Wavelets for Efficient Algorithms in Electronic Design Automation

by Nick Soveiko

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Ottawa-Carleton Institute for Electrical and Computer Engineering
Department of Electronics
Carleton University

The undersigned recommend to the Faculty of Graduate Studies and Research acceptance of this thesis

"Wavelets for Efficient Algorithms in Electronic Design Automation"

submitted by Nick Soveiko (M.Sc.) in partial fulfillment of the requirements for the degree of Doctor of Philosophy

---

Chairman. Department of Electronics

Professor Michel Nakhla

Thesis Supervisor

Professor Michel Nakhla

External Examiner

Professor Tapan K. Sarkar

ABSTRACT

With current advances in electronics, numerical methods once applicable only to analog design are becoming essential for digital design as well. The scale of present day analog and, particularly, digital design requires that the traditional numerical techniques for analysis and simulation be much more effective than in the past. Development of not only efficient, but gracefully scalable numerical methods is a top priority in Electronics Design Automation. This work attempts to treat the problem of efficient and scalable numerical methods for EDA in the scope of multiresolution analysis. It shows how analysis and simulation problems can be treated in a systematic way based on the generalized operator equation formulation. Wavelet bases are presented within this framework. The thesis puts particular emphasis on the circuit analysis and simulations applications.

The thesis presents a newly developed Harmonic Balance-like method for steady state analysis of nonlinear circuits under periodic excitations, which is representative of the class of problems described by nonlinear differential equations. The technique features sparse representation of both derivative operator and nonlinear term and shows significant advantage over traditional methods, particularly for analysis of large scale, highly nonlinear, multitone and broadband circuits.

A number of other applications are also considered, namely transient analysis of nonlinear circuits, interconnect macromodelling and capacitance extraction for multiconductor transmission lines. A new approach to capacitance extraction using wavelets is presented featuring extremely aggressive thresholding of the stiffness matrix.

ACKNOWLEDGEMENTS

I would like to express my gratefulness to the following people for their invaluable help and support. First and foremost, to my supervisor Prof. Michel Nakhla (Carleton) for his patience and understanding, among many other things. To Prof. George Costache (University of Ottawa) for his kindness. And in no particular order to: Prof. Peter Galko (University of Ottawa), Prof. Christoph Schwab (ETH Zurich), Prof. Wolfgang Dahmen (IGPM Aachen), Prof. Albert Cohen (Université Pierre et Marie Curie, Paris), Prof. Gilbert Strang (MIT), Dr. Roni Khazaka (McGill), Emad Gad (Carleton) and Anestis Dounavis (Carleton).

Many of the pictures reproduced in this thesis were created with the help of WaveLab, a free wavelet toolbox for Matlab. Many thanks to all the people that are behind this wonderful piece of software.

Special thanks must go to Ritchie, Roger, both Ians, Jon, Steve and Don for the inspiration. To Jim for *one*, to Petey for *that* sustain, Darek and Andrzej for introducing me to the art of combining malt, barley, hops, yeast and water (with wonderful results!), to all my friends on Bob (and some that are not), and last but not least, to my mother, without whom...

TABLE OF CONTENTS

LIST OF FIGURES

3. Steady state analysis of nonlinear circuits

LIST OF TABLES

LIST OF ABBREVIATIONS

AMD	Approximate Minimum Degree (matrix reordering)
BE	Backward Euler (integration formula)
BEM	Boundary Elements Method
BJT	Bipolar Junction Transistor
BVP	Boundary Value Problem
CAD	Computer Aided Design
CPU	Central Processing Unit
EDA	Electronics Design Automation
EM	Electromagnetic(s)
FBS	Forward/Backward substitution
FD	Finite Differences
FEM	Finite Elements Method
FFT	Fast Fourier Transform
FIR	Finite Impulse Response (of a digital filter)
FWT	Fast Wavelet Transform
HB	Harmonic Balance (particularly referring to the Fourier Series expansion of circuit equations)
HPF	High Pass Filter
IF	Intermediate Frequency
IM	Intermodulation
IVP	Initial Value Problem

L2	Level 2 cache memory
LNA	Low Noise Amplifier
LO	Local Oscillator
LPF	Low Pass Filter
LU	LU decomposition (factorization of a matrix into a product of lower and upper triangular matrices)
MD	Minimum Degree (matrix reordering)
MNA	Modified Nodal Admittance
MoM	Method of Moments
MTL	Multiconductor Transmission Line
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
QMF	Quadrature Mirror Filter
RAM	Random Access Memory
RCM	Reverse Cuthill-McKee (matrix reordering)
RF	Radio Frequency
RMS	Root Mean Square
SAMD	Symmetric Approximate Minimum Degree (matrix reordering)
SRCM	Symmetric Reverse Cuthill-McKee (matrix reordering)
SVD	Singular Value Decomposition
TEM	Transverse Electromagnetic Mode
VLSI	Very Large Scale Integration

LIST OF SYMBOLS

This thesis deals with vastly different subjects from such fields as linear analysis, approximation theory, multiresolution analysis, electromagnetics and circuit analysis. Literature covering these fields has established notation with many symbols having different meaning when used in different context. In order to keep notation consistent with the established in the literature, it was decided to reuse some symbols. Scope of such symbols is noted whenever possible.

RELATIONS AND AUXILIARY SYMBOLS

Symbol:	Definition:	Introduced in:
$a \leftrightarrow b$	Statement *a* is equivalent to statement *b*	
$a \rightarrow b$	Statement *b* follows from statement *a*	
$\exists\alpha$	There exist α (such that....)	
$\langle x, y\rangle$	Inner product of vectors *x* and *y*	(2.10)-(2.15)
$\|x\|$	Norm of vector *x*	(2.18)-(2.21)
$\{x_k\}$	Set of vectors x_k	
$[a, b, c]$	Vector consisting of scalars *a*, *b*, *c*	
$[T_{k,l}]$	Matrix *T* containing elements $T_{k,l}$ at *k*-th row and *l*-th column	
$a \in S$	*a* is a member of set *S*	
$a \subset S$	*a* is a subset of set *S*	
$M^{\perp}$	Orthogonal complement to *M*	page 13

$\overline{\bigcup V_j}$	Union of sets $\{V_j\}$
$\overline{\bigcap V_j}$	Intersection of sets $\{V_j\}$
$V \oplus W$	Direct sum of sets V and W
$x \Rightarrow \alpha$	Transformation of x into α
A^T	Matrix A transposed
A^{-1}	Matrix A inverted
∇^2	Operator $\nabla^2 = \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$
$i = \overline{M \dots N}$	Variable i takes all integer values from M to N inclusive
(a, b)	Interval on the real axis from a to b

SYMBOLS BASED ON LATIN ALPHABET

Symbol:	Definition:	Introduced in:
C	A constant	
C	Set of all complex numbers (Chapter 2)	
C	Capacitance matrix (Section 4.3)	(4.25)
C	Imaginary admittance matrix in MNA equations (Chapters 3 and 4)..	(3.1)
$\hat{C}$	Imaginary admittance matrix in expanded MNA equations (Chapters 3 and 4)	(3.5)
D	Matrix representation of the derivative operator (Chapters 3 and 4)	(3.6)

f	Frequency
$f(x)$	Function of x
$G(z)$	Frequency response of a high pass digital filter (Chapter 2)
$G(r,r')$	Green's function (Section 4.3) .... (4.15)
G	Real admittance matrix in MNA equations (Chapters 3 and 4) .... (3.1)
$\hat{G}$	Real admittance matrix in expanded MNA equations (Chapters 3 and 4) (3.5)
H	Hilbert space .... page 12
$H(z)$	Frequency response of a low pass digital filter (Chapter 2)
h, g	Coefficients of the refinement equation ....(2.41)-(2.50) also known as filter coefficients of the QMFs .... page 37
$\tilde{h}$, $\tilde{g}$	Coefficients of the biorthogonal refinement equation ....(2.41)-(2.50) also known as filter coefficients of the decomposition QMFs .... page 37
J	Highest level of wavelet decomposition (Section 4.3)..... (4.17)
J	Jacobian matrix (Chapters 3 and 4) .... (3.8)
L	Linear operator (Section 2.2) .... page 14
$L^2(0, 1)$	Set of functions, square integrable on interval (0,1) ..... (2.56)
L	Number of vanishing moments for the wavelet (Section 2.4 and onwards) .... (2.68)

M	Length of h and g .... page 26
N	(General) number of something, particularly number of unknowns
N_c	Number of conductors in a multiconductor transmission line (Chapter 4)
N_f	Number of frequencies in truncated grid (Chapter 3)
N_H	Highest order of intermodulation products (Chapter 3)
N_t	Number of time points for analysis (Chapters 3 and 4)
N_x	Number of unknowns in MNA equations (Chapters 3 and 4)
$O(f(N))$	Order of; g = O(f(N)) means that there exist K such that $g(N) \le Kf(N)$. Particularly referring to computational cost in terms of elementary operations, such as additions and multiplications, or units of storage.
$\mathrm{Proj}_{V_j} f$	Projection operator or vector f onto space V_j .... page 13
Q	Matrix of per unit length charges .... (4.24)
r	Radius vector in Cartesian coordinates (Chapter 4) ..... (4.15)
r_m	Connection coefficients for the scaling function (Chapter 3) .. .. (3.21)
$\Re$	Real line .... page 29
$\mathrm{supp}\varphi$	Support of φ, i.e. interval on which φ has nonzero values .... (4.16)
$\tilde{T}$, T	Forward and inverse transform matrices

t	Time
u	Vector of independent sources in MNA equations (Chapters 3 and 4) .... (3.1)
u, v	Generalized orthogonal bases
V_j	Multiresolution approximation space at level j ....(2.55)-(2.60)
W_j	Orthogonal complement of V_j in V_{j-1} .... page 30
x, y, z	Cartesian coordinates (Chapter 4)
x	Vector of unknowns (particularly in Chapters 3 and 4)
Z	Set of integer numbers (including zero and negative numbers)

