# SYMBOLIC ANALYSIS OF ELECTRONIC CIRCUITS USING WAVELET TRANSFORM

*A. A. Al-Itaby and Prof. F. M. Al-Naima*
*Department of Computer Engineering,*
*College of Engineering,*
*Al-Nahrain University,*
*BAGHDAD, IRAQ*

## ABSTRACT

In recent years, symbolic analysis has become a well-established technique in circuit analysis and design. The symbolic expression of network characteristics offers convenience for frequency response analysis, sensitivity computation, and fault diagnosis. The aim of the paper is to present a method for symbolic analysis that depends on the use of the wavelet transform (WT) as a tool to accelerate the solution of the problem as compared with the numerical interpolation method that is based on the use of the fast Fourier transform (FFT).

**التحليل الرمزي للدوائر الالكترونية بأستخدام تحويل الموجة**

الخلاصة

اصبح التحليل الرمزي للدوائر في السنوات الأخيرة تقنية موثوق بها في تحليل وتصميم الدوائر الإلكترونية. وتوفر هذه التعابير الرمزية لمواصفات الشبكات وسائل ملائمة جداً في تحليل الاستجابة الترددية وحسابات الحساسية وتشخيص الأعطال. إن الهدف من هذا البحث هو تقديم طريقة للتحليل الرمزي للدوائر والذي يعتمد على تحويل الموجة كوسيلة لتسريع حل المسائل بالمقارنة مع الطريقة العددية والتي أساسها تحويل فوريير السريع.

المهندس علي عبد الإله العتابي والأستاذ الدكتور فوزي محمد منير النعمة
قسم هندسة الحاسوب- جامعة النهرين- بغداد-العراق

## 1.INTRODUCTION

It is obvious that the methods of symbolic analysis can be divided mainly into two categories. These are the topological and numerical methods [1]. Each one of these methods has its own advantages and disadvantages. For instance, in topological methods the number of elements represented as symbols is large but the circuits that can be handled is small, while in numerical methods, fairly large networks can be handled but the number of symbolic variables should not exceed 10. The direct application of numerical interpolation method can be used to solve problems of system matrix size of 30 and about 10 elements only represented as variables beside the complex frequency "s" [2,3].

Many algorithms have been developed and from these determinant and flow graph methods appear to be favoured in terms of flexibility and efficiency [3]. All approaches suffer from restrictions inherent to the problem, the escalation of computer time and memory requirements with increase in circuit size. One serious limitation of such methods, in practice, is the rapidly increasing amount of computations required as the number of symbols to be handled increases [3]. This will, in fact, increase the time required to solve the linear system equation of the circuit.

The numerical interpolation method for obtaining the symbolic analysis suffers from serious limitation in practice, which is the rapidly increasing amount of computations required as the number of symbols to be handled increases. This, in fact, reflects the amount of time required to perform the analysis. For this reason, it is useful to find an approach to minimize the computations required by the numerical interpolation as minimum as possible. The usual numerical interpolation method is based on the use of the FFT. One way to reduce the computations required by the numerical interpolation is to search for a transform that will perform the required task, besides minimizing the computations, and hence, reduces the required time to perform the analysis as compared to the FFT. As an example, the ***Hartly Transform (HT)*** could be used to replace the FFT for the symbolic analysis and a comparison could be made between the HT and the FFT to see which is better from the point of view of reducing the required computation, and hence the time of doing the analysis. One other promising transform that may replace the FFT is the ***Wavelet Transform (WT)*** [4,5]. A new approach to minimize the computations and the time required is the ***neural network*** approach to the interpolation problem that allows to get the solution in a real time [6].

The method proposed in this paper tries to reduce the time required by the numerical interpolation method to solve the system equation by using the wavelet transform.

## 2.NUMERICAL INTERPOLATION METHOD FOR SYMBOLIC ANALYSIS

Numerical interpolation methods are based on the theory and implementation of numerical methods for generating symbolic functions of networks. They seem to have a lower computational cost than other well-known symbolic analysis algorithms such as a parameter extraction method.
The following discussion will introduce the idea of using interpolation in finding network transfer functions using the Discrete Fourier Transform (DFT) [3,7,8,9].

## 2.1 POLYNOMIAL INTERPOLATION

First, We find N+1 points by evaluating the function:

$$P_N(x) = det[A(x)] \quad (1)$$

at $x_0, x_1, \ldots, x_N$ where N is the maximum power of x. Now, there are N+1 distinct points $(x_i, y_i=P_n(x_i))$, i=0, 1, …,N. Both $x_i$ and $y_i$ may be real or complex numbers. We wish to find the coefficients of the polynomial:

$$P_N(x) = \sum_{n=0}^{N} a_n x^n \quad (2)$$

such that the polynomial passes through the given points.

Inserting $x_i$ into the polynomial (2), We obtain the set of equations:

$$a_0 + a_1 x_i + a_2 x_i^{\,2} + \cdots + a_N x_i^{\,N} = y_i \qquad i = 0, 1, \cdots, N \quad (3)$$

with unknowns $a_0, a_1, a_2, \ldots, a_N$. Since there are N+1 unknown coefficients and the same number of equations, We can write the matrix equation:

$$\begin{bmatrix} 1 & x_0 & x_0^{\,2} & \cdots & x_0^{\,N} \\ 1 & x_1 & x_1^{\,2} & \cdots & x_1^{\,N} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_N & x_N^{\,2} & \cdots & x_N^{\,N} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_N \end{bmatrix} \quad (4)$$

Or:

$$[X][A] = [Y] \quad (5)$$

The solution of (5) provides the unknown coefficients.

As we have the choice of selecting the points $x_i$, the question arises as to what the choice should be in order to obtain the best possible result. It can be shown that the interpolation with real $x_i$ is, in general, numerically unstable [7].

