfully produce a realization[14]. Furthermore, the author found this method to be ineffective when applied to other test examples.) With RC networks, realizations with seven nodes (plus earth) and fifteen elements evolved from minimal structures of five nodes and eight elements, requiring approximately twenty-five topological modifications. Although similar RLC networks could not be located, RLC realizations with a similar number of elements, but with only six nodes, were obtained for other problems from similar starting topologies. This would seem to indicate possibilities beyond the reduction of networks synthesized by standard methods (e.g. artificial transmission lines) as suggested by di Mambro[13].

The principal achievements of the research were to:

(1) improve the efficiency of the optimization (sections 3.6 and 3.7),

(2) modify the criteria for the recognition of the desirability of element removal (section 3.8),

(3) include criteria to recognize when node removal is indicated (section 3.9),

(4) develop criteria to determine the location of a minimum of the overall error function (section 3.10),

(5) develop an improved method for the introduction of elements (section 4.4),

(6) develop a successful method for the introduction of groups of elements, i.e. node addition (section 5.8).

These improved techniques, allied to their improved utilization (by ensuring the coefficients are matched effectively and by including a restart facility), produced three networks of theoretical significance:

(i) an RLC series-parallel realization of Fialkow's functions (section 6.8),

(ii) RLC realizations of a network containing a negative numerator coefficient in the short-circuit transfer admittance function after the removal of common factors, all poles of both impedance and admittance being compact (section 6.9),

(iii) An RC realization of Lucal's functions with only six nodes plus the reference node (section 6.6).

Whilst a deal of success has been achieved with the composite strategy developed, none of the techniques currently incorporated in the program is necessarily to be regarded as beyond improvement. All could be improved, either as a unique change or, as one of a series of changes to produce a different strategy. This fact is illustrated by the need for recourse to a restart strategy after some topological additions. Consider for example the question of node addition. An addition may be said to fail in several manners:

(i) an attempted introduction may not be feasible as the method has only an empirical and intuitive basis,

(ii) the node may be introduced at values which result in the rejection of the addition when it is, in fact, feasible,

(iii) one node may be included in preference to others which would have produced better results,

(iv) on isolated instances, the method will fail to include a node in any of the potential positions.

The methods used in this research assume that an approximation stage has been accomplished, i.e. the requirements have been transformed into the short-circuit admittance parameters. These parameters are rational functions in p , the complex frequency variable, and the coefficients are equated in the design process. The relative merits of the alternatives to coefficient matching are well documented in the literature and the salient points have been restated in this thesis. Many feel, because of the mathematical basis of coefficient matching, that if computer aided design is to be achieved it will be by this, rather than an alternative method. (This mathematical aspect also facilitates the adoption of automated, rather than interactive, design.) Perhaps a re-evaluation of these alternatives, incorporating some of the developments beneficial in design based on coefficient matching, would be enlightening. (The author will discuss, at a later stage, why the coefficient matching design may be insufficient in itself.)

The behaviour of any optimization routine will obviously be dependent on the accuracy of the analysis. (Several jobs run by the author at the latter stages of the research were successful only when double precision was employed.) For the magnitude of the networks considered, the accuracy of the analysis was not a limiting factor in the performance of the package. Consequently, the author conducted little research into the analysis routine. In fact it is evident that there are several modifications which could be implemented that would increase the accuracy without recourse to an alternative method.

Worthwhile research could be undertaken into an evaluation of the accuracy produced by this method[13,21-24], and improved versions, to establish the practical limitations of size, element disparities etc. Similarly a more extensive evaluation of the fast Fourier transform method[25,26] could be undertaken.

The less direct adjoint method[48-51] of calculating the derivatives, which determines the response to real frequencies, appears to be preferred merely because it is 'in vogue'. An accurate comparison with the method devised at Leicester should be undertaken.

The power of the optimization routines contained in the original version of the program was improved in two ways. Firstly, by modifying the criteria that determine which algorithm should be used and, secondly, by improving the linear searches of the two algorithms, thus ensuring a continued reduction of the error. An alternative second order gradient descent algorithm[40-42] was examined but was found to be superfluous to the requirements of an idealized RLC network. Further, it is unlikely, because of the increased computational effort required as the number of variables is increased, that this method will be viable when applied to larger networks. The limitations of size (etc.) on this combined CG/GN algorithm were not fully explored and research in this direction would be beneficial. The largest networks tackled by the author had ten nodes and twenty RC elements and variable common factors.

