COMPUTER ORIENTED ALGORITHMS FOR ANALYSIS OF ELECTRICAL NETWORKS AND ITS IMPLEMENTATION WITH PROGRAMMING

Kankana Bhattacharyya*
Department of Electrical Engineering, Assam Engineering College, Guwahati-781013
*For correspondence. (kankanabhattacharyya94@gmail.com)

Abstract: The paper is concerned with computational techniques for solving complex electrical networks mainly with the help of the C programming language. All electrical networks can be represented with the help of matrices. Solution of the matrices gives us the solution for the electrical circuits. The solution of the matrices can be found analytically but it involves a very tedious process. Therefore, matrix computation has been done with the help of the C programming language. Hence, complex electrical circuits can be solved by computational techniques easily and efficiently. The paper studies the analytical ways for the solution of linear network systems and then it focuses on its implementation through C algorithms. The prime objective is to present a systematic evaluation of realistic network circuits with a programming perspective.

Keywords: electrical network; matrix computation; C algorithms

1. Introduction:

Circuit analysis enables the numerical solving of various electrical networks. By solution of electrical networks, we mean determination of the voltages and currents of all the nodes and the branches. Computer aided circuit analysis [1] employs methods to solve various electrical circuits via algorithms [2] implemented through programming. As we continue to analyse even more complex circuits, it will become obvious rather quickly that it is easy to make errors during the analysis, and verifying solutions by hand can be time-consuming. The use of computer-aided analysis programmes [3] for the study of electric circuits has become increasingly popular. There are many reasons for using computer-aided analysis programmes for analysing circuits. Remarkably the high computational speed of digital computers offers great saving of time when large circuits are to be analysed. Besides, while hand calculation can easily commit errors, the computer always produces accurate answers provided accurate and viable information about a circuit is specified.

However, computers do not ‘think’ although they are powerful and efficient workhorses for performing tedious and time-consuming calculations. Without understanding the circuit to be analysed, we will never use any computer-aided analysis programme effectively and correctly. Although computer aided circuit analysis is a relatively quick means of determining voltages and currents in a circuit, we should be careful not to allow simulation packages to completely replace traditional ‘paper and pencil’ analysis. There are several reasons for these. First, in order to design we must be able to analyse. Overreliance on software tools can inhibit the development of necessary analytical skills. Secondly, it is virtually impossible to use a complicated software package over a long period of time without making some kind of data entry error.

In further chapters, we discuss the solution of electrical networks via programming methods mainly constructed in C.

2. Methodology:

2.1. Circuit Analysis:

The *Mesh Current Method*, also known as the *Loop Current Method*, uses simultaneous equations, Kirchhoff's Voltage Law, and Ohm's Law to determine unknown currents in a network [4]. It differs from the Branch Current method in that it does *not* use Kirchhoff's Current Law, and it is usually able to solve a circuit with less unknown variables and less simultaneous equations.

Mesh analysis is carried out in the circuits given below:

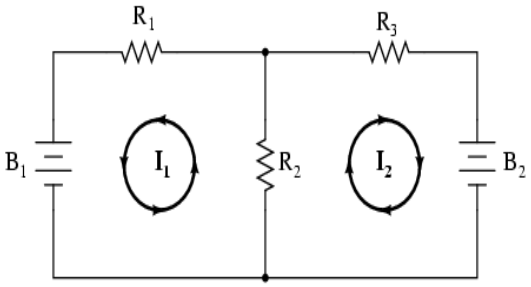

Figure 1: Circuit with 2 loops.

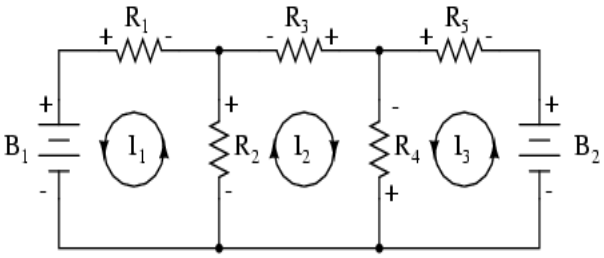

Figure 2. Circuit consisting of 3 loops

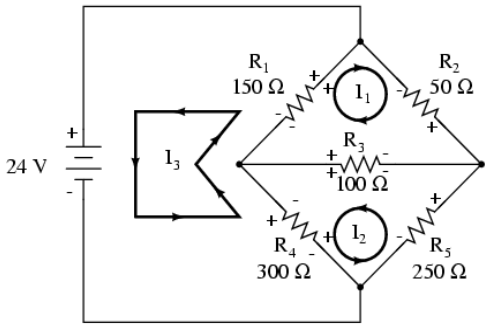

Figure 3: Circuit representing Wheatstone bridge.