## 2.2 THE USE OF THE DISCRETE FOURIER TRANSFORM IN INTERPOLATION

We will derive this interpolation by introducing first a special symbol for the matrix X in (5):

$$X = [x_i^{\,n}] \quad (6)$$

where the index i and the exponent n run from 0 to N. If We choose the set of points $x_i$ to be uniformly spaced on the unit circle in the complex plane, then these points are :

$$x_0 = 1, \quad x_k = \exp[\frac{j2k\pi}{N+1}], \; k = 1, 2, \ldots, N. \quad (7)$$

Introduce the substitution:

$$w = exp[\frac{j2\pi}{N+1}] \quad (8)$$

Then:

$$x_k = w^k \quad (9)$$

And:

$$X = [w^{in}] \quad (10)$$

It can be shown that [3]:

$$X^{-1} = \frac{1}{N+1} I w^{-in} I = \frac{1}{N+1} X^{*} \qquad (11)$$

Where $X^{*}$ denotes the transpose conjugate matrix and I runs from 0 to N.

The solution of (5) with the points defined by (7) is:

$$A = X^{-1} Y = \frac{1}{N+1} I w^{-in} I Y \qquad (12)$$

or:

$$a_n = \frac{1}{N+1} \sum_{k=0}^{N} y_k \; w^{-nk} \qquad (13)$$
$$n = 0, 1, 2, \ldots, N.$$

The original polynomial in (2), evaluated at $x_k$ , can be written as:

$$y_k = \sum_{n=0}^{N} a_n \; w^{nk} \qquad (14)$$

Equation (13) and (14) represent the solution of one another. They are called the Discrete Fourier Transform (DFT) pair.

To improve the speed of the method, one can use a fast algorithm in interpolation. Algorithms that reduce the computational cost of DFT are, in general called the Fast Fourier Transform (FFT). The DFT has been studied extensively. It can be programmed in a very efficient way, particularly when $N+1=2^m$, m being a positive integer. The number of operations required in this case is m (N+1) [3,5].

## 3. THE USE OF THE WAVELET TRANSFORM (WT)

In this section, the use of the Discrete Wavelet Transform (DWT) will be illustrated. Before this, the DWT must be briefly explained.

### 3.1 THE WAVELET TRANSFORM

Like the FFT, the Discrete Wavelet Transform (DWT) is a fast linear operation that operates on a data vector whose length is an integer power of two, transforming it into a numerically different vector of the same length. Also, like the FFT, the WT is invertible and in fact orthogonal, that is, the inverse transform when viewed as a big matrix, is simply the transpose of the transform. Both FFT and DWT, therefore, can be viewed as a rotation in space, from the input space (or time) domain, where the basis functions are the unit vectors $e_i$, or Dirac delta functions in the continuum limit, to a different domain. For the FFT, this new domain has basis functions that are the familiar sines and cosines. In the wavelet domain, the basis functions are somewhat more complicated and have the fanciful names ***"mother functions"*** and ***"wavelets"*** [10].

Of course, there are an infinitely of possible bases for function space, almost all of them uninteresting. What makes the wavelet basis interesting is that, unlike sines and cosines, individual wavelet functions are quite localized in space; simultaneously, like sines and cosines, individual wavelet functions are quite localized in frequency or (more precisely) characteristic scale. The particular kind of dual localization achieved by wavelets renders large classes of functions and operators sparse, or sparse to some high accuracy, when transformed into the wavelet domain. Analogously with the Fourier domain, where a class of computations, like convolutions, become computationally fast, there is a large class of computations (those that can take the advantage of sparsity) that become computationally fast in the wavelet domain [4,7,10].

Unlike sines and cosines, which define a unique Fourier transform, there is no one single unique set of wavelets; in fact, there are infinitely many possible sets. Roughly, the different sets of wavelets make different trade-offs between how compactly they are localized in space and how smooth they are.

### *3.2 DAUBECHIES WAVELET FILTER COEFFICIENTS*

A particular set of wavelets is specified by a particular set of numbers called wavelet filter coefficients. Here, we will largely restrict ourselves to wavelet filters in a class discovered by Daubechies. This class includes members ranging from highly localized to highly smooth. The simplest (and most localized) member, often called ***DAUB4***, has only four coefficients, $c_0$, $c_1$, $c_2$, and $c_3$ [4,7,10].

Consider the following transformation matrix acting on a column vector of data to its right:

$$W = \begin{bmatrix} c_0 & c_1 & c_2 & c_3 & & & & & & & & \\ c_3 & -c_2 & c_1 & -c_0 & & & & & & & & \\ & & c_0 & c_1 & c_2 & c_3 & & & & & & \\ & & c_3 & -c_2 & c_1 & -c_0 & & & & & & \\ \vdots & \vdots & & & & & & & & & & \\ \vdots & \vdots & & & & & & & & & & \\ \vdots & \vdots & & & & & & & & & & \\ & & & & & & c_0 & c_1 & c_2 & c_3 & & \\ & & & & & & c_3 & -c_2 & c_1 & -c_0 & & \\ c_2 & c_3 & & & & & & & c_0 & c_1 & & \\ c_1 & -c_0 & & & & & & & c_3 & -c_2 & & \end{bmatrix} \qquad (15)$$

Here, blank entries signify zeroes. Note the structure of this matrix. The first row generates one component of the data convoluted with the filter coefficients $c_0$, $c_1$, $c_2$, and $c_3$. Likewise, the third, fifth, and other odd rows. If the even rows followed this pattern offset by one, then matrix would be a circulant, that is, an ordinary convolution that could be done by FFT methods. (Note how the last two rows wrap around like convolutions with periodic boundary conditions.) Instead of convolution with $c_0$, $c_1$, $c_2$, and $c_3$, however, the even rows perform a different convolution, with coefficients $c_3$,$-c_2$, $c_1$, and $-c_0$. The action of the matrix, overall, is thus to perform two related convolutions, then to decimate each of them by half (throw away half the values), and interleave the remaining halves.