A further related topic for research could be an investigation of the effects of constraints on the element values, a reasonable practical requirement. Similarly, underdefined problems could also be examined. These underdefined problems could arise in two foreseeable circumstances. For several reasons, it can prove beneficial to commence designs from the smallest possible structures. However, the degree of difficulty experienced and time taken in achieving a realization will be dependent on the proximity of the network to a suitable topology. The author had insufficient time to examine whether it is reasonable to start with a network with the minimum number of nodes but a large number of elements, which are consequently underdefined, and use a suitable optimization technique to modify the values. It would, perhaps, be possible to remove redundant elements, by say, their sensitivities, and hence commence the conventional design from the suitable remanent network.

Alternatively, with all the examples considered in this thesis, the networks were defined by three short-circuit admittance parameters and, hence, four cofactors. A situation could arise where the design requirements are so flexible that only one parameter, and hence two cofactors, are necessary to define the network, as in Calahan's[9] original formulation of a coefficient matching problem. Although this intrinsically reduces the difficulty of producing a solution, it may lead to an underdefined problem.

Cutteridge[15] developed an algorithm which removes elements prior to their reduction to a negligible value. This algorithm was modified to increase its efficiency when applied to networks with more components. Similarly, the criteria for determining when element addition was required were also improved. The determination of a minimum of the overall error function is important not only because it intimates that the current network is unlikely to satisfy the requirements but, also, the element addition algorithms work more efficiently from the actual minima.

Chapter 4 was devoted to element addition. The original package included a method of element addition[33] but this had limitations which increased as the error was reduced. An improved method was developed, based on the GN algorithm. However, some problems are already apparent with this method at low errors. A fuller appraisal of the range of applicability is necessary. (Elements have, so far, been successfully included to RC networks with ten nodes.) Questions as to the value of the method when applied to situations where multilinearity does not exist (e.g. direct matching of real frequency characteristics) still have to be answered.

The author believes that the seven programming modifications detailed below would facilitate the evolution of designs using the current techniques or usefully extend the scope of the package. However this poses the question of whether the improvements will be justified by the extra computation and storage required.

(i) The current package can only design RC and RLC networks. An extension to enable the design of RL and LC networks would provide greater scope. (Section 3.3)

(ii) A method involving the evaluation of the gradients could be included to facilitate the removal of elements driven to small values. (Section 3.8) However a method of this kind would not aid a situation where an extreme valued common factor accompanies the element, possibly producing a minimum of the error function. (Section 6.5)

(iii) The removal of nodes should be allowed, with possible safeguards when the structure is already minimal. (Section 6.3)

(iv) The value of any common factor removed should be calculated and the necessary modifications implemented. (Section 3.5)

(v) Wye-delta transformations are advisable when only three elements of one type are connected to a particular node. This will facilitate the subsequent removal, if necessary, of any of these three elements. (Otherwise the elements will tend to large values - section 3.9.)

(vi) If any one node is connected to only two other nodes it is advisable to reverse the connections (as in Figure 5.7), and investigate possible element additions, before introducing a further node.

(vii) It is possible that the creation of a new node will produce a node which is identical in topology to an existing node. (i.e. the new node is at the intersection of the same types of elements as another node which are in turn connected, at the alternative end, to the same nodes - section 6.3.) If such a situation remains after optimization, then the new node addition should be rejected (restarting the evolution with an alternative node addition) unless a true minimum of the overall error function, signified by a low SSQGNC , has been located.

One region into which the program could be extended is to take further account of practical considerations for the existing range of elements, e.g. parasitic elements, constraints on element values and ratio values, sensitivities etc. Constraints on the element values have already been mentioned. It could be argued that effects such as parasitics and sensitivities could be dealt with in a third stage of the design process, say by design centering[29]. The coefficients that are used in the matching process are only approximations to the actual requirements. Consequently, it may prove that a final tuning of the network values to the desired response will always be required. Furthermore, the effect of parasitics is dependent upon the range and the values of the frequencies to which the network is required to respond and, therefore, may not always be a significant factor. Things are further complicated when the extension to active networks is contemplated. Consider, for example, these two contrasting standpoints. The linear approximation of, say, a transistor may be inaccurate to a large degree when compared to that of a capacitor, hence, it would seem to be wasteful to devote undue effort to producing an extremely accurate model of a capacitor. However, unseen parasitics can lead to problems with active devices (e.g. oscillation) and are therefore significant.