By applying mesh analysis, the various voltages and currents in any branch in the above circuits can be found out.

3. Procedure:

3.1. Matrix Computation:

A matrix is a rectangular array of elements, where each element can be either a numeric value or an algebraic expression [5]. Thus, in matrix there could be real or complex numbers, or algebraic expressions, or with matrices themselves. However, matrix computation will involve matrices which have elements in numerical form. To specify any element of a matrix, we replace whatever labels the rows and the columns have by numbers, say 1 to m for rows and 1 to n for columns. The element on row i and column j of matrix A is defined as a_{ij}. Thus element in row 2 and column 3 of matrix A is denoted as a_{23} or $a_{2,3.}$

3.2. Solution of simultaneous equations:

A set of n linear simultaneous equations is given as follows:

$$a_{11}x_1 + a_{12}x_2 + \ldots\ldots\ldots\ldots a_{1n}x_n = b_1$$
$$a_{21}x_1 + a_{22}x_2 + \ldots\ldots\ldots\ldots a_{2n}x_n = b_2$$
$$\ldots\ldots\ldots\ldots$$

$$a_{n1}x_1 + a_{n2}x_2 + \ldots\ldots\ldots\ldots a_{nn}x_n = b_n$$

may be written in matrix form as Ax=b. Where A is an n*n coefficient matrix having typical element a_{ij} and x and b are column vectors of the variables and right-hand sides respectively. The solution of such a set of equations for x is a key operation in the solution of a vast number of problems.

Cramer's rule gives the solution in determinant form. Whereas a solution of the equation by Cramer's rule is numerically uneconomical as compared with elimination methods, and also difficult to automate, it does not establish that there is a unique solution for all equations provided $|A| \neq 0$.

A more appropriate method of solution of linear simultaneous equations is by finding the inverse using Gaussian Elimination which has been discussed below in the following chapter.

3.3. Method of Gaussian elimination:

Elimination methods are the most important methods of solving simultaneous equations either by hand or computing [6], essentially when the number of equations is not very large.
To perform Gaussian elimination starting with the system of equations

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1\,k} \\ a_{21} & a_{22} & \cdots & a_{2\,k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k\,1} & a_{k\,2} & \cdots & a_{k\,k} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \end{bmatrix}, \tag{2}$$

We compose the "augmented matrix equation"

$$\left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1\,k} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2\,k} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{k\,1} & a_{k\,2} & \cdots & a_{k\,k} & b_k \end{array}\right] \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix}. \tag{3}$$

Here, the column vector in the variables $\mathbf{x}$ is carried along for labelling the matrix rows. Next, we perform elementary row operations to put the augmented matrix into the upper triangular form

$$\left[\begin{array}{cccc|c} a'_{11} & a'_{12} & \cdots & a'_{1\,k} & b'_1 \\ 0 & a'_{22} & \cdots & a'_{2\,k} & b'_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & a'_{k\,k} & b'_k \end{array}\right]. \tag{4}$$

We then solve the equation of the kth row for x_k, then substitute back into the equation of the $(k-1)$th row to obtain a solution for x_{k-1}, etc., according to the formula

$$x_i = \frac{1}{a'_{i\,i}} \left(b'_i - \sum_{j=i+1}^{k} a'_{i\,j}\, x_j \right). \tag{5}$$

LU decomposition of a matrix is frequently used as part of a Gaussian elimination process for solving a matrix equation.

3.4. Algorithm (Naïve Gaussian elimination):

This algorithm finds the n-element solution vector x for a matrix equation Ax=b, where A is an n*n matrix with known elements and b is an n-element column vector with known elements.

Step 1: The **A** matrix is augmented by appending the **b** vector on the right. The augmented matrix is denoted as **C**.

Step 2: Row 1 is designated as the working row and column 1 as the working column. The intersection of the working row and working column is called working position. Let i denote the index of the working position:i←1.(The matrix element c_{ii} located in the working position is called the pivot, or pivot element).

Step 3:From each row k that lies below the working row(i.e., for k=i+1,i+2,…..n), subtract s times row i where $s=c_{kj}/c_{ij}$.(This operation will place zeroes in column j of rows i+1 through n).