It is useful to think of the filter $c_0$, $c_1$, $c_2$, and $c_3$ as being a smoothing filter called H, something like a moving average of four points. Then, because of the minus sign, the filter $c_3$, $-c_2$, $c_1$, and $-c_0$, call it G, is not a smoothing filter. In fact, the c's are chosen so as to make G yield, insofar as possible, a zero response to a sufficiently smooth data vector. This is done by requiring the sequence $c_3$, $-c_2$, $c_1$, and $-c_0$ to have a certain number of vanishing moments. When this is the case for p moments (starting with the zeroth), a set of wavelets is said to satisfy an "approximation condition of order p". This result in the output of H, decimated by half, accurately representing the data's "smooth" information. The output of G, also decimated is referred to as the data's "detail" information [10].

For such a characterization to be useful, it must be possible to reconstruct the original data vector of length N from its N/2 smooth or s-components and its N/2 detail or d-components. That is effected by requiring the matrix (15) to be orthogonal, so that its inverse is just the transposed matrix:

$$W^T = W^{-1} = \begin{bmatrix} c_0 & c_3 & & & \cdots & \cdots & & & c_2 & c_1 \\ c_1 & -c_2 & & & & & & & & \\ c_2 & c_1 & c_0 & c_3 & & & & & & \\ c_3 & -c_0 & c_1 & -c_2 & & & & & & \\ & & & & \ddots & & & & & \\ & & & & & c_2 & c_1 & c_0 & c_3 & \\ & & & & & c_3 & -c_0 & c_1 & -c_2 & \\ & & & & & & c_2 & c_1 & c_0 & c_3 \\ & & & & & & c_3 & -c_0 & c_1 & -c_2 \end{bmatrix} \qquad (16)$$

Now, since

$$W\,W^{-1} = W\,W^{T} = I \qquad (17)$$

where I is the identity matrix, one sees immediately that matrix (16) is the inverse of matrix (15) if and only if the following two equations hold:

$$\begin{aligned} c_0^{\,2} + c_1^{\,2} + c_2^{\,2} + c_3^{\,2} &= 1 \\ c_2 c_0 + c_3 c_1 &= 0 \end{aligned} \qquad (18)$$

If additionally, we require the approximation condition of order p=2, then two additional relations are required:

$$\begin{aligned} c_0^{\,2} + c_1^{\,2} + c_2^{\,2} + c_3^{\,2} &= 1 \\ c_2 c_0 + c_3 c_1 &= 0 \end{aligned} \qquad (19)$$

Equations (18) and (19) are 4 equations for the 4 unknowns $c_0$, $c_1$, $c_2$, and $c_3$, first recognized and solved by Daubecheies. The unique solution (up to a left-right reversal) is:

$$c_0 = \frac{(1+\sqrt{3})}{4\sqrt{2}} \qquad c_1 = \frac{(3+\sqrt{3})}{4\sqrt{2}}$$

$$c_2 = \frac{(3-\sqrt{3})}{4\sqrt{2}} \qquad c_3 = \frac{(1-\sqrt{3})}{4\sqrt{2}} \qquad (20)$$

In fact, DAUB4 is only the most compact of a sequence of wavelet sets: If we have six coefficients instead of four, there would be three orthogonality requirements in equation (18) (with offsets of zero, two and four), and we could require the vanishing of p=3 moments in equation (19). In this case, ***DAUB6***, the solution coefficients can also expressed in closed form:

$$c_0 = \frac{(1+\sqrt{10}+\sqrt{5+2\sqrt{10}})}{16\sqrt{2}}$$

$$c_1 = \frac{(5+\sqrt{10}+3\sqrt{5+2\sqrt{10}})}{16\sqrt{2}}$$

$$c_2 = \frac{(10-2\sqrt{10}+2\sqrt{5+2\sqrt{10}})}{16\sqrt{2}}$$

$$c_3 = \frac{(10-2\sqrt{10}-2\sqrt{5+2\sqrt{10}})}{16\sqrt{2}}$$

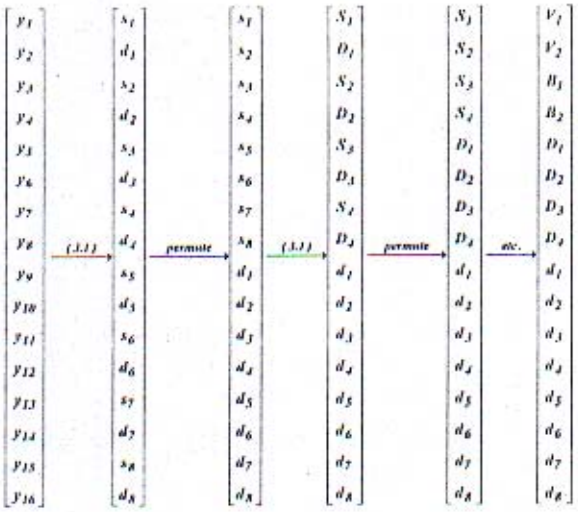
$$c_4 = \frac{(5+\sqrt{10}-3\sqrt{5+2\sqrt{10}})}{16\sqrt{2}}$$

$$c_5 = \frac{(1+\sqrt{10}-\sqrt{5+2\sqrt{10}})}{16\sqrt{2}} \qquad (21)$$

For higher p, up to 10, Daubechies has tabulated the coefficients numerically. The number of coefficients increases by two each time p is increased by one.

### *3.3 THE DISCERTE WAVELET TRANSFORM (DWT)*

The DWT consists of applying a Wavelet coefficient matrix like (15) hierarchically, first to the full data vector of length N, then to the "smooth" vector of length N/2, then to the **"smooth?smooth"** vector of length N/4, and so on until only a trivial number of **"smooth?...?smooth"** components (usually 2) remain. The procedure is sometimes called a ***pyramidal algorithm*** (or ***Mallat's*** pyramid algorithm), for obvious reasons. The output of the DWT consists of these remaining components and all the "detail" components that were accumulated along the way. A diagram should make the procedure clear :




(22)

If the length of the data vector was a higher power of two, there would be more stages of applying (15) (or any other Wavelet coefficients) and permuting. The end point will always be a vector with two **V's** and a hierarchy of **B's, D's, d's**, etc. Notice that once d's are generated, they simply propagate through to all subsequent stages.