Di Mambro[13] described how the present method of analysis could be simply extended to include active devices which can be represented by voltage dependent current sources. One such device model is the hybrid-Π model of a bipolar transistor (Figures 7.1 and 7.2) which can be used in all three configurations.

Some of the effects of including active devices will be more obvious than others. An active network will only be fully defined by all four short-circuit admittance functions and hence five cofactors of the nodal admittance determinant. Furthermore, it will be possible to design without inductors with their associated practical difficulties. A further possible effect could be the cheaper production of a satisfactory network. A mass-produced thick or thin film printed circuit device (capable of representing transistors, resistors and capacitors) may prove cheaper than a passive RLC alternative.

The hybrid-Π model would appear to be an ideal candidate for a first investigation into the design of active networks. An initial investigation could attempt to solve simple problems of, say, neutralisation - i.e. adding external feedback components to overcome the effects of the internal feedback of the transistor. However, there are other factors to consider. The model is only valid at a certain operating point. To retain this level of biasing any d.c. input has to be removed from the device by a coupling reactance (capacitor). Hence, only capacitive connections should be made to a transistor terminal. Furthermore, it may be necessary to include these capacitors in the device model, or at least constrain them within certain values. Coupling capacitors affect the lower cut-off frequency and so should be large (limited by size and cost). In contrast, the impedance (1/pC) needs to be sufficiently large so that the a.c. signal does not affect the biasing resistor, which has thus to be known. Similarly, the characteristics of a single transistor can vary considerably with temperature and so extra external components may be required to nullify these effects. These factors indicate the requirement to design with a specified transistor rather than vary the component values (within certain practical limits) and either locate or manufacture components with appropriate characteristics. Furthermore, for many purposes suitable transistors will be known. For example, Ryder[59] states that many amplifier designs involve predicting and designing around the frequency and delay distortion which results from the coupling capacitors.

A simple active device can be modelled effectively by an equivalent circuit, as the hybrid-Π. However, it would seem to be preferable to model larger devices, such as complete integrated circuits, by their external performance characteristics. This would require minor modification to the formulation of the analysis routine rather than an alternative method.

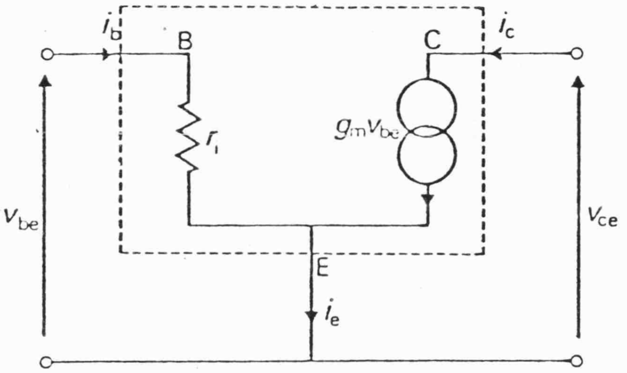


FIGURE 7.1: FIRST ORDER TRANSISTOR SMALL SIGNAL EQUIVALENT CIRCUIT

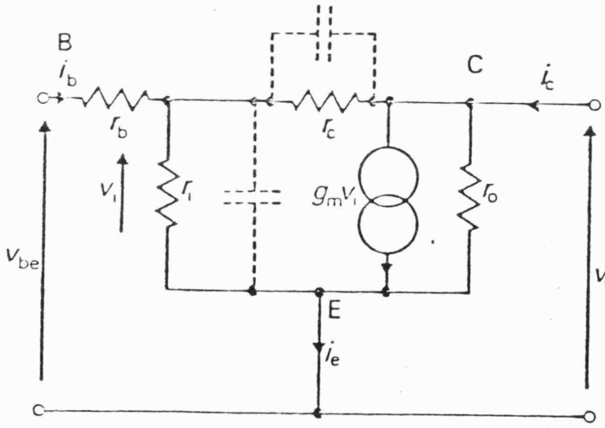


FIGURE 7.2: THE COMPLETE HYBRID - ΠEQUIVALENT OF A TRANSISTOR

APPENDICES

1. Optimum Values for k , the Normalizing Variable

The expressions derived relate to an overall error value F , comprising of the summation of the individual error functions f_i , such that

$$F = \sum_{i=1}^{m} f_i^2 \qquad (1)$$

where, m is the total number of coefficients equated.