SYMBOLS BASED ON GREEK ALPHABET

Symbol:	Definition:	Introduced in:
α, β, γ	Connection coefficients (Chapter 3)	(3.18)-(3.20)
γ	Relaxation coefficient for hard thresholding (Section 4.3)	page 128
ε	Thresholding parameter for soft thresholding..........	page 126
σ	Surface charge density	(4.15)
τ	Period of the fundamental frequency in the circuit (Chapter 3)	(3.2)
Φ	Residue of a nonlinear algebraic equation	(3.6)
ϕ	Scalar electrostatic potential	(4.14)

$\varphi_{j,k}(x)$	Scaling function in general; reconstruction scaling function for a biorthogonal family page 19
$\psi_{j,k}(x)$	Wavelet function in general; reconstruction scaling function for a biorthogonal family .... page 19
$\tilde{\varphi}_{j,k}(x)$	Decomposition scaling function for a biorthogonal family ....(2.37)-(2.40)
$\tilde{\psi}_{j,k}(x)$	Decomposition wavelet function for a biorthogonal family ....(2.37)-(2.40)
Ω	Domain consisting of the surface of conductors (Chapter 4) .... (4.15)
Ω	Set of frequencies used for steady state analysis (Chapter 3) .... (3.38)
ω	Angular frequency (Chapter 3)

1. Introduction

1.1 Background and motivation

This thesis is dedicated to the treatment of selected problems in Electronics Design Automation (EDA) in general, and circuit analysis in particular, with the help of wavelets. The broad range of numerical problems arising in EDA field can be systematically treated in the mathematical framework of functional analysis, linear algebra and approximation theory. From the mathematical point of view, most of these problems can be represented in the form of either differential or integral operator equations. With such approach, wavelet methods become very mainstream. In fact, the only difference is that wavelets are just another basis for expansion of operator equations. This basis, however, possesses some special properties that, when taken proper advantage of, can provide algorithms that are significantly superior to the ones utilizing traditional bases in terms of accuracy, speed and memory storage requirements.

Over the past 15 years substantial progress has been made in the development of both wavelet theory and applications. However, among many applications of wavelets, numerical analysis has often been overlooked in favour of signal processing and approximation theory. Moreover, if we look at the numerical applications, EDA problems in particular have not been thoroughly studied. From a number of publications that appeared in the last 10 years, many seem to suffer from the same syndrome “let’s show that this can be also done with wavelets” rather than concentrating on development of fast algorithms that take advantage of wavelet properties. Development of the mathematical theory of wavelets initially caused certain excitement in the scientific computations community, however we will show that not all applications of wavelets result in immediate advantages, particularly in terms of efficiency. Great care should be taken with respect to claims of efficiency. In some cases, established traditional non-wavelet methods are so well researched, engineered and implemented, that direct CPU time comparison does not show particular advantage of wavelet methods.

1.2 Outline of the thesis

This thesis is organized as following.

Chapter 2 contains essential mathematical background from the areas that are necessary to understand the applications described further. We start with overview of linear analysis, Hilbert spaces and operators and proceed to derive wavelet formulation on simple illustrated examples leading to a concept of multiresolution analysis. The chapter concludes with discussion of wavelet properties and Fast Wavelet Transform.

One of the reasons for the slow introduction of wavelets into the engineering curricula is the visible lack of accessible texts, particularly concentrating on the numerical applications. The only one that, in author's opinion, presents theory in a proper manner ([8]) is written by a physicist, not an engineer (although approach is quite similar), and was published by a university in Switzerland, thus not widely known and available. This is the motivation behind inclusion of a rather lengthy wavelet background in Chapter 2. This material should not be viewed as a comprehensive wavelet tutorial, but should contain all the background relevant to the discussed applications.

Chapter 3 considers application of wavelets to the solution of steady state analysis of nonlinear circuits. This problem is described by a nonlinear ordinary differential equation with periodic boundary condition. We abandon the traditional approach of expanding the equation in Fourier basis in favour of wavelet bases. Because of their local support, wavelet bases provide sparse O(N) representation for the Jacobian matrix. Computational cost analysis is performed that shows that wavelet expansion gains significant advantage over traditional approach, particularly for multitone, highly nonlinear, large scale and broadband circuits. Theoretical results are supported by two numerical case studies.

Chapter 4 addresses other applications of wavelets to the analysis of lumped and distributed parameter circuits. First, we review transient analysis of nonlinear lumped parameter circuits and conclude that in this area it is extremely difficult for wavelet methods to compete with established time marching schemes. The rest of the chapter is dedicated to analysis of distributed parameter circuits, particularly quasi-TEM interconnect analysis. Review of the interconnect macromodelling shows that wavelets have potential for analysis of nonuniform transmission lines, but proof of passivity has to be provided before this potential can be realised. The chapter concludes with discussion of the problem of extracting transmission line physical parameters, particularly capacitance matrix, for interconnect simulations.

Chapter 5 summarizes the material presented in this thesis and outlines possible directions for future research.

1.3 Original contribution

The primary original contribution of this thesis is in the development of wavelet technique for steady state analysis of nonlinear circuits. The principal disadvantage of the traditional Harmonic Balance technique is in the appearance of dense blocks in Jacobian matrix, which dominate O(N^4) computational complexity of the problem. Harmonic Balance formulation quickly becomes too expensive to use for highly nonlinear and multitone circuits. The wavelet method developed in this thesis provides O(N) sparse Jacobian by construction and although it does not show particular advantages over traditional methods for single tone simulations and is slower than traditional methods for mild to medium nonlinearities due to primitive spectrum truncation technique, it shows significant advantage for highly nonlinear and multitone circuits. Wavelet method can be further accelerated by developing time- and/or frequency domain adaptive schemes. In addition, it opens the whole new area of exploring steady state methods in bases other than Fourier series.

The secondary contribution of this thesis is in the area of capacitance extraction. Application of wavelets to the capacitance extraction has been known before from the literature. The original contribution here is in the exploitation of the idea that surface charge on the conductor can be computed accurately without accurate computations of the charge distribution itself. This allows development of the hard thresholding concept and it's successful application to extraction problems. Wavelet extraction methods with hard thresholding have potential of successfully competing with the best available techniques for extraction of physical parameters.

2. BACKGROUND: WAVELETS

2.1 LINEAR ANALYSIS

Linear analysis is the field of mathematics that is cornerstone to the numerical problems arising in EDA industry. This section is intended to provide a brief, accessible, yet rigorous background on the linear spaces, vectors, projections and linear operators. Theory presented here will be extensively used later in this chapter when we will proceed to the wavelet analysis as well as in subsequent chapters dealing with applications of wavelets to various computationally extensive EDA problems.

There is a vast amount of books covering linear analysis. Definitions and discussions presented here are for the most part based on an excellent text [1], which can be consulted for more detailed presentation of the material.

2.1.1 LINEAR SPACE

The set S containing vectors a, b, c,... is a linear space if the following rules for addition (2.1)-(2.4) and multiplication (2.5)-(2.8) apply:

$$(a+b)+c = a+(b+c)\,. \tag{2.1}$$

There exists a zero vector $0 \in S$ such that

$$a + 0 = 0 + a = a. \tag{2.2}$$

For every $a \in S$, there exists $-a \in S$ such that

$$a + (-a) = (-a) + a = 0. \tag{2.3}$$

$$a + b = b + a. \tag{2.4}$$

Multiplication rules (α and β are scalars):

$$\alpha(\beta a) = (\alpha\beta)a \tag{2.5}$$

$$1a = a \tag{2.6}$$

$$\alpha(a + b) = \alpha a + \alpha b \tag{2.7}$$

$$(\alpha + \beta)a = \alpha a + \beta a \tag{2.8}$$

Linear space S is said to have dimension n if it possesses a set of n independent vectors and if every set of n+1 vectors is dependent. If for every positive integer n we can find n independent vectors in S, then S has infinite dimension.

Basis of linear space S is a set of independent vectors $\{\varphi_k\}$, such that for any vector $x \in S$

$$x = \sum_{(k)} \alpha_k \varphi_k \tag{2.9}$$

The representation (2.9) with respect to a given basis is unique.

2.1.2 INNER PRODUCT

S is called an inner product space if for every ordered pair of vectors $(x, y) \subset S$, there exist a unique complex scalar denoted by $\langle x, y\rangle$ that satisfies all of the following conditions:

$$\langle x, y\rangle = \overline{\langle y, x\rangle} \tag{2.10}$$

where overbar indicates complex conjugate.

$$\langle x + y, z\rangle = \langle x, z\rangle + \langle y, z\rangle \tag{2.11}$$

$$\langle \alpha x, y\rangle = \alpha\langle x, y\rangle \text{, for all } \alpha \in C \tag{2.12}$$

$$\langle x, x\rangle \geq 0 \text{ with equality if and only if } x=0 \tag{2.13}$$

Real inner products are also of significant practical interest. For them, the first condition simplifies to:

$$\langle x, y\rangle = \langle y, x\rangle . \tag{2.14}$$

On interval (α, β) inner product is usually computed as a weighted integral:

$$\langle f, g\rangle = \int_{\alpha}^{\beta} w(\zeta)f(\zeta)g(\zeta)d\zeta . \tag{2.15}$$

Throughout this work the weighting function $w(\zeta)$ will be assumed identical to 1 unless explicitly noted otherwise.