Step 4: If i>n-1, go to step 5;otherwise designate a new working position:i←i+1 and return to step 3.

Step 5: At this point, the modified sub matrix A will be in upper triangular form. Set $x_n \leftarrow b_n/a_{nn}$, where $a_{nn}=c_{nn}$ and $b_{nn}=c_{nn+1}$

Step 6: For j=n-1,n-2,…….,1,set $x_i = \frac{1}{a'_{i\,i}}\left(b'_i - \sum_{j=i+1}^{k} a'_{i\,j}\, x_j\right)$, where $a_{jk}=c_{jk}$ and $b_j=c_{jn+1}$

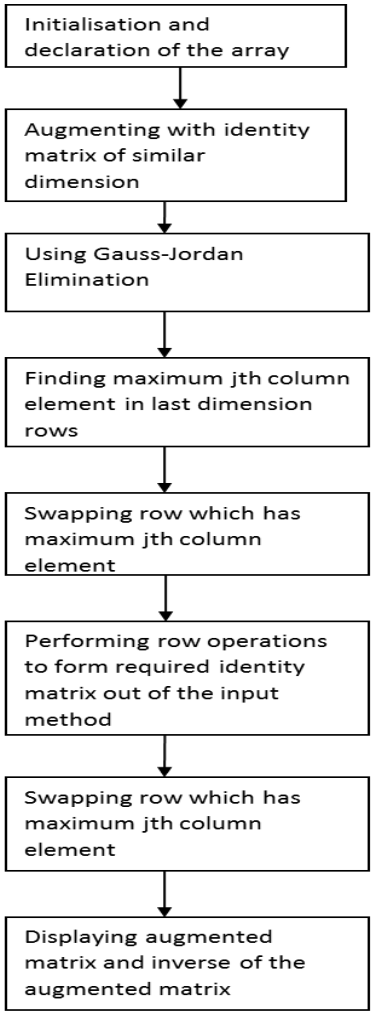

Figure 4: Flow Diagram for Programming Perspective.

3.5. Matrix representation of complex numbers in programming:

Generally, representation of a.c. circuits consisting of elements like inductance and capacitance is done with the use of the operator j. In matrix representation, these generally denote of a complex matrix.

In the programming perspective, complex matrices can be implemented in the following two ways:

1. Storing real and imaginary parts interleaved, usually done by the use of structures in C as shown below-

```
struct
    {
double real;
double imaginary;
    } complex com[m][n];
```

This method is generally applied when we need to work on the matrix element-wise.

2. Storing real and imaginary parts of the matrix separately as shown below-

```
Struct
    {
double rearray[m][n];
double iarray[m][n];
    } Carray ;
```

This method is more efficient than the above mentioned method and enables us to deal with the entire matrix on the whole.

4. Conclusion:

The paper covered the various approaches of solution of electrical networks. Matrix representation of electrical circuits (both dc and ac) and the solution of electrical circuits (voltage, currents associated with the branch and nodes) have been discussed analytically and further their implementation through programming has been elaborated.

Initially, the mesh analysis of circuits and their representation in matrix form was accounted for. The solutions of linear equation were then described. Under that, the Cramer's rule was elaborated. Another efficient way of solution of linear equations was determining the inverse of the matrix computed from the linear network given. The Gaussian Elimination helps in the determination of the matrix inverse. The algorithm associated with Gaussian Elimination and the counter flow diagram for programming purpose has been mentioned. The paper ends with the representation of complex matrices with their programming perspective. In this way complex electrical networks can be simplified by computational techniques.

Acknowledgement:

It is with immense gratitude that I acknowledge the support and help of my Guide Dr PRABIN KUMAR BORA, Professor in the Department of Electrical and Electronics Engineering, Indian Institute of Technology, Guwahati under whose guidance the project was carried out.

I thank the Library of the institution for providing me the necessary books for the required amount of time and share with it the credit of my work as I found most of the material of my paper there.

Last but not the least; I am indebted to my parents and my family for their keen interest and constant support. Without their support this paper would not have been possible.

References:

[1] William H. Hayt and Jack E Kemmerly and Steve M. Durbin, Engineering Circuit Analysis,2013, McGraw Hill Education (India) Private Limited, New Delhi.
[2] Applied Circuit Analysis, Matthew N.O Sadiku
[3] Jennings Alan, Matrix computation, 1992.
[4] Computer Aided Circuit Analysis, Pearson pdf.
[5] Stewart G.U, Introduction to matrix computation, 1973.
[6] ieeexplore.ieee.org