Thus,

$$\frac{\partial f}{\partial k} = 2 \sum_{i=1}^{m} f_i \cdot \frac{\partial f_i}{\partial k} \qquad (2)$$

Formulation 1

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

$$\frac{\partial f_i}{\partial k} = - \frac{c_i}{k^2 a_i} - \frac{a_i}{c_i} \qquad (3)$$

$$\frac{\partial F}{\partial k} = 2 \sum_{i=1}^{m} \left(\frac{c_i}{ka_i} - \frac{ka_i}{c_i}\right) \cdot \left(- \frac{c_i}{k^2 a_i} - \frac{a_i}{c_i}\right) \qquad (4)$$

To minimize F with respect to k , $\frac{\partial F}{\partial k} = 0$ (5)

hence,

$$k = \left[\frac{\sum \left(\frac{c_i}{a_i}\right)^2}{\sum \left(\frac{a_i}{c_i}\right)^2} \right]^{\frac{1}{4}} \qquad (6)$$

This corresponds to an error value of

$$F = 2\sqrt{\sum_{i=1}^{m} \left(\frac{c_i}{a_i}\right)^2 \sum_{i=1}^{m} \left(\frac{a_i}{c_i}\right)^2 - 2m} \tag{7}$$

Similarly,

Formulation 2

$$f_i = \frac{c_i}{ka_i} - 1$$

gives

$$k = \frac{\sum \left(\frac{c_i}{a_i}\right)^2}{\sum \left(\frac{c_i}{a_i}\right)} \tag{8}$$

corresponding to an error of

$$F = m - \frac{\left[\sum_{i=1}^{m} \left(\frac{c_i}{a_i}\right)\right]}{\sum_{i=1}^{m} \left(\frac{c_i}{a_i}\right)^2} \tag{9}$$

Formulation 3

$$f_i = \frac{ka_i}{c_i} - 1$$

gives

$$k = \frac{\sum \left(\frac{a_i}{c_i}\right)}{\sum \left(\frac{a_i}{c_i}\right)^2} \tag{10}$$

Formulation 1a

$$f_i = \left(\frac{c_i}{ka_i}\right)^n - \left(\frac{ka_i}{c_i}\right)^n$$

gives

$$k = \left[\frac{\sum \left(\frac{c_i}{a_i}\right)^{2n}}{\sum \left(\frac{a_i}{c_i}\right)^{2n}}\right]^{1/4n} \qquad (11)$$

Formulation 2a

$$f_i = \left(\frac{c_i}{ka_i}\right)^n - 1$$

gives

$$k = \left[\frac{\sum \left(\frac{c_i}{a_i}\right)^{2n}}{\sum \left(\frac{c_i}{a_i}\right)^{n}}\right]^{1/n} \qquad (12)$$

2. An Expression for the Optimum Value of One Common Factor

If the original desired coefficients are of value α_i, then the desired coefficients with a common factor of the form $(p + A)$, will be

$$a_i = A\alpha_i + \alpha_{i-1} \qquad (13)$$

Thus,

$$\frac{\partial a_i}{\partial A} = \alpha_i \qquad (14)$$

If

$$f_i = \frac{ka_i}{c_i} - 1 \qquad (15)$$

and $$F = \sum f_i^2 \qquad (16)$$

then

$$\frac{\partial F}{\partial A} = 2\sum\left(\frac{ka_i}{c_i} - 1\right) . \left(\frac{\partial k}{\partial A} . \frac{a_i}{c_i} + \frac{k}{c_i}\frac{\partial c_i}{\partial A}\right) \qquad (17)$$

but from (10)

$$k\,\frac{\partial k}{\partial x}\sum\left(\frac{a_i}{c_i}\right)^2 - \frac{\partial k}{\partial x}\sum\frac{a_i}{c_i} = 0$$

hence, substituting for k ,

$$\frac{\partial F}{\partial A} = 0$$

when

$$\sum\left(\frac{a_i}{c_i}\right)\sum\frac{a_i}{c_i^2}\,\frac{\partial a_i}{\partial x} - \sum\left(\frac{a_i}{c_i}\right)^2\sum\frac{\partial a_i}{\partial x}\cdot\frac{1}{c_i} = 0 \qquad (18)$$