2.1.3 ORTHOGONALITY

The concepts used throughout this work involve the notions of orthogonality and orthonormality.

Two vectors x and y are orthogonal if their inner product is zero:

$$\langle x, y \rangle = 0 \tag{2.16}$$

Set of basis functions $\{\varphi_k\}$ is orthogonal if the following is true:

$$\langle \varphi_i, \varphi_j \rangle = \delta > 0 \text{ if and only if } i = j \text{ and zero otherwise.} \tag{2.17}$$

Whenever δ in (2.17) is equal to 1, such basis is also called *orthonormal*.

2.1.4 CONVERGENCE IN A NORMED LINEAR SPACE

A linear space S is called a normed (equipped with a norm) linear space if, for every vector $x \in S$, there is assigned a unique real number $\|x\| \in R$ such that the following rules apply:

$$\|x\| \geq 0 \text{ with equality if and only if } x = 0 \tag{2.18}$$

$$\|\alpha x\| = |\alpha| \|x\| \text{ where } \alpha \text{ is an arbitrary scalar} \tag{2.19}$$

$$\|x_1 + x_2\| \le \|x_1\| + \|x_2\| \tag{2.20}$$

There exist many different definitions of the norm, but by far the most popular is the norm induced by the inner product:

$$\|x\| = \sqrt{\langle x, x \rangle} \tag{2.21}$$

One of the important consequences of introduction of the normed linear spaces is that a norm provides a measure of the “closeness” of one vector to another. One can note from rule (2.18) that $\|x - y\| = 0$ if and only if x and y are the same vector. Therefore, closeness between x and y can be mathematically indicated as $\|x - y\| < \varepsilon$.

This observation brings up a concept of convergence. Among many forms of convergence, there are two that are fundamental for establishing concrete “boundaries” on the linear space. The type of boundary that is necessary is the one that ensures that the limit on a vector sequence in a linear space is also contained in that space.

In a normed linear space S, a sequence of vectors $\{x_k\}\big|_{k=1}^{\infty}$ *converges* to a vector $x \in S$ if, given an arbitrarily small number $\varepsilon > 0$, there exist a number N such that $\|x - x_k\| < \varepsilon$ whenever $k > N$. Convergence of x_k to x is usually denoted as $x_k \to x$ or

$$\lim_{k \to \infty} x_k = x \tag{2.22}$$

We can also note here that due to the continuity of the inner product, order of application of the limit and the inner product to a sequence converging is S is interchangeable:

$$\lim_{k \to \infty} \langle x_k, h \rangle = \langle \lim_{k \to \infty} x_k, h \rangle \tag{2.23}$$

The notion of convergence introduced above is a strict one and it also implies *Cauchy convergence*, which is defined as following.

In S, a sequence $\{x_k\}\Big|_{k=1}^{\infty}$ converges in Cauchy sense if, given an arbitrarily small number $\varepsilon > 0$, there exist a number N such that $\|x_m - x_n\| < \varepsilon$ whenever $min(m, n) > N$:

$$\lim_{m, n \to \infty} \|x_m - x_n\| = 0 \tag{2.24}$$

Convergence in Cauchy sense is a weak form of convergence and in it's turn does not necessarily imply strict convergence in sense of (2.22). In other words, it is possible for two vectors of the sequence to become arbitrarily close to each other without the sequence itself approaching a limit in S. This observation leads to the introduction of *complete* spaces, in which Cauchy convergence does imply strict convergence.

A normed linear space is said to be complete if every Cauchy sequence in the space converges to a vector in that space.

2.1.5 HILBERT SPACE

A linear space is called a *Hilbert space* if it is complete in the norm induced by it's inner product. Therefore, in any Hilbert space, Cauchy convergence implies convergence.

In numerical analysis we are often concerned with subsets of vectors in a linear space. Such subsets are obtained by approximate solutions of the differential and integral equations describing the problem, be it initial value problem or boundary value one.

One such subset of vectors is called a *linear manifold*. If S is a linear space and α, β are arbitrary scalars, then M is a linear manifold of S, provided that $\alpha x + \beta y \in M$ whenever $x, y \subset M$. It is easy to show that M is also a linear space and inherits many of the properties of S. One can view a linear manifold as a "subspace" in S, e.g. one spanned by a finite dimensional subset of basis vectors in an otherwise infinitely dimensional space. A linear manifold is *closed* if it contains the limits of all sequences that can be constructed from it's members. Therefore, a closed linear manifold in a Hilbert space is a Hilbert space on it's own.

Within the framework of Hilbert spaces, it is possible to construct a general approach to approximation of vectors and functions. Let x be a vector in Hilbert space H and let $\{u_k\}\big|_{k=1}^{m}$ be an orthonormal set in H:

$$x_m = \sum_{k=1}^{m} \alpha_k u_k . \qquad (2.25)$$

“Best approximation” $\{\alpha_k\}\big|_{k=1}^{m}$ of x on $\{u_k\}\big|_{k=1}^{m}$ is then given by

$$\alpha_k = \langle x, u_k \rangle \qquad (2.26)$$

which also constitutes projection of x onto a closed linear manifold H_m, a Hilbert space spanned by $\{u_k\}\big|_{k=1}^{m}$. As a consequence of this, best approximation performed according to (2.26) also produces a residue vector $e_m = x - x_m$ which is orthogonal to x_m.

The above results for the approximation of a vector $x \in H$ by a vector $x_m \in H_m \subset H$ can be generalized. We will need a concept of manifold that is orthogonal to a given manifold. If M is a linear manifold in H, then a vector $e \in H$ is a member of set $M^{\perp}$ if it is orthogonal to every vector in M. The set $M^{\perp}$ is also a linear manifold since linear combinations of vectors in $M^{\perp}$ are also orthogonal to vectors in M. In fact, $M^{\perp}$ is also closed. The closed linear manifold $M^{\perp}$ is then called the *orthogonal complement* to M.

2.1.6 Projection theorem

The concept of “best approximation” introduced above can now be formulated rigorously using the notion of the orthogonally complementary spaces. This formulation is also know as the Projection Theorem which is fundamental to the approximation theory and numerical methods for solution of operator equations.

Let x be any vector in a Hilbert space H, and let $M \subset H$ be a closed linear manifold in H. Then, there is a unique vector $y_k \in M \subset H$, closest to x in the sense that $\|x - y_k\| \le \|x - y\|$ for any vector y in M. Furthermore, the necessary and sufficient condition that y_k is the unique vector minimizing $\|x - y_k\|$ is that the approximation residue defined as $e = x - y_k$ is contained in the orthogonal complement $M^{\perp}$ of the projection space M.

2.2 Linear operators in Hilbert space

2.2.1 Matrix formulation

Let S be a linear space. An operator L is a mapping that assigns a vector $x \in S$ another vector $y \in S$:

$$Lx = y \tag{2.27}$$

The operator L is linear if the mapping (2.27) is such that for any two vectors x and y in the domain of L, their linear combination $\alpha x + \beta y$ is also in the domain of L and the following holds:

$$L(\alpha x + \beta y) = \alpha L x + \beta L y \tag{2.28}$$

Equation (2.27) can be solved for x. First, we write a best approximation for x in basis $\{u_k\}\big|_{k=1}^{m}$:

$$x = \lim_{m \to \infty} \sum_{k=1}^{m} \alpha_k u_k \tag{2.29}$$

$$Lu = L \lim_{m \to \infty} \sum_{k=1}^{m} \alpha_k u_k = \lim_{m \to \infty} \sum_{k=1}^{m} \alpha_k L u_k \tag{2.30}$$

take the inner product of both sides with another basis $\{v_l\}\big|_{l=1}^{m}$:

$$\left\langle \lim_{m \to \infty} \sum_{k=1}^{m} \alpha_k L u_k, v_l \right\rangle = \langle y, v_l \rangle \tag{2.31}$$

$$\lim_{m \to \infty} \sum_{k=1}^{m} \alpha_k \langle L u_k, v_l \rangle = \langle y, v_l \rangle \tag{2.32}$$

$$\left[\langle L u_k, v_l \rangle\right]\left[\alpha_k\right] = \left[\beta_l\right] \tag{2.33}$$

Equation (2.33) is a matrix equation for unknown column vector $\left[\alpha_k\right]$ and can be solved using available techniques. Matrix A, defined as

$$\left[A_{kl}\right] = \left[\langle L u_k, v_l \rangle\right] , \tag{2.34}$$

is a projection of operator L onto finite dimensional closed linear manifold spanned by $\{u_k\}\big|_{k=1}^{m}$ and $\{v_l\}\big|_{l=1}^{m}$. The process of approximating a linear operator with a matrix is also know as discretization of the operator.

Bases $\{u_k\}$ and $\{v_l\}$ can be the same or they can be different. Depending on $\{u_k\}$ and $\{v_l\}$, (2.34) can result in different well-known discretization schemes, such as Method of Moments [2], Galerkin method [2], Finite Differences [3] and Finite Elements [4].

For example, if $\{u_k\}$ and $\{v_l\}$ are interpolating polynomials on an interval and order of $\{u_k\}$ polynomial is 2 more than the order of $\{v_l\}$ polynomial, the discretization scheme becomes Finite Differences. For arbitrary $\{u_k\}$ and $\{v_l\}$ the method is usually referred to as Method of Moments, and for $\{u_k\} = \{v_l\}$ as Galerkin method. Basis set $\{u_k\}$ is called *expansion* functions while basis set $\{v_l\}$ is usually referred to as *weighting* or *testing* functions.