A value $d_i$ of any level is termed a ***"Wavelet coefficient"*** of the original data vector; the final values $V_1$, $V_2$ should strictly be called ***"mother-function coefficients"***, although the term "Wavelet coefficients" is often used loosely for both d's and final V's. Since the full procedure is
a composition of orthogonal linear operations, the whole DWT is itself an orthogonal linear operator.

To invert the DWT, one simply reverses the procedure, starting with the smallest level of the hierarchy and working (in eq. (22)) from right to left. The inverse matrix (16) is of course used instead of matrix (15).

Procedures that embody the DWT and IDWT (Inverse Discrete Wavelet Transform) are available to be used later for obtaining the symbolic analysis using the Wavelet transform.

### *3.4 THE USE OF DWT FOR FAST SOLUTION OF LINEAR SYSTEMS*

One of the most interesting, and promising, wavelet applications is linear algebra [10]. The basic idea is to think of integral operator (that is, a large matrix) as a digital image. Suppose that the operator compresses well under a two-dimensional wavelet transform, i.e., that a large function of its wavelet coefficients are so small as to be negligible. Then any system involving the operator becomes a ***sparse system*** in the wavelet basis. In other words, to solve:

$$A \cdot x = b \qquad (23)$$

we first wavelet-transform the operator A and the right-hand side b by:

$$\tilde{A} \equiv W \cdot A \cdot W^T \quad , \quad \tilde{b} \equiv W \cdot b \qquad (24)$$

where W represents the one-dimensional wavelet transform, then solve:

$$\tilde{A} \cdot \tilde{x} = \tilde{b} \qquad (25)$$

which is a ***sparse system*** in the wavelet basis, and hence, this property can be used to solve this system in a faster way than usual, by using methods for solving the sparse systems, so that we can obtain the results almost in a real-time manner.

Finally, transform to the answer by the inverse wavelet transform:

$$x = W^T \cdot \tilde{x} \qquad (26)$$

The results will appear with a high accuracy as compared with the use of other transforms to perform the same task.

The method discussed above was implemented and verified for solving numerical linear systems in a fast way. It is

also adopted to solve the linear system that will be obtained when performing the symbolic analysis. The problem that will arise is to do the above operations in a ***symbolic way***, and hence solving the linear system symbolically as fast as possible. This problem is overcome and applied to solve the symbolic linear system to convert it to a sparse symbolic system in the wavelet basis. This system is then solved using a method for solving the sparse system symbolically also. This, in fact, reduces the time required to obtain the symbolic analysis as will be shown later.

### *3.5 THE WAVELET MATRICIES*

As one can see from eq. (15), the W matrix of dimension $4 \times 4$ is as shown below:

$$W = \begin{bmatrix} c_0 & c_1 & c_2 & c_3 \\ c_3 & -c_2 & c_1 & -c_0 \\ c_2 & c_3 & c_0 & c_1 \\ c_1 & -c_0 & c_3 & -c_2 \end{bmatrix} \quad (27)$$

Where $c_0$, $c_1$, $c_2$, and $c_3$ are the DAUB4 filter coefficients as explained previously.

Now, the W matrix of dimension $8 \times 8$ is as shown below:

$$W = \begin{bmatrix} c_0 & c_1 & c_2 & c_3 & 0 & 0 & 0 & 0 \\ c_3 & -c_2 & c_1 & -c_0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c_0 & c_1 & c_2 & c_3 & 0 & 0 \\ 0 & 0 & c_3 & -c_2 & c_1 & -c_0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_0 & c_1 & c_2 & c_3 \\ 0 & 0 & 0 & 0 & c_3 & -c_2 & c_1 & -c_0 \\ c_2 & c_3 & 0 & 0 & 0 & 0 & c_0 & c_1 \\ c_1 & -c_0 & 0 & 0 & 0 & 0 & c_3 & -c_2 \end{bmatrix} \quad (28)$$

Note the sparsity as the dimension of the matrix increases. The above matrix contains 64 elements, 32 of them are zeros. The W matrix of dimension $16 \times 16$ contains more zero entries in it and so on for higher order of W matrices. This will lead, when we use it to transform a linear system, to obtain a sparse system that makes its solution easier and faster. Table 1 shows a comparison between the size of the W matrices and the number of zeros included in them.

**Table 1 Comparison between the size of the W matrices and the number of their zeros**

| *SIZE OF W MATRIX* | *TOTAL NUMBER OF ELEMENTS* | *TOTAL NUMBER OF ZEROS* | *SPARSITY RATIO* %* |
|---|---|---|---|
| *4X4* | *16* | *NONE* | *0* |
| *8X8* | *64* | *32* | *50* |
| *16X16* | *256* | *192* | *75* |
| *32X32* | *1024* | *896* | *87.5* |
| *64X64* | *4096* | *3840* | *93.75* |
| *128X128* | *16384* | *15872* | *96.875* |

** Sparsity Ratio= (Total Number of Zeros) / (Total Number of Elements).*

It was found that the number of zeros in the W matrix for DAUB4 filter can be found by the formula:

$$Z = D^2 - 4D \quad (29)$$

where Z is the number of zeros and D is the dimension of the W matrix (D = 4, 8, 16,...).

### *3.6 HARMONIC WAVELET TRANSFORM*

The wavelets studied so far have all been derived with real coefficients. For example, N wavelet coefficients can be computed by solving the N nonlinear algebraic equations that define them. When this is done, it turns out that the underlying spectrum of a wavelet with N coefficients becomes more box-like as N increases. This fact led to seek a wavelet w(x) whose spectrum is exactly like a box so that the magnitude of its Fourier transform $W(\omega)$ is zero except for an octave band of frequencies. The corresponding complex wavelet is:

$$w(x) = \left(e^{j4\pi x} - e^{j2\pi x}\right) / j2\pi x \qquad (30)$$

## 4. APPLICATION EXAMPLES

This section presents some examples of using the previously mentioned algorithm that depends on the use of the DWT as compared with the use of the FFT from the point of view of reducing the amount of calculations and hence the execution time. The software required to perform this task is written using the language of the MATLAB package.