substituting for (11) and (12), and rearranging

$$A = \frac{\left[\sum\left(\frac{\alpha_i}{c_i}\right)\sum\left(\frac{\alpha_{i-1}}{c_i}\right)^2 - \sum\left(\frac{\alpha_{i-1}}{c_i}\right)\sum\left(\frac{\alpha_i\alpha_{i-1}}{c_i}\right)\right]}{\left[\sum\frac{\alpha_{i-1}}{c_i} . \sum\left(\frac{\alpha_i}{c_i}\right)^2 - \sum\left(\frac{\alpha_i}{c_i}\right)\sum\left(\frac{\alpha_i\alpha_{i-1}}{c_i^2}\right)\right]} \qquad (19)$$

(the terms in A^2 cancel.)

3. The Optimum Values of Virtual Elements

Suppose that an additional variable, μ , is introduced corresponding to the growth of a virtual element. The polynomial coefficients c_i are multilinear functions of the network elements, hence, provided the existing elements are fixed in value,

$$c_i = c_{i0} + \mu c'_{i0} \tag{20}$$

where

c_{i0} is the value of c_i when $\mu = 0$

and c'_{i0} is the value of $\frac{\partial c_i}{\partial \mu}$ when $\mu = 0$.

Substituting the value c_i into equation (9), and equating $(\frac{\partial F}{\partial \mu})$ to zero, Cutteridge[33] obtained the expression

$$\mu = \frac{\sum_{i=1}^{m} (\frac{c_{i0}}{a_i}) \sum_{i=1}^{m} (\frac{c_{i0}}{a_i}) (\frac{c'_{i0}}{a_i}) - \sum_{i=1}^{m} (\frac{c'_{i0}}{a_i}) \sum_{i=1}^{m} (\frac{c_{i0}}{a_i})}{\sum_{i=1}^{m} (\frac{c'_{i0}}{a_i}) \sum_{i=1}^{m} (\frac{c_{i0}}{a_i}) (\frac{c'_{i0}}{a_i}) - \sum_{i=1}^{m} (\frac{c_{i0}}{a_i}) \sum_{i=1}^{m} (\frac{c'_{i0}}{a_i})^2} \tag{21}$$

This can be substituted back into equation 9 to obtain the corresponding error value.

4. Alternative Methods of Representing the Common Factors

Consider a topology which produces three common factors (p + A), (p + B) and (p + C). Let the original, required coefficients be α_i and the revised coefficients be a_i .

Hence,

$$a_i = ABC.\alpha_i + (AB + BC + AB).\alpha_{i-1} + (A + B + C).\alpha_{i-2} + \alpha_{i-3} \; . \qquad (22)$$

When the values of the individual common factors (A,B and C) are varied, higher order derivatives will exist, e.g.

$$\frac{\partial a_i}{\partial A} = BC\alpha_i + (B + C)\alpha_{i-1} + \alpha_{i-2}$$
$$\frac{\partial^2 a_i}{\partial A \partial B} = C\,\alpha_i + \alpha_{i-1}$$
$$\frac{\partial^3 a_i}{\partial A \partial B \partial C} = \alpha_i \qquad (23)$$

It is possible to construct an alternative set of variables x_i such that, with this example,

$$x_1 = ABC$$

$$x_2 = AB + AC + BC$$

$$x_3 = A + B + C$$

i.e. $a_i = x_1\alpha_i + x_2\alpha_{i-1} + x_3\alpha_{i-2}$ (24)

In this case only first order derivatives will exist, e.g.

$$\frac{\partial a_i}{\partial x_i} = \alpha_i$$
$$\frac{\partial^2 a_i}{\partial x_i \partial x_2} = 0 \; . \qquad (25)$$

and the first order derivatives are easily calculated, whatever the number of common factors.

To calculate the values of the new variables y_i (i = 1 to r) when a new common factor, value X, is introduced

$$y_1 = x_1 . X$$

$$y_n = x_{n-1} + X$$

otherwise

$$y_i = X.x_i + x_{i-1} . \qquad (26)$$

The removal of a common factor is similarly straightforward.

5. An RLC-Series - Parallel Network Having no RC - Series - Parallel Equivalent

Consider the network of Figure 6.33 as a solution of functions 6.17. The arguments expressed in section 6.8 concerning Fialkow's functions still hold. Namely, the network (if it is a realization) is realized by introducing an extra pole into the two subnetworks with residues of identical magnitude but opposite sign effectively cancelling the pole.