If $\{u_k\}$ are eigenvectors of L, the resulting matrix A is diagonal with eigenvalues of L on the main diagonal.

2.2.2 NEED FOR NEW BASES

Efficiency of all discretization techniques primarily depend on how “good’ or “bad” a basis is. The obvious optimal basis for discretization is the eigenfunctions of operator L. However, this involves solution of the eigenvalue problem which is cost prohibitive for most practical cases. A “good” basis should be:

- easy to handle, i.e. to generate and to compute inner products;
- provide *sparse* representation of the operator;

- provide well conditioned representation of the operator or allow efficient preconditioning.

One of the principal disadvantages for FEM/FD as well as for traditional MoM/Galerkin schemes is the need to regenerate the bases $\{u_k\}$ and $\{v_l\}$ and recompute matrix A (2.34) whenever a problem does not converge to desirable accuracy with m basis functions. Grid refinement techniques can be applied instead of regeneration in some cases, but a systematic and generalized way of refinement must be established. Fourier basis as well as different families of orthogonal polynomials provide means for refinement without regenerating the basis, but they don't have local support, which in many cases leads to a dense matrix.

Whenever L, x and/or y in equation (2.27) have frequency content that is localized in space/time, different degrees of resolutions in different regions must be applied for accurate representation at low cost. Windowed Fourier transform can provide necessary localization framework for such scenario. However, the problem with the windowed Fourier transform is that for typical real-life signals and functions, high frequency components would have a short timespan and won't be properly localized with a certain window size, while the low-frequency components would have a long timespan and won't be captured by the same window at all. A work on exploring a variable window width transform led to construction of one of the first families of wavelets, *modulated Gaussian* or *morlets* [5].

A generally “optimal” basis should retain advantages of the FD/FEM (local support, sparse representation) with the orthogonality of resolution levels inherent to Fourier series and orthogonal polynomials.

2.3 Wavelet concepts

2.3.1 From pulse functions to Haar wavelets

Let us illustrate the idea of orthogonal refinement with an example of approximating a function in basis of pulse functions (Fig. 2.1). Graph on the left represents approximation in rather coarse basis. To improve the approximation, we refine the basis such that each basis function has half the support of the original basis (right graph). This reduces approximation error, however all the approximation coefficients (2.26) have to be completely recalculated in the new refined basis.

Let’s denote space spanned by the original basis $\{\varphi_{j,k}\}$ as V_j and one spanned by the refined basis $\{\varphi_{j+1,k}\}$ as V_{j+1}. The question that arises here is that is it possible to construct a basis $\{\psi_{j,k}\}$ which will complement $\{\varphi_{j,k}\}$ in V_{j+1}? In other words, is it possible to find a set of basis functions that will refine the original approximation without the need to recalculate already existing coefficients?

For such a simple basis it’s indeed trivial to construct a complementary basis (Fig. 2.2). It’s easy to see that $\psi_{j,k}(x)$ complements $\varphi_{j,k}(x)$ to the

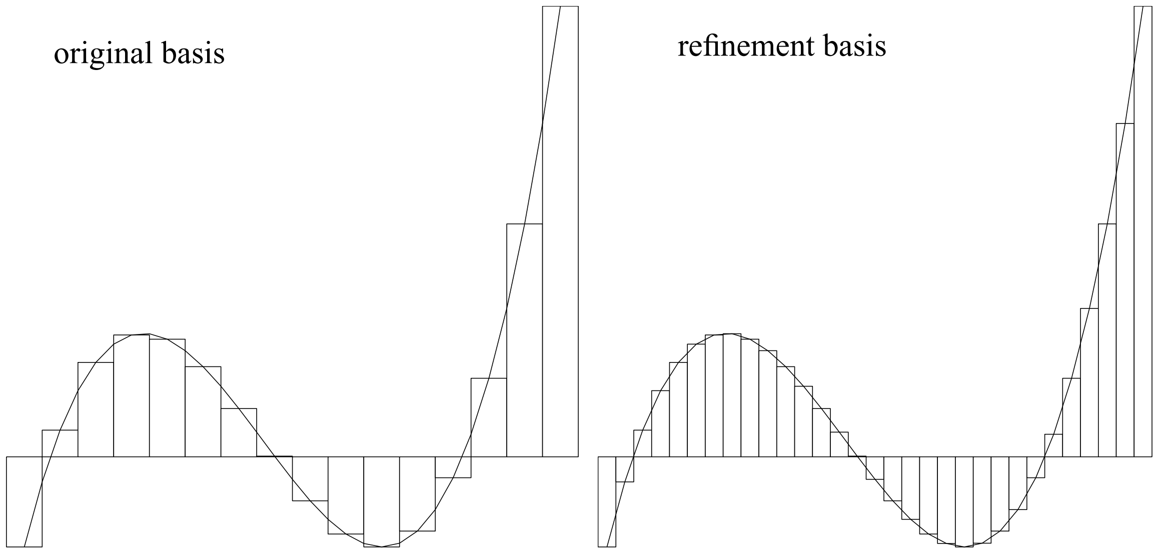


FIGURE 2.1. Refinement in the basis of pulse functions.

refined basis $\varphi_{j+1,k}(x)$. Expansion in new basis will retain all the previously calculated coefficients and only half has to be calculated from scratch.

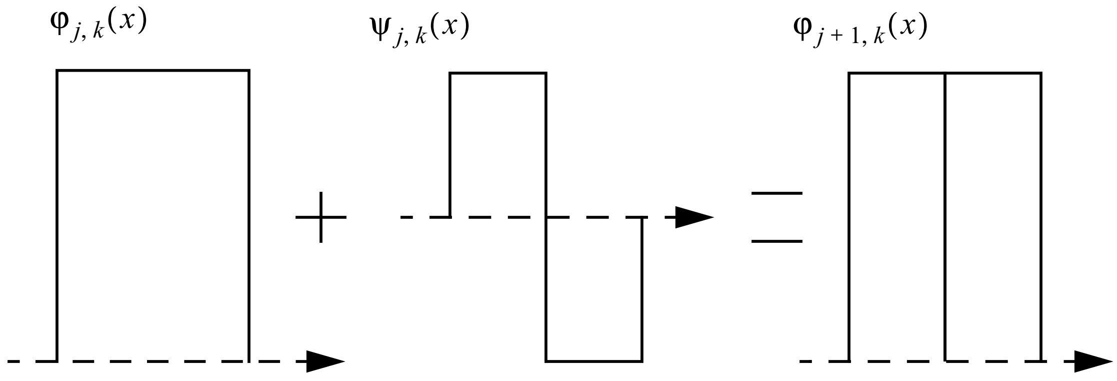


FIGURE 2.2. Refinement through orthogonal complement.

We should also mention here that $\psi_{j,k}(x)$ is not only complementary, but also orthogonal to $\varphi_{j,k}(x)$. This is a very comfortable fact provided that both bases are also orthogonal on their own.

Such basis functions are Haar functions, named after a mathematician who first proposed such construction in 1911 ([5]). However, it was not until 1980s when these functions were incorporated into rigorous mathematical framework of *multiresolution analysis*. Basis functions $\varphi_{j,k}(x)$ that capture average value of the approximated function were given the name of *scaling functions*, while $\psi_{j,k}(x)$ that represent variation were given the name *wavelet functions* or *wavelets*[1].

Though Haar wavelets are not optimal for many applications, their simplicity is useful for illustration of many wavelet concepts on a very intuitive level.

2.3.2 TRANSLATIONS AND DILATIONS.

At this point we have introduced the double indexation of wavelets and scaling function. The index pair (*j*,*k*) reflects the fact that a wavelet basis set (Fig. 2.3) is formed by *translations* and *dilations* of original mother wavelet $\psi_{0,0}$ or scaling function $\varphi_{0,0}$ (Fig. 2.4). The first index *j* refers to the scaling level, while the second index *k* refers to the translated position