For the purpose of fair comparison, two versions of the symbolic analysis programs were written, one uses the FFT (called SAUFFT: Symbolic Analyzer Using Fast Fourier Transform) and the other uses the DWT (called SAUDWT: Symbolic Analyzer Using Discrete Wavelet Transform) in numerical interpolation. Also, the circuits were used in both programs to perform the symbolic analysis. The results are obtained using a Pentium II microprocessor that operates on 233 MHz frequency and with 16 MB RAM memory.

**EXAMPLE 1**: Consider the RC ladder circuit shown in Fig.1. It is desired to find the voltage transfer function $V_o/V_i$. This circuit contains passive elements only with 10 symbolic variables. The description of the circuit was input to the program in a SPICE-like format.

The analysis of this circuit using program SAUDWT and SAUFFT yields the same transfer function but with different times of execution. The result is as shown below:

**THE NUMERATOR IS:**

1

**THE DENOMINATOR IS:**

$C_1\ C_2\ C_3\ C_4\ C_5\ R_1\ R_2\ R_3\ R_4\ R_5\ s^5 +$
$(R_4\ C_1\ R_1\ R_2\ C_3\ C_4\ R_5\ C_5 +$
$R_4\ R_3\ R_2\ C_1\ R_1\ C_2\ C_3\ C_4 + R_4\ R_2\ C_1\ R_1\ C_2\ C_4\ R_5\ C_5 + R_2\ C_1\ R_1\ C_2\ R_3\ C_4\ R_5\ C_5 + R_4\ R_3\ R_2\ C_1\ R_1\ C_2\ C_3\ C_5 + R_4\ R_3\ C_1\ R_1\ C_3\ C_4\ R_5\ C_5 + R_4\ R_3\ R_1\ C_2\ C_3\ C_4\ R_5\ C_5 + R_3\ R_2\ C_1\ R_1\ C_2\ C_3\ R_5\ C_5 + R_4\ R_3\ C_2\ R_2\ C_3\ C_4\ R_5\ C_5)\ s^4 +$

$(R_4\ C_1\ R_1\ C_4\ R_5\ C_5 + R_4\ C_2\ R_2\ C_4\ R_5\ C_5 + R_4\ R_3\ R_1\ C_2\ C_3\ C_4 + R_3\ R_2\ C_1\ R_1\ C_2\ C_3 + R_4\ C1\ R_1\ R_2\ C_3\ C_5 + R_4\ R_1\ C_2\ C_4\ R_5\ C_5 + R_4\ R_2\ C_3\ C_4\ R_5\ C_5 + R_4\ R_3\ C_2\ R_2\ C_3\ C_4 + R_4\ R_1\ C_3\ C_4\ R_5\ C_5 + R_4\ R_3\ C_2\ R_2\ C_3\ C_5 + R_3\ R_1\ C_2\ C_3\ R_5\ C_5 + R_1\ C_2\ R_3\ C_4\ R_5\ C_5 + R_4\ C_3\ R_3\ C_4\ R_5\ C_5 + R_4\ R_3\ R_1\ C_2\ C_3\ C_5 + R_4\ R_3\ C_1\ R_1\ C_3\ C_5 + C_1\ R_1\ R_3\ C_4\ R_5\ C_5 + R_3\ C_1\ R_1\ C_3\ R_5\ C_5 + C_1\ R_1\ R_2\ C_3\ R_5\ C_5 + C_2\ R_2\ R_3\ C_4\ R_5\ C_5 + R_4\ C_1\ R_1\ R_2\ C_3\ C_4 + R_4\ R_3\ C_1\ R_1\ C_3\ C_4 + R_3\ C_2\ R_2\ C_3\ R_5\ C_5 + R_3\ R_2\ C_1\ R_1\ C_2\ C_5 + C_1\ R_1\ R_2\ C_4\ R_5\ C_5 + R_2\ C_2\ R_1\ C_2\ R_3\ C_4 + R_4\ R_2\ C_1\ R_1\ C_2\ C_5 + R_2\ C_1\ R_1\ C_2\ R_5\ C_5 + R_4\ R_2\ C_1\ R_1\ C_2\ C_4)\ s^3 +$

$(R_4\ R_2\ C_3\ C_5 + R_4\ C_3\ R_3\ C_5 + R_4\ C_2\ R_2\ C_4 + C_1\ R_1\ R_5\ C_5 + R_3\ C_2\ R_2\ C_3 + C_2\ R_2\ R_5\ C_5 + R_4\ R_1\ C_3\ C_5 + R_4\ R_1\ C_3\ C_4 + R_1\ C_2\ R_5\ C_5 + R_2\ C_3\ R_5\ C_5 + R_1\ C_4\ R_5\ C_5 + R_4\ C_2\ R_2\ C_5 + R_1\ C_2\ R_3\ C_4 + C_4\ R_4\ R_5\ C_5 + R_4\ R_1\ C_2\ C_5 + R_4\ R_2\ C_3\ C_4 + R_2\ C_4\ R_5\ C_5 + C_1\ R_1\ R_2\ C_4 + C_1\ R_1\ R_3\ C_4 + R_1\ C_3\ R_5\ C_5 + C_2\ R_2\ R_3\ C_4 + R_4\ C_1\ R_1\ C_5 + C_1\ R_1\ R_2\ C_3 + R_2\ C_1\ R_1\ C_2 + R_4\ C_3\ R_3\ C_4 + R_3\ C_2\ R_2\ C_5 + C_3\ R_3\ R_5\ C_5 + R_3\ R_1\ C_2\ C_5 + C_1\ R_1\ R_2\ C_5 + R_3\ C_1\ R_1\ C_3 + R_3\ C_1\ R_1\ C_5 + R_4\ R_1\ C_2\ C_4 + R_4\ C_1\ R_1\ C_4 + R_3\ R_1\ C_2\ C_3 + R_3\ C_4\ R_5\ C_5)\ s^2 +$