Functions 6.17 can be expressed in terms of their poles as:

$$y_{11} = y_{22} = \frac{1}{800000}\left[r_1 . \; p + r_o + \frac{p . r_\alpha}{p + \alpha} + \frac{p . r_\beta}{p + \beta}\right]$$

$$-y_{12} = \frac{1}{8000000}\left[r_1 . p + r_o - \frac{p . r_\alpha}{p + \alpha} - \frac{p . r_\beta}{p + \beta}\right]$$

where $r_1 = 3$, $r_o = 16170$

$$r_\alpha, r_\beta = \frac{8086.335\sqrt{0.060225} \pm 1962.753775}{\sqrt{0.060225}}$$

and $\alpha, \beta = 0.255 \pm \sqrt{0.060225}$

Subnetwork 1 satisfies the functions

$$y_{11} = y_{22} = \frac{1}{800000}\left[r_1 . p + \frac{p . r_\alpha}{p + \alpha} - \frac{p . r_A}{p + A}\right]$$

$$-y_{12} = \frac{1}{8000000}\left[r_1 . p - \frac{p . r_\alpha}{p + \alpha} + \frac{p . r_A}{p + A}\right]$$

and subnetwork 2 satisfies the functions

$$y_{11} = y_{22} = \frac{1}{800000}\left[r_o + \frac{p . r_\beta}{p + \beta} + \frac{p . r_A}{p + A}\right]$$

$$-y_{12} = \frac{1}{8000000}\left[r_o - \frac{p . r_\beta}{p + \beta} - \frac{p \ r_A}{p + A}\right]$$

where $A = \alpha - \dfrac{3\alpha^2}{r_\alpha}$

and

$$r_A = \frac{(3\alpha - r_\alpha)^2}{r_\alpha} \quad .$$

The element values can be determined to be

$$C_{13} = 7.5 \times 10^{-6}$$

$$G_{03} = 2 . C_{13} . (A + \alpha)$$

$$L_{03} = 2 . C_{13} . A . \alpha$$

$$G_{12} = (r_o - r_\beta - r\ ) \div 8000000$$

$$G_{13} = 2 . (r_A + r_\beta) \div 8000000$$

$$C_{45} = \frac{G_{13} . 2 . (\beta . r_A + r_\beta . A)}{A\beta(r_A + r_\beta)}$$

$$C_{05} = \frac{G_{13} \div 2 . (r_A + r_\beta) . C_{45}}{\{C_{45} . (Ar_A + r_\beta) - 2 . G_{13} . (r_A + r_\beta)\}}$$

$$G_{05} = \frac{(r_A . \beta + A . r_\beta)}{(r_A + r_\beta)} . C_2$$

REFERENCES

1. M.E. VAN VALKENBURG, "Introduction to Modern Network Synthesis", New York, Wiley, 1960.

2. L. WEINBERG, "Network Analysis and Synthesis", New York, McGraw-Hill, 1962.

3. A. FAILKOW, "A Limitation of the Series-Parallel Structure", IEEE Trans. Circuit Theory, Vol. CT-15, pp.124-132, 1968.

4. Computing Europe, Vol.6, No.41, 19th October, 1978.

5. D.A. CALAHAN, "Computer-Aided Network Design", New York, McGraw-Hill, 1972 (1968).

6. S.W. DIRECTOR, edt., "Computer-Aided Circuit Design: Simulation and Optimization", Benchmark Papers in Electrical Engineering and Computer Science, 1973.

7. M.R. AARON, "The Use of Least Squares in System Design", IRE Trans. Circuit Theory, CT-3, 224-231, (1956).

8. C.A. DESOER and S.K. MITRA, "Design of Lossy Ladder Filters by Digital Computer", IRE Trans. Circuit Theory, CT-8, 192-201 (1961).

9. D.A. CALAHAN, "Computer Design of Linear Frequency Selective Networks", Proc. IEEE, 53, No.11, 1701-1706, (1965).

10. S.W. DIRECTOR and R.A. ROHRER, "Automated Network Design - The Frequency Domain Case", IEEE Trans. Circuit Theory, CT-16, No.3, 330-337 (1969).

11. G. KAPLAN, "Computer-Aided Design", IEEE Spectrum, Vol.12, No.10, October 1975, pp.40-47.

12. D.J. WRIGHT, "Studies in Optimisation as an Aid to Circuit Synthesis and Design", Ph.D. Thesis, University of Leicester, 1974.