1. Literal translation of french *ondelette*.

of this basis function on this particular scaling level, such that $\varphi_{j,k}(x)$ and $\psi_{j,k}(x)$ form orthonormal bases:

$$\varphi_{j,k}(x) = \sqrt{2}^{j}\varphi_{0,0}(2^{j}x - k) \tag{2.35}$$

$$\psi_{j,k}(x) = \sqrt{2}^{j}\psi_{0,0}(2^{j}x - k) \tag{2.36}$$

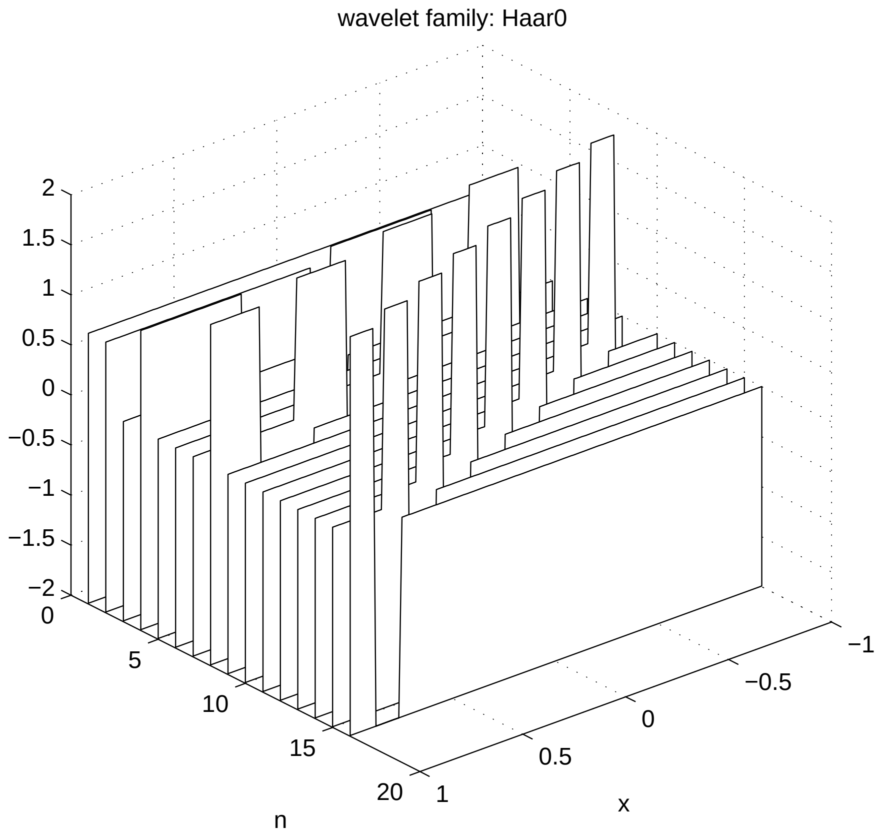


FIGURE 2.3. Haar basis on [-1, 1].

At higher resolution levels, denoted by higher values of j, greater ranges of k are required to span the same interval. In fact, with common dyadic construction of basis given by (2.35)-(2.36), the number of basis functions on each resolution level doubles.

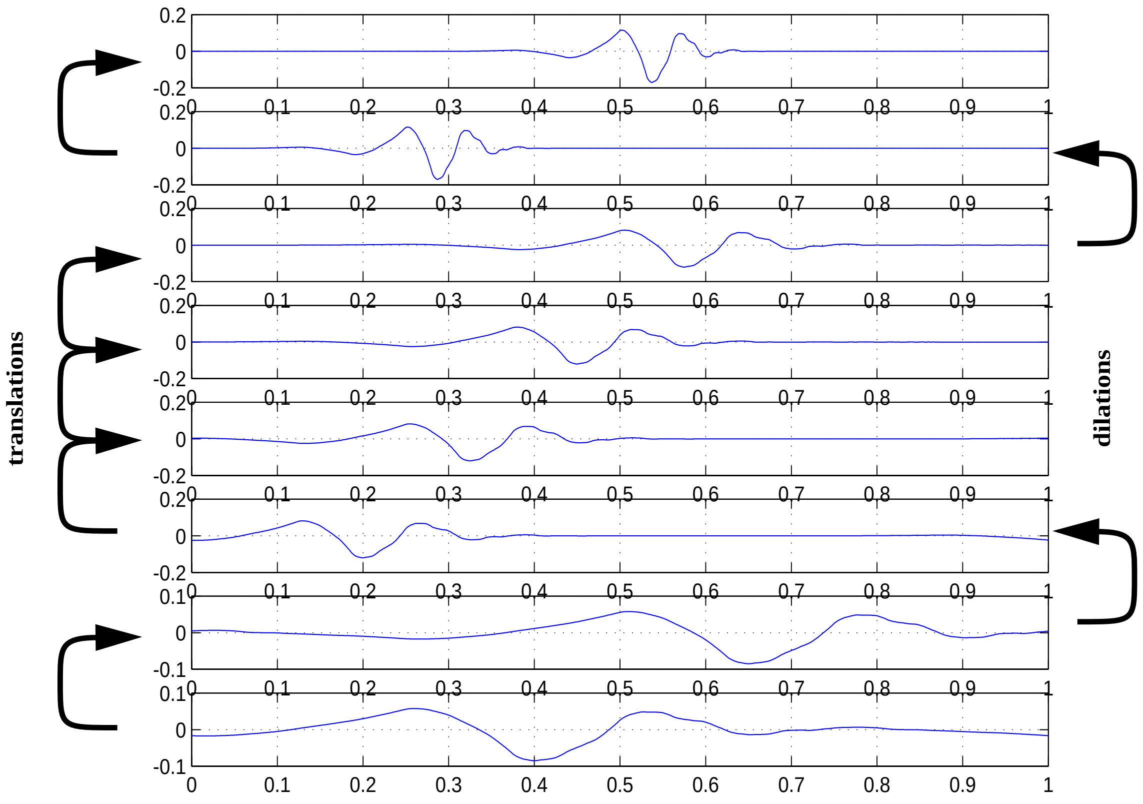


FIGURE 2.4. Translations and dilations of the mother wavelet.

2.3.3 BIORTHOGONALITY.

Bases $\varphi_{j,k}(x)$ in (2.25) and (2.26) are called synthesis basis and analysis basis respectively. In wavelet analysis terms reconstruction and decomposition are also in common use and have essentially the same meaning. In general case, they need not be the same[1]. Analysis basis will then be

1. A well-know in engineering example of biorthogonality is Shannon basis, which consists of delta functions for analysis and sinc functions for synthesis.

denoted $\tilde{\varphi}_{j,k}(x)$, with synthesis basis $\varphi_{j,k}(x)$ as before. Wavelet bases $\tilde{\varphi}_{j,k}(x)$, $\tilde{\psi}_{j,k}(x)$ and $\varphi_{j,k}(x)$, $\psi_{j,k}(x)$ must satisfy *biorthogonality* relations [8]:

$$\langle \tilde{\varphi}_{j,k}, \varphi_{j,q} \rangle = \delta_{kq} \tag{2.37}$$

$$\langle \tilde{\psi}_{i,k}, \varphi_{j,q} \rangle = 0, \quad (i \ge j) \tag{2.38}$$

$$\langle \psi_{i,k}, \tilde{\varphi}_{j,q} \rangle = 0, \quad (i \ge j) \tag{2.39}$$

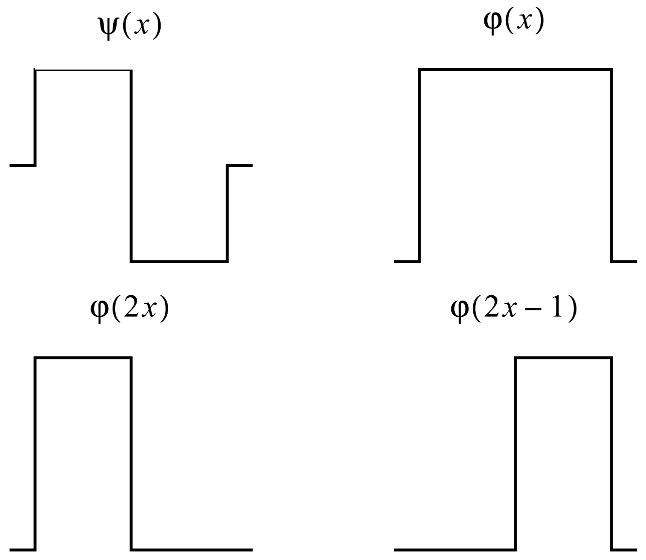
$$\langle \psi_{i,k}, \tilde{\psi}_{j,q} \rangle = \delta_{ij}\delta_{kq} \tag{2.40}$$

$\varphi_{j,k}(x)$ is then called a dual function of $\tilde{\varphi}_{j,k}(x)$ and vice versa. The same applies also to $\psi_{j,k}(x)$ and $\tilde{\psi}_{j,k}(x)$.

2.3.4 Refinement equation.

An important concept of wavelet analysis is the existence of *refinement equation*, which allows representation of basis functions (both scaling functions and wavelets) at level k in terms of scaling functions at level k+1.

As it can be easily seen from Fig. 2.5, refinement coefficients for the Haar basis indeed exist and can be found by observation. In general case we can write refinement equations for biorthogonal bases:



$$\varphi(x) = \frac{1}{2}\varphi(2x) + \frac{1}{2}\varphi(2x-1)$$

$$\psi(x) = \frac{1}{2}\varphi(2x) + \left(-\frac{1}{2}\right)\varphi(2x-1)$$

FIGURE 2.5. Refinement equation for Haar wavelets.

$$\varphi(x) = \sum_{m=-M}^{M} h_j \varphi(2x-m) \tag{2.41}$$

$$\psi(x) = \sum_{m=-M}^{M} g_j \varphi(2x-m) \tag{2.42}$$

$$\tilde{\varphi}(x) = \sum_{m=-M}^{M} \tilde{h}_j \tilde{\varphi}(2x-m) \tag{2.43}$$

$$\tilde{\psi}(x) = \sum_{m=-M}^{M} \tilde{g}_j \tilde{\varphi}(2x-m) \tag{2.44}$$

Refinement coefficients h_j, $\tilde{h}_j$ and g_j, $\tilde{g}_j$ are not independent. They must satisfy symmetry relations:

$$g_{i+1} = (-1)^{i+1}\tilde{h}_{-i} \tag{2.45}$$

$$\tilde{g}_{i+1} = (-1)^{i+1}h_{-i} \tag{2.46}$$

and, in the case of biorthogonal wavelets, biorthogonality relations also hold:

$$\sum_{(l)} h_{l-2i}\tilde{h}_{l-2j} = \delta_{ij} \tag{2.47}$$

$$\sum_{(l)} g_{l-2i}\tilde{g}_{l-2j} = \delta_{ij} \tag{2.48}$$

$$\sum_{(l)} h_{l-2i}\tilde{g}_{l-2j} = 0 \tag{2.49}$$

$$\sum_{(l)} \tilde{h}_{l-2i}g_{l-2j} = 0 \tag{2.50}$$

For orthogonal (as opposed to biorthogonal) wavelets, similar conditions immediately follow from (2.41)-(2.50) by setting $\tilde{h} = h$ and $\tilde{g} = g$.