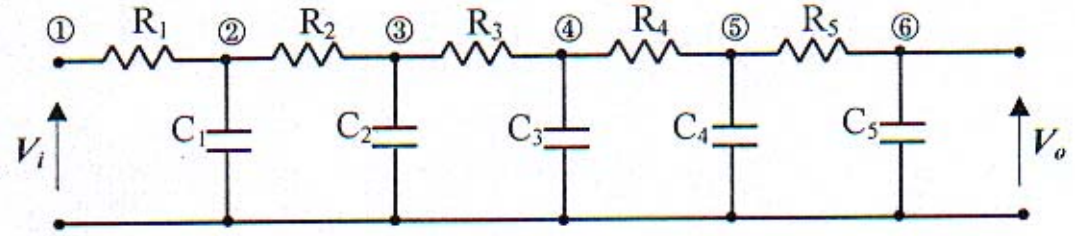



*Fig.1 Circuit of example 1*

The analysis of this circuit using program SAUDWT and SAUFFT yields the same transfer function but with different times of execution. The result is as shown below:

$$(C_4\ R_4 + C_5\ R_5 + R_2\ C_2 + C_1\ R_1 + R_3\ C_4 + R_1\ C_3 + R_1\ C_5 + R_3\ C_5 + R_2\ C_3 + R_1\ C_4 + R_2\ C_5 + R_1\ C_2 + R_4\ C_5 + C_3\ R_3 + R_2\ C_4)\ s + 1$$

```
TIME  OF EXECUTION OF PROGRAM
SAUDWT:
TIME= 7 SECONDS.
TIME  OF EXECUTION OF PROGRAM
SAUFFT:
TIME=19 SECONDS.
```

**EXAMPLE 2:** Consider the circuit shown in Fig.2. The circuit contains eight symbolic variables, which are $C_1$, $C_2$, $C_3$ ,$C_4$ , $g_{m1}$, $g_{m2}$ , $g_{m3}$, and $g_{m4}$, where the $g_m$'s are the transconductances of the OTA ( Operational Transconductance Amplifier) devices. The active devices are modeled using the nullator-norator equivalent circuit.

**THE NUMERATOR IS..**

$$g_{m1}\ g_{m2}\ g_{m3}\ g_{m4}$$

**THE DENOMINATOR IS..**

$$C_1\ C_2\ C_3\ C_4\ s^4 + C_1\ C_2\ C_3\ g_{m4}\ s^3 + (g_{m3}\ g_{m2}\ C_1\ C_4 + C_1\ C_2\ g_{m4}\ g_{m3})\ s^2 + (g_{m3}\ g_{m2}\ g_{m1}\ C_4 + C_1\ g_{m4}\ g_{m3}\ g_{m2})\ s + g_{m3}\ g_{m1}\ g_{m2}\ g_{m4}$$

```
TIME  OF EXECUTION OF PROGRAM
SAUDWT:
TIME=4 SECONDS.
TIME  OF EXECUTION OF PROGRAM
SAUFFT:
TIME=10 SECONDS.
```

**EXAMPLE 3:** Consider the circuit shown in Fig. 3. The circuit contains 11 symbolic elements and 4 OPAMP's (Operational Amplifiers). After analyzing this circuit using the two programs, the result was:

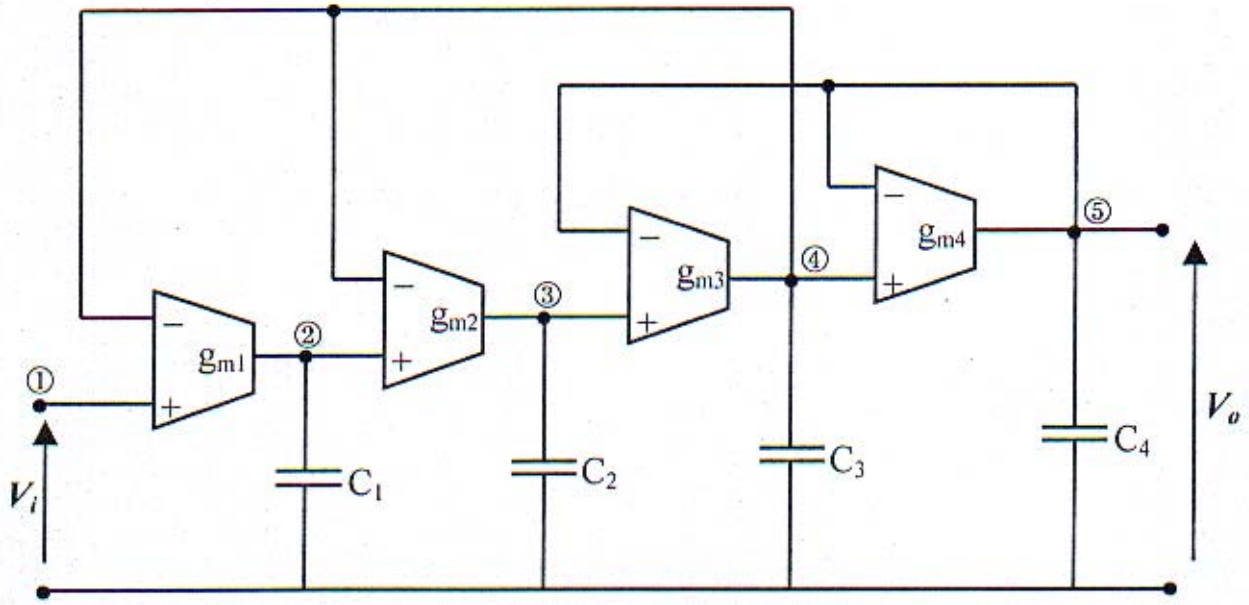



*Fig. 2 Circuit of example 2*

**THE NUMERATOR IS..**

$-R_{10}\ R_7\ R_4\ R_5\ C_2\ R_2\ R_3\ C_1\ R_1\ s^2 - R_{10}(R_7\ R_4\ R_5\ C_2\ R_2\ R_3 - C_2\ R_2\ R_9\ R_1\ R_3\ R_5)\ s - R_{10}\ R_7\ R_4\ R_6\ R_1$