13. P.H. DI MAMBRO, "A Study of the Evolutionary Approach to Network Synthesis Using Coefficient Matching", Ph.D. Thesis, University of Leicester, 1974.

14. O.M.H.O. HEGAZI, "An Investigation into Current Possibilities in Automated Network Design", Ph.D. Thesis, University of Leicester, 1977.

15. O.P.D. CUTTERIDGE, "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,19

16. A.J. KRZECZKOWSKI, "Theory and Design of Lumped Linear Three-Terminal RC Networks", Ph.D. Thesis, University of Leicester, 1976.

17. O.P.D. CUTTERIDGE, "Automated Synthesis of Non-Series-Parallel 3-Terminal RC Networks", Proc.Fifth Colloquium on Microwave Communication (Budapest, Hungary), June 1974.

18. O.P.D. CUTTERIDGE and P.H. DI MAMBRO, "Some Examples Demonstrating Feasibility of Evolutionary Approach to Linear-Network Synthesis", Electron.Lett., Vol.10, pp.30-31, 1974.

19. H.M. LUCAL, "Synthesis of Three-Terminal RC Networks", IRE Trans. Circuit Theory, Vol.CT-2, pp.308-316, 1955.

20. B.R. SMITH and G.C. TEMES, "An Iterative Approximation Procedure for Automatic Filter Synthesis", IEEE Trans. Circuit Theory, CT-12, 107-112 (1965).

21. P.H. DI MAMBRO, "An Investigation into the Uses of Coefficient Matching in Network Synthesis", M.Sc. Thesis, University of Leicester, 1970.

22. O.P.D. CUTTERIDGE and P.H. DI MAMBRO, "Simultaneous Generation of the Coefficients of Network Polynomials and their Partial Derivatives from the Nodal Admittance Matrix", Electron. Lett. 1970, Vol.6, pp.308-310.

23. O.P.D. CUTTERIDGE and P.H. DI MAMBRO, "Simultaneous Generation of the Partial Derivatives from the Nodal Admittance Matrix: Further Details and Results", Electron. Lett., 1971, Vol.7, pp.3-4.

24. O.P.D. CUTTERIDGE, P.H. DI MAMBRO and D.J. WRIGHT," "Accuracy Problems Arising in the Simultaneous Generation of Network Polynomial Coefficients and their Partial Derivatives", Unpublished report, 1973.

25. K. SINGHAL and J. VLACH, "Interpolation Using the Fast Fourier Transform", Proc. IEEE 60, 1558 (1972).

26. K. SINGHAL and J. VLACH, "Accuracy and Speed of Real and Complex Interpolation", Computing 11, 147-158 (1973).

27. Dr. A.R. BANJERJEE, Private Communication, 1975.

28. S.K. MITRA, "Analysis and Synthesis of Linear Active Networks", New York, Wiley, 1969.

29. K.S. TAHIM, "Statistical Analysis and Design of Electrical Networks", Ph.D. Thesis, University of London, 1978.

30. O.P.D. CUTTERIDGE,"Parallel Decomposition of 3-terminal RC Networks", Proc. IEE, Vol.112, pp.2220-2226, 1965.

31. P. HANSEN and G. WANET, "Synthesis of LC 3-Terminal Networks by Mixed Integer Programming", IEEE Trans. Circuit Systems, Vol.CAS-21, pp.199-203, 1974.

32. R. FLETCHER and C.M. REEVES. "Function Minimisation by Conjugate Gradients" Computer Journal, Vol.7, pp.149-154, 1964.

33. O.P.D. CUTTERIDGE, "Expressions for Optimum Values to Grow Virtual Elements", IEE Journal on ELEC. Circuits and Systems, Vol.2, No.1, Jan.1978, pp.29-30.

34. KENDALL, L. SU, "Time-Domain Synthesis of Linear Networks", New Jersey, Prentice Hall, 1971.

35. W. CAUER, "Synthesis of Linear Communication Networks", New York, McGraw-Hill, 1958.

36. O.P.D. CUTTERIDGE, "Numerical Experience with Some Two-Part Programs for the Solution of Non-Linear Simultaneous Equations", Conf. on Numerical Methods for Non-Linear Optimization, University of Dundee, 1971.