Rather non-trivial proof of these formulas can be found in [5].

Equations (2.41)-(2.46) together form a system of equations for vectors h and $\tilde{h}$. Each solution uniquely determines a family of wavelets containing $\tilde{\varphi}_{j,k}(x)$, $\tilde{\psi}_{j,k}(x)$ and $\varphi_{j,k}(x)$, $\psi_{j,k}(x)$. It is easy to see that the number of possible solutions depends on the numbers M and $\tilde{M}$ of nonzero coefficients in h and $\tilde{h}$. For $M = \tilde{M} = 2$, equations (2.41)-(2.46) are reduced to a system of 2 independent equations with 2 unknowns that produce a unique orthogonal solution $h = \tilde{h} = \left[\frac{1}{2}, -\frac{1}{2}\right]$. For $M = \tilde{M} = 3$, there are no known orthogonal solutions. For higher values of *M*, number of unknowns exceeds the number of equations and additional restrictions can be imposed to produce a solution that generates wavelets with desirable qualities. For example, for $M = \tilde{M} = 4$, there is one well known orthogonal solution

$$h = \tilde{h} = \left[\frac{1+\sqrt{3}}{4}, \frac{3+\sqrt{3}}{4}, \frac{3-\sqrt{3}}{4}, \frac{1-\sqrt{3}}{4}\right], \qquad (2.51)$$

generating second order Daubechies wavelet (Fig. 2.6) and a biorthogonal solution

$$h = \left[\frac{1}{4}, \frac{3}{4}, \frac{3}{4}, \frac{1}{4}\right],\ \tilde{h} = \left[-\frac{1}{2}, \frac{3}{2}, \frac{3}{2}, -\frac{1}{2}\right] \qquad (2.52)$$

that generates a quadratic spline wavelet (Fig. 2.7) for the reconstruction basis. Several other solutions are possible if we loosen the constraint of $M = \tilde{M} = C$ with a more relaxed one of $M + \tilde{M} = 2C$. For example, with $M = 3$, $\tilde{M} = 5$ and $C = 4$ as before, 2 more solutions are possible (their scaling functions are represented in Figs. 2.8 and 2.9 correspondingly):

$$h = \left[\frac{1}{2}, 1, \frac{1}{2}\right],\ \tilde{h} = \left[-\frac{1}{4}, \frac{1}{2}, \frac{3}{2}, \frac{1}{2}, -\frac{1}{4}\right] \qquad (2.53)$$

$$h = [0, 1, 0],\ \tilde{h} = \left[-\frac{1}{16}, 0, \frac{9}{16}, 1, \frac{9}{16}, 0, -\frac{1}{16}\right] \qquad (2.54)$$

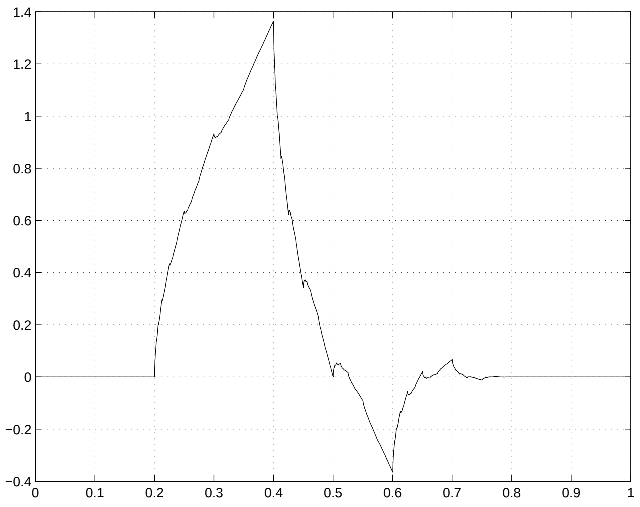


FIGURE 2.6. Scaling function generated by (2.51).

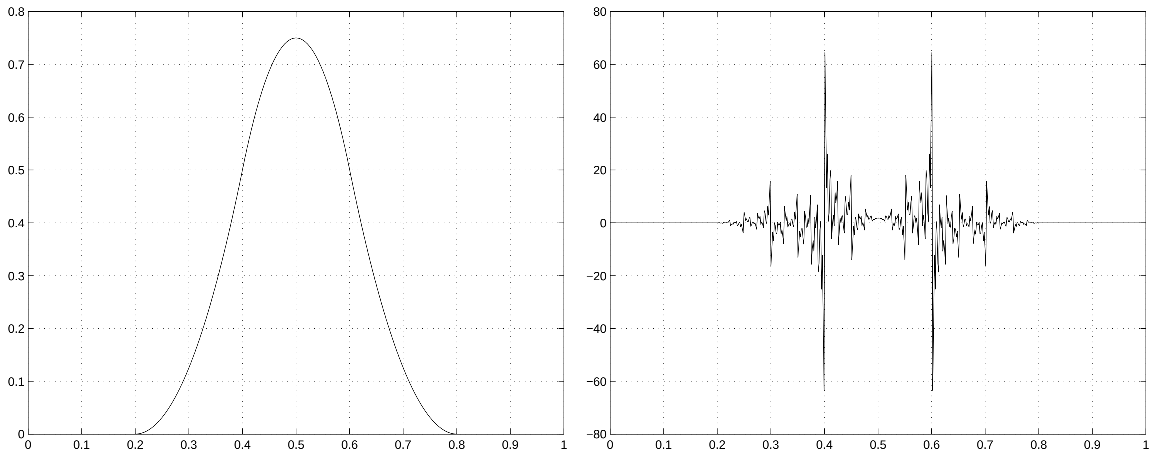


FIGURE 2.7. Biorthogonal scaling functions generated by (2.52). Left: reconstruction basis. Right: decomposition basis.

For higher values of M several orthogonal solution are possible. For example, for M=8 there exist a solution in the Daubechies family (Fig. 2.10) that maximizes the number of vanishing moments for given support width (more on this later) and a solution in the Symlet family (Fig. 2.11) that produces a most “symmetrical” wavelet among all orthogonal solutions.

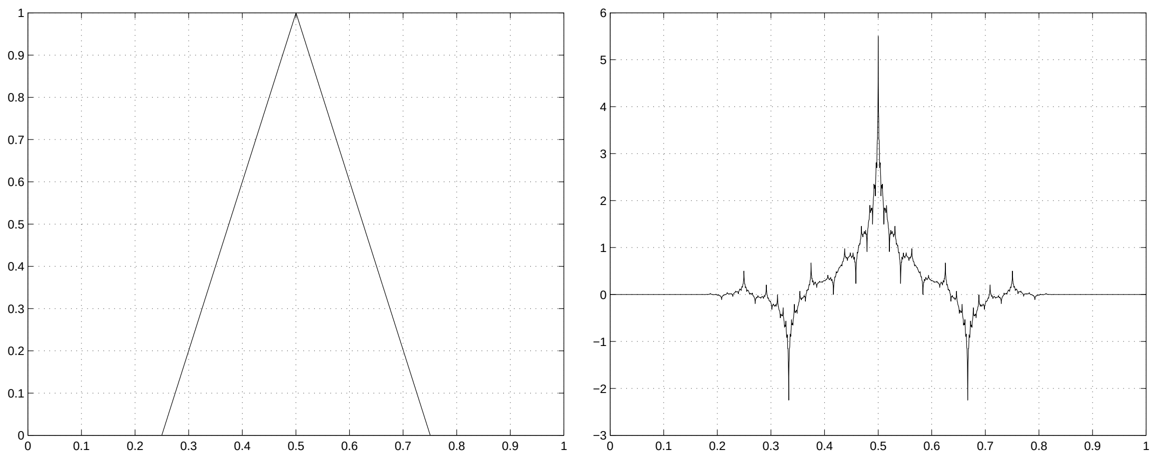


FIGURE 2.8. Biorthogonal scaling functions generated by (2.53). Left: reconstruction basis. Right: decomposition basis.

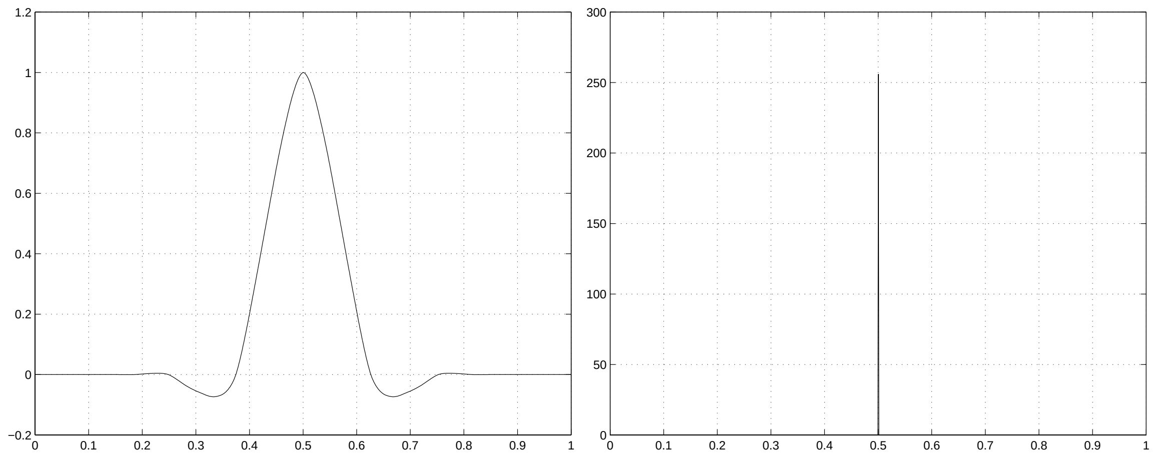


FIGURE 2.9. Biorthogonal scaling functions generated by (2.54). Left: reconstruction basis. Right: decomposition basis.