**THE DENOMINATOR IS..**

$R_7\ R_9\ R_4\ R_5\ C_2\ R_2\ R_3\ C_1\ R_1\ s^2 + R_7\ R_9\ R_4\ R_5\ C_2\ R_2\ R_3\ s + R_7\ R_9\ R_4\ R_6\ R_1$

***EXECUTION TIME:***
***PRGOGRAM ONE SAUDWT:***
***TIME=10 SECONDS.***
***PROGRAM TWO SAUFFT:***
***TIME=25 SECONDS.***

## 5. PERFORMANCE COMPARISON BETWEEN THE FFT AND THE DWT

Fig.4 shows a simple comparison between the performance of the FFT and the DWT for their use in the symbolic analysis. From the figure, we can see that for small number of symbolic variables, the performance of the two transforms is almost the same. At large number of symbolic variables, however, the difference becomes very clear between the two transforms. Also, one can see that the DWT continues in providing the analysis for large number of symbolic variables with excellent time, while in FFT, the time increases rapidly with increasing the symbolic variables and it fails at certain number of symbolic variables to provide the required results. It should be mentioned that these results (those shown in Fig.4) are taken for a certain set of circuits

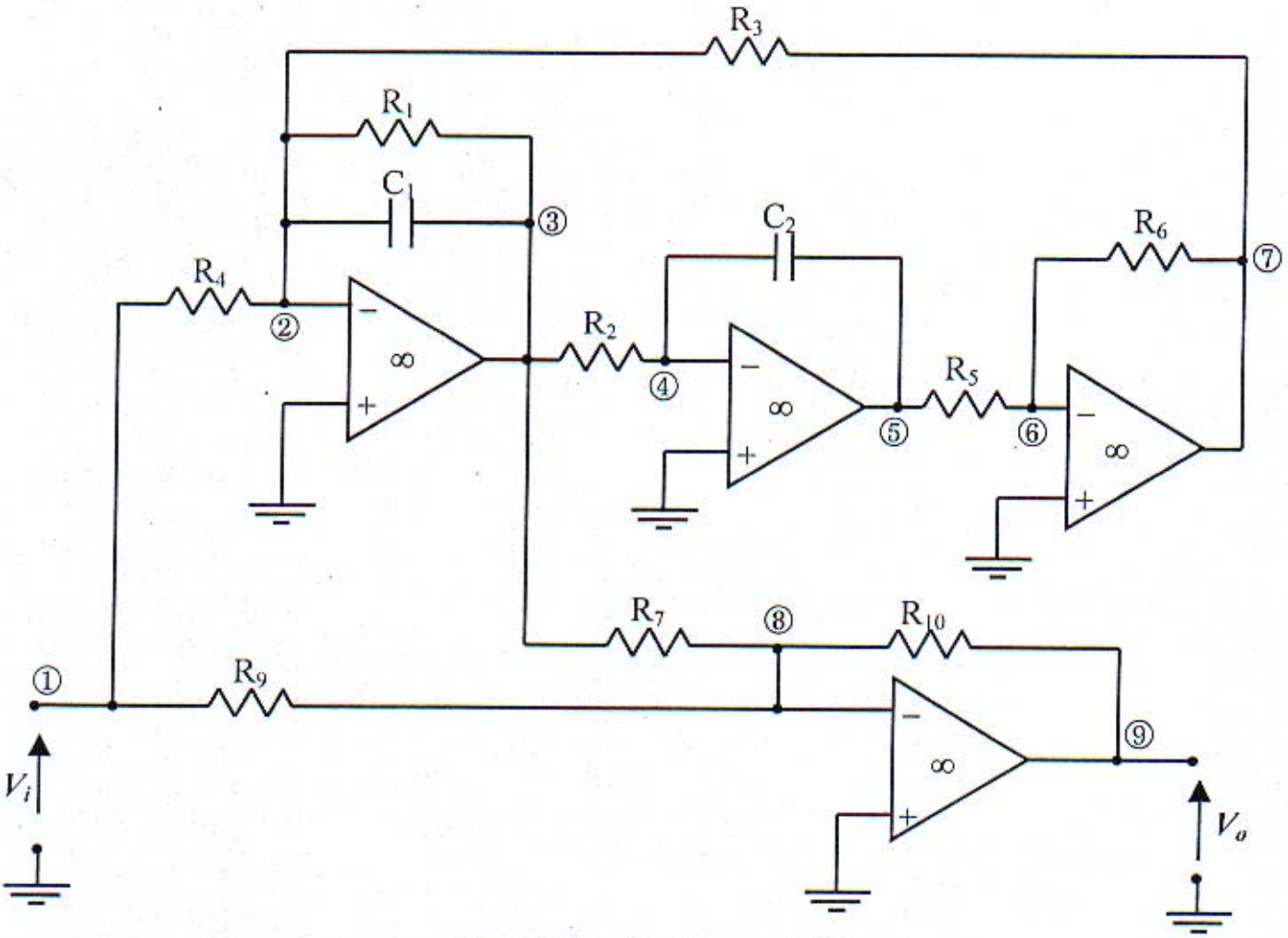



*Fig. 3 Circuit of example 3*

and applied to programs SAUDWT and SAUFFT for the purpose of fair comparison. Of course, not only the number of symbolic elements affects the required time of execution, but also the configuration of the circuit, that is the number of nodes and branches. The figure shows the results up to about 26 symbolic variables and circuits with larger number of variables can also be analyzed with the program SAUDWT only.