37. O.P.D. CUTTERIDGE, "Electrical Network and General System Synthesis Using Optimization Techniques", (in Optimization and Design, eds. M. Avriel, M.J. Rijkaert and D.J. Wilde) pp.76-80, New Jersey, Prentice Hall, 1973.

38. C. POTTLE, "Comprehensive Active Network Analysis by Digital Computer - A State-Space Approach", Proc. Third Allerton Conference, University of Illinois, pp.659-668, 1965.

39. C. POTTLE, "A 'Textbook' Computerized State-Space Network Analysis Algorithm", IEEE Trans. Circuit Theory, CT-16, No.4, 566-568 (1969).

40. O.P.D. CUTTERIDGE, "Powerful 2-Part Program for Solution of Non-Linear Simultaneous Equations", Electron. Lett., 1974, Vol.10, No.10

41. M. DOWSON, "Multimodal Univariable Search Techniques and their Application to Optimization Problems", M.Phil. Thesis, University of Leicester, 1978.

42. J.T. HENDERSON, "Optimization Techniques and their Application", Ph.D. Thesis, University of Leicester, 1979.

43. H.M. MARKOWITZ, "The Elimination Form of the Inverse and its Applications to Linear Programming", Management Sci., 3, 255-269 (1957).

44. W.F. TINNEY and J.W. WALKER, "Direct Solutions of Sparse Network Equations by Optimally Ordered Triangular Factorization", Proc.IEEE, 55, No.11, 1801-1809 (1967).

45. H.B. LEE, "An Implementation of Gaussian Elimination for Sparse Systems of Linear Equations", Proc.Symp. Sparce Matrices, 75-83 (1968).

46. F.G. GUSTAVSON, W.M. LINIGER and R.A. WILLOUGHBY, "Symbolic Generation of an Optimal Crout Algorithm for Sparse Systems of Linear Equations", J.Assn. Computing Machinery, 17, No.1, 87-109 (1970).

47. SHU-PARK CHAN, "Introducing Topological Analysis of Electrical Networks" New York, Holt, Rinehart and Winston Inc., 1969.

48. R.A. ROHRER, "Fully Automated Network Design by Digital Computer: Preliminary Considerations", Proc. IEEE, 55, No.11, 1929-1939 (1967).

49. G.D. HATCHTEL and R.A. ROHRER, "Techniques for the Optimal Design and Synthesis of Switching Circuits", Proc. IEEE, 55, No.11, 1864-1877 (1967).

50. S.W. DIRECTOR and R.A. ROHRER, "The Generalized Adjoint Network and Network Sensitivities", IEEE Trans. Circuit Theory, CT-16, No.3, 318-323, (1969).

51. S.W. DIRECTOR, "LU Factorization in Network Sensitivity Computations", IEEE, Trans. Circuit Theory, CT-18, No.1, 184-185 (1971).

52. O.P.D. CUTTERIDGE and M. DOWSON, "Methods for the Solution of Nonlinear Simultaneous Equations Incorporating Some Variations of Levenberg's Technique", University of Leicester internal report, 73-13, July 1973.

53. R. FLETCHER and M.J.D. POWELL, "A Rapidly Convergent Descent Method for Minimization", Computer J., 163-168 (1963).

54. R.E. MASSARA and J.K. FIDLER, "Computer Optimization of Frequency Selective Networks", Proc. 1974, European Conf. on Circuit Theory and Design, IEE Conf. publication No.116, pp.133-140.

55. O.P.D. CUTTERIDGE and A.J. KRZECZKOWSKI, "Improved Methods of Synthesizing Linear Networks by Coefficient Matching", IEEE Trans. Circuit Systems, Vol.CAS-22, pp.486-489, 1975.

56. A. FIALKOW, "The Matrix of a Transformerless Network", Quart.Appl. Math. Vol.22, pp.57-70, 1964.

57. A. FIALKOW, "Series-Parallel Grounded Two-Ports", Quart.Appl.Math., Vol.24, pp.195-213, 1966.

58. O.P.D. CUTTERIDGE and A.J. KRZECZKOWSKI, "Further Non-Series-Parallel 3-Terminal RC Networks Having No Series-Parallel Equivalent", IEEE Trans.Circuit Systems, Vol.CAS-22, pp.381-385, 1975.

59. J.D RYDER, "Electronic Fundamentals and Applications", 4th Ed., Prentice Hall.