2.3.5 MULTIRESOLUTION ANALYSIS.

We proceed with the concept of multiresolution analysis [5] on $\Re$ that consists of successive approximation spaces V_j, sometimes also called ladder of spaces:

$$\dots V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset \dots \tag{2.55}$$

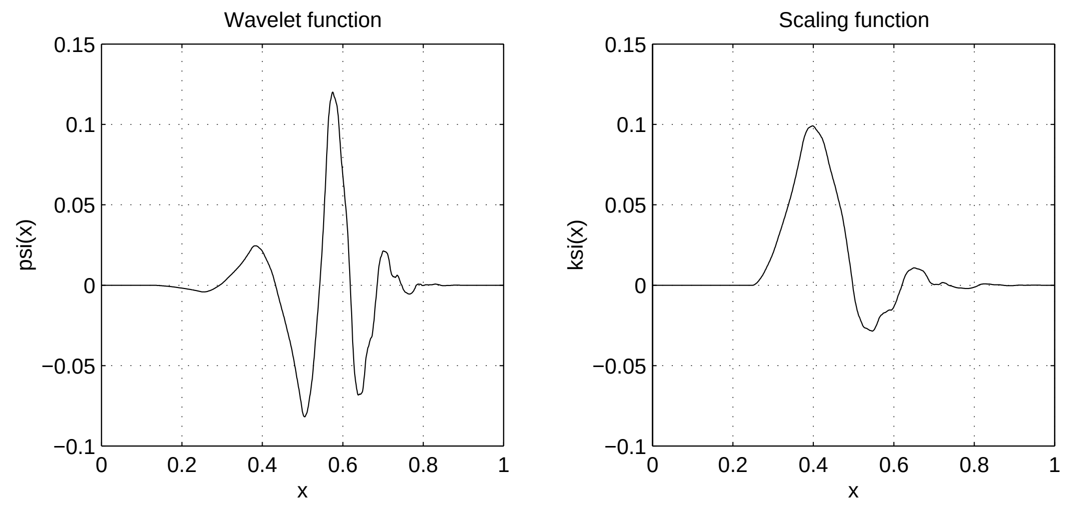


FIGURE 2.10. Daubechies family wavelet and scaling function, *M*=8.

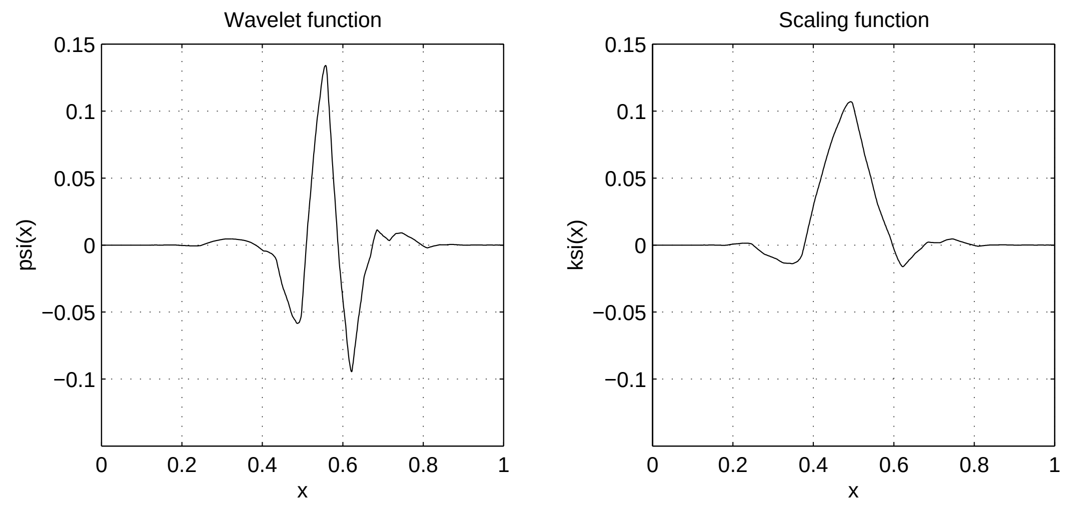


FIGURE 2.11. Symlet family wavelet and scaling function, *M*=8.

$$\overline{\bigcup_{j \in Z} V_j} = L^2(\Re) \tag{2.56}$$

$$\overline{\bigcap_{j \in Z} V_j} = \{0\} \tag{2.57}$$

$$f(\zeta) \in V_j \leftrightarrow f(2^{-j}\zeta) \in V_o \tag{2.58}$$

$$f(\zeta) \in V_0 \to f(\zeta - n) \in V_o \quad \text{for all} \quad n \in Z \tag{2.59}$$

$\exists\varphi \in V_o$ such that $\varphi_{0,n}(x) = \varphi(x-n)$ constitute an orthonormal

basis for V_o (2.60)

Whenever the ladder of spaces V_j satisfies these five properties (2.55)-(2.60), there exist a function ψ such that

$$\text{Proj}_{V_{j+1}} f = \text{Proj}_{V_j} f + \sum_{k \in Z} \langle f, \psi_{j,k} \rangle \psi_{j,k} \tag{2.61}$$

Collection $\{\psi_{j,k}(x) = 2^{j/2}\psi(2^j x - k) \ , \ k \in Z\}$ automatically constitutes an orthonormal basis in W_j which satisfies following conditions:

$$V_{j+1} = V_j \oplus W_j, \tag{2.62}$$

i.e. W_j is an orthogonal complement of V_j in V_{j-1},

$$W_j \perp W_i \quad \text{if} \quad j \neq i, \tag{2.63}$$

i.e., all these subspaces are mutually orthogonal and by virtue of (2.56) and (2.57) allow a decomposition of the space of square integrable functions $\boldsymbol{L}^2(\Re)$:

$$\boldsymbol{L}^2(\Re) = \bigoplus_{j \in Z} W_j, \tag{2.64}$$

and W_j spaces inherit the scaling property (2.58) from the V_j:

$$f(\zeta) \in W_j \leftrightarrow f(2^{-j}\zeta) \in W_0 . \tag{2.65}$$

Functions

$$\psi_{j,k}(x) = 2^{j/2}\psi(2^j x - k) \ , \ k \in Z \tag{2.66}$$

that span $W_j, j \in Z$ are called *wavelet functions*, and corresponding functions that span $V_j, j \in Z$:

$$\varphi_{j,k}(x) = 2^{j/2}\varphi(2^j x - k) \ , \ k \in Z \tag{2.67}$$

are called *scaling functions*.

There also exist other families of wavelets that are not orthogonal and/or do not have scaling functions associated with them, but we will not consider these families for the reasons explained later.

2.4 PROPERTIES OF WAVELETS

2.4.1 LOCAL SUPPORT

By limiting the number of nonzero refinement coefficients in (2.41)-(2.46), we ensure that the wavelets and scaling functions generated by them have *local support*. Local support means that $\tilde{\varphi}_{j,k}(x)$, $\tilde{\psi}_{j,k}(x)$ and $\varphi_{j,k}(x)$, $\psi_{j,k}(x)$ are all identical to zero outside of a closed interval. This is a very important fact because wavelets not only generate a ladder of approximation spaces like Fourier basis and orthogonal polynomials, but also each basis function (be it scaling or wavelet function) is responsible for approximation of a vector only on a small interval, like the truncated on interval polynomials used in FEM.

Because support is directly linked to the number of nonzero refinement coefficient, it is also responsible for the “degree of freedom” we have while constructing wavelets. As the refinement vectors h and $\tilde{h}$ get longer, more solutions of the refinement equations (2.41)-(2.50) are possible and therefore more restrictions can be imposed on them to produce a wavelet family with desirable properties. This means that support width can be viewed as a resource we can spend to construct a proper wavelet. The trade-off for a wavelet family with wider support is the decrease in localization ability.

Orthogonal wavelets necessarily have identical h and $\tilde{h}$ and therefore, identical support and identical properties for both decomposition and reconstruction bases.

2.4.2 Regularity and vanishing moments

One can mention from the above examples that wavelets can be rather unsmooth functions. Haar wavelet is piece wise constant and therefore is not differentiable everywhere. Daubechies wavelets (e.g. Fig. 2.6) do not have a well defined derivative, but they can be characterized by a certain degree of *regularity*. Regularity here is understood as a measure of the smoothness of a function. Regularity of α implies that the α-th derivative of a wavelet belongs to $L^2(\Re)$, i.e. is a square-integrable function. Regularity directly corresponds to how fast frequency spectrum of the wavelet vanishes towards higher frequencies.

Other important property of wavelets is the existence of *vanishing moments*: wavelet $\psi_\lambda(x)$ is said to have L vanishing moments if the following holds:

$$\int x^l \psi_\lambda(x) dx = 0 \quad \text{for} \quad l = \overline{0 \ldots L} \qquad (2.68)$$

High number of vanishing moments also corresponds to how fast frequency spectrum of the wavelet vanishes towards lower frequencies.