## 6. CONCLUDING COMMENTS

The application of wavelets is still new. The subject is developing fast and many questions remain to be answered. For example, What is the best choice of wavelet to use for a particular problem? How far does the harmonic wavelet's computational simplicity compensate for its slow rate of decay in the x-domain (proportional to $x^{-1}$)? . For condition monitoring, the DWT (using families of orthogonal wavelets) will be competing with time-frequency methods using the Short-Time Fourier Transform (STFT) and the Wigner-Ville distribution [4,10]. Orthogonal wavelets give fast algorithms and there is no redundancy: N data points give N wavelet amplitudes. Instead of a signal's mean-square being given by the

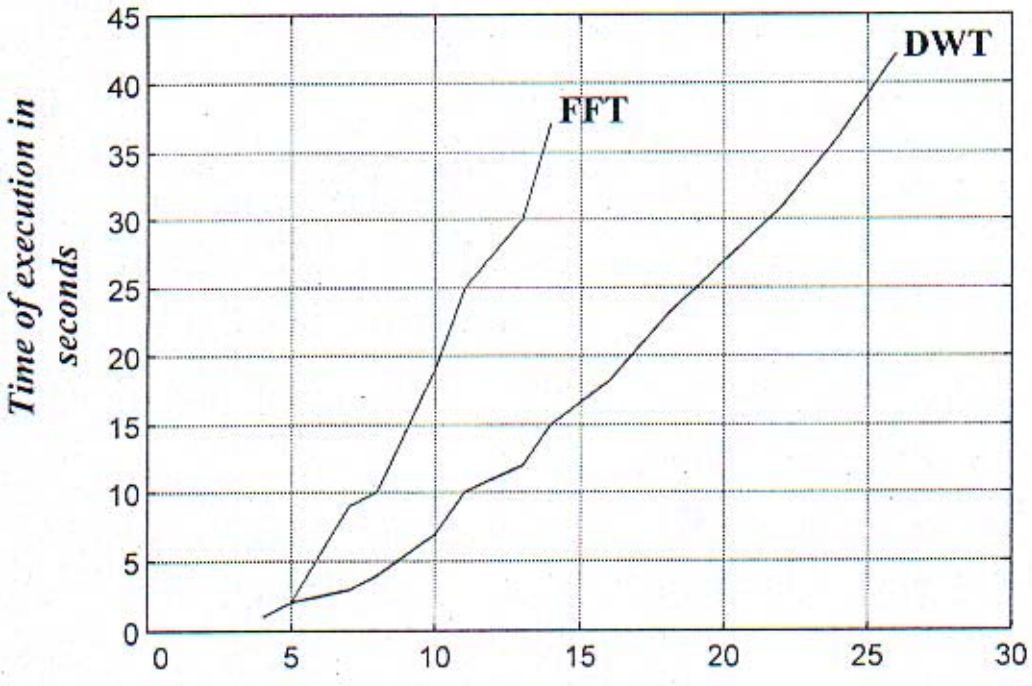



***Fig.4 Comparison in the timing performance between the DWT and the FFT transforms when used in the symbolic analysis***

area under its spectral density curve, mean-square is given by the volume under a two-dimensional wavelets surface with time (or distance) as one axis and wavelet level (a measure of frequency) as the other axis. In contrast, the STFT and Wigner-Ville methods provide redundant information than would be needed to reconstruct the signal being analyzed and the computations take longer to complete [4].

Most of the usefulness of wavelets rests on the fact that wavelet transforms can usefully be severely ***truncated***, that is, turned into sparse expansions. The case of Fourier transforms is different: FFTs are ordinarily used without truncation, to compute fast convolutions, for example. This works because the convolution operator is particularly simple in Fourier basis [4,5,10].

Harmonic wavelets can be described by a simple analytical formula, they are compact in the frequency domain, and are described by a complex function. Dilation wavelets cannot be expressed in functional form, they are compact in the x-domain.

## REFERENCES

[1] **G. Gielen, P. Wambaq,** and **W. Sansen,** " Symbolic Analysis Methods and Applications for Analog Circuits: A tutorial Overview ", Proc. IEEE, Vol. 82, No. 2, February 1994.

[2] **K. Singhal** and **J. Vlach,** " Symbolic Analysis of Analog and Digital Circuits ", IEEE Transactions on Circuits and Systems, Vol. CAS-24, No.11, November 1977.

[3] **K. S. Yeung,** " Symbolic Network Function Generation Via Discrete Fourier Transform ", IEEE Transactions on Circuits and Systems, Vol. CAS-31, No. 2, 1984.

[4] **D. E. Newland,** *An Introduction to Random Vibrations, Spectral and Wavelet Analysis*, Longman Scientific and Technical, 1993.

[5] **C. S. Burrus, R. A. Gopinath, and H. Guo,** *Introduction to Wavelets and Wavelet Transforms*, Prentice-Hall, 1998.

[6] **S. Osowski,** " Interpolation Through Neural Networks ", Institute of Electrical Engineering, Poland, 1996.

[7] **J. Vlach** and **K. Singhal,** *Computer Methods for Circuit Analysis and Design*, Van Nostrand-Reinhold, 1983.

[8] **K. Singhal** and **J. Vlach,** " Generation of Immittance Functions in Symbolic Form for Lumped Distributed Active Networks ", IEEE Transactions on Circuits and Systems, Vol. CAS-21, No.1, January 1974.

[9] **K. Yeung** and **F. Kumbi,** " Symbolic Matrix Inversion With Application to Electronic Circuits ", IEEE Transactions on Circuits and Systems, Vol. CAS-35, No. 2, February 1988.

[10] **H. P. William, T. V. William, A. T. Saul,** and **P. E. Brian,** *Numerical Recipes in C, The Art of Scientific Computing*, Cambridge, 1992.