Scaling function may (e.g. for Coiflets) or may not (e.g. for Daubechies wavelets) have higher order moments equal to zero.

Unfortunately, all of the above properties can not be achieved simultaneously. If we limit ourselves to local support wavelets, we can not get *infinite regularity* - such wavelets should necessarily have at least *almost* local support (e.g. exponential decay). For a given support width, one can construct either a wavelet with highest number of vanishing moments (Daubechies wavelets), which is highly irregular and asymmetry is well pronounced, or a wavelet with least asymmetry (symlet), which will have significantly smaller number of vanishing moments.

Support width directly translates into how well frequency spectrum of a given wavelet is localized. The wider is the support, the better such wavelet is localized in frequency domain (Fig. 2.12). This example clearly illustrates the property of time-frequency localization. Wavelets have zero average and therefore have no frequency content at zero frequency. Wavelets with more coefficients also have more vanishing moments ($N/2$ vanishing moments for Daubechies family with N nonzero coefficients and $2N$-1 support width). More vanishing moments translates to faster decay of wavelet spectrum in low frequency range. One can also see that higher order wavelets with wider support are fairly smooth and exhibit increasingly oscillatory behaviour which translates to faster decay of the frequency content towards the higher frequencies.

Orthogonal wavelets necessarily have identical h and $\tilde{h}$ and therefore, identical support for both and identical properties for decomposition and reconstruction bases. Sometimes it becomes advantageous to put emphasis on different properties for the decomposition and reconstruction wavelets. For example, more vanishing moments lead to more effective approximation, while greater regularity of the reconstruction basis results in smooth synthesis of the approximated function. Biorthogonal wavelets can provide exactly that additional degree of freedom. Earlier examples of spline interpolating wavelets (Figs. 2.7-2.9) illustrate this observation. Within the same “sum of support” for decomposition and reconstruction

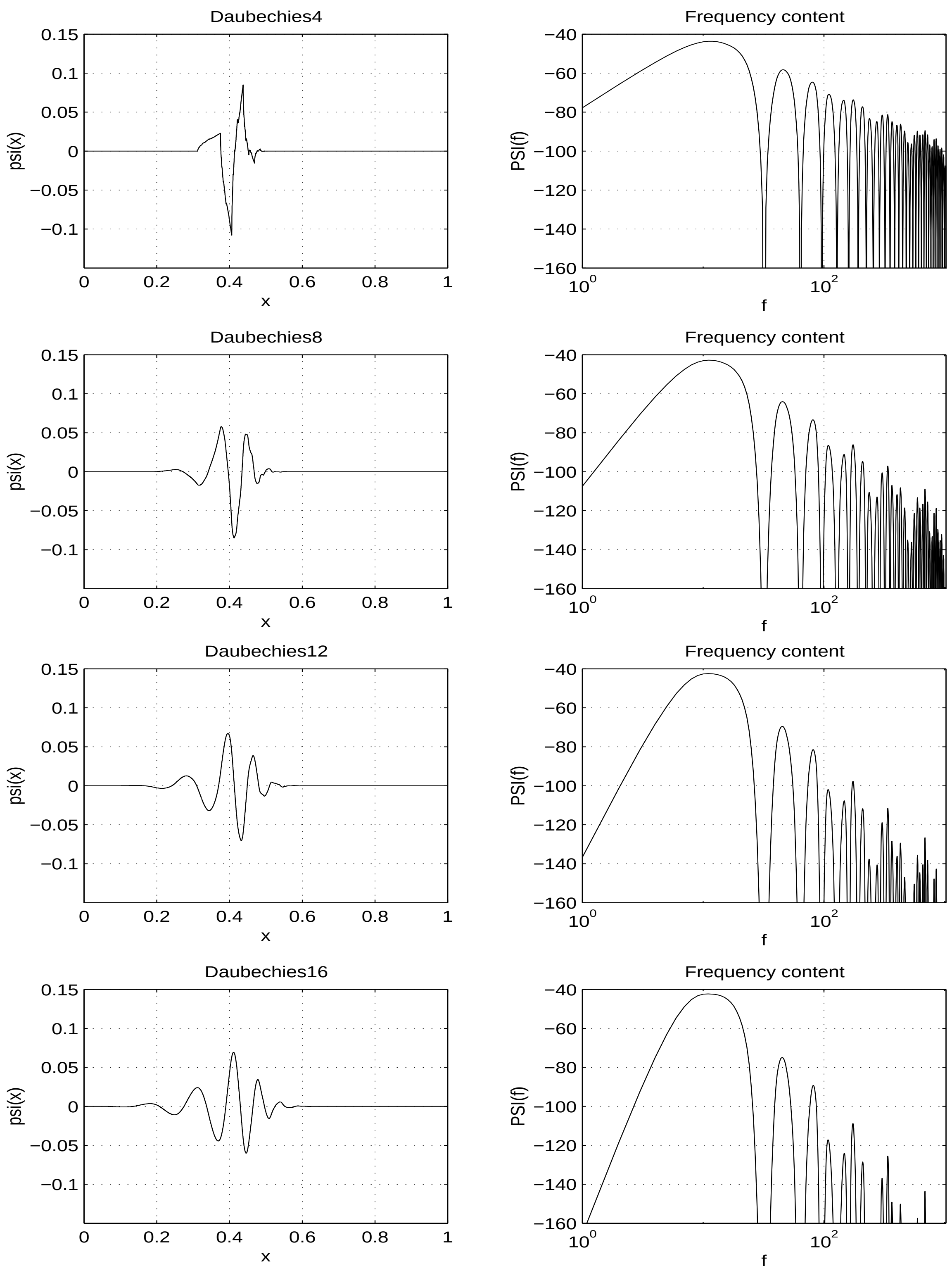


FIGURE 2.12. Time-frequency localization. Daubechies wavelets with 4, 8, 12 and 16 coefficients and their frequency content.

scaling functions, we can split this resource in different ways giving more regularity to the reconstruction basis at the expense of decomposition.

2.4.3 Fast wavelet transform

In order for an orthogonal transform to be numerically efficient, there should exist a fast algorithm of computing the expansion coefficients (2.26). In general, $\{x_k\}\Big|_{k=1}^{N} \Rightarrow \{\alpha_k\}\Big|_{k=1}^{N}$ transform is equivalent to multiplication of vector $[x_k]$ by an $N \times N$ transform matrix *T*:

$$[\alpha_k] = [T_{k,l}] \cdot [x_l] \tag{2.69}$$

Computing (2.69) in general case requires O(N^2) operations. If matrix *T* has a special structure, faster algorithms exist, such as O(*N*log*N*) FFT algorithm for Fourier Transform.

We start construction of the Fast Wavelet Transform (FWT) with an example of computing (2.69) in Haar basis with one level of scaling functions and one level of wavelets ($\varphi(\zeta)$, $\psi(\zeta)$ in Fig. 2.5 and their dilations along *x* axis). Computing approximation coefficients corresponding to the scaling functions $\varphi_i(\zeta) = \varphi(\zeta - i)$ requires computing of a moving average of vector *x* over two samples. This is equivalent to convolving *x* with a Finite Impulse Response (FIR) digital filter with filter taps equal to $h = \left[\frac{1}{2}, \frac{1}{2}\right]$ [7]. Similarly, wavelet coefficients can be computed by convolving *x* with the FIR $g = \left[-\frac{1}{2}, \frac{1}{2}\right]$ (Fig. 2.13).

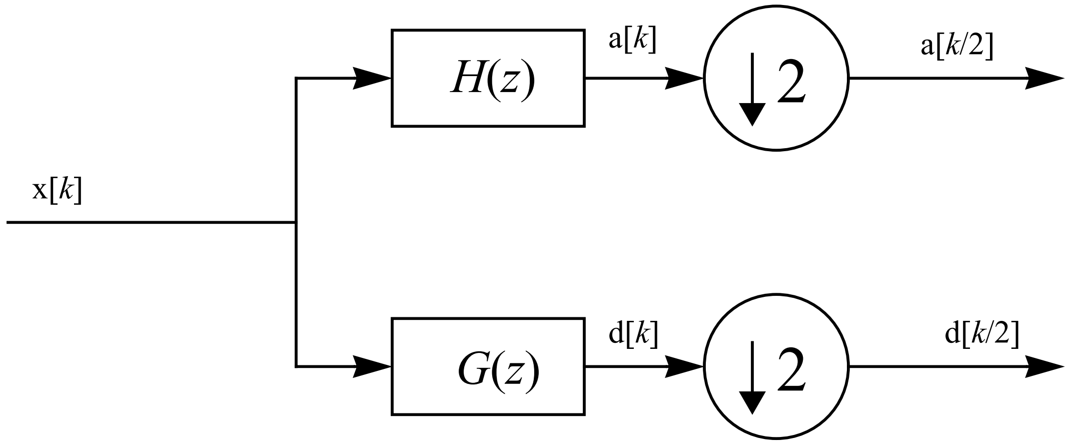


FIGURE 2.13. Computing one level of FWT with FIR filter bank.

For each k samples of the input vector x, each of the LPF $H(z)$ and HPF $G(z)$ will compute k samples of the approximation vector a[k] and k samples of the detail vector d[k] respectively. This is twice as much information as compared to what we feed to the input. This redundancy occurs because the filters are computing twice as many approximation coefficients as required by (2.26). In fact, because dilations of Haar wavelets have zero overlap (Fig. 2.3), we only need every other entry of a[k] and d[k]. This also makes sense from the signal processing point of view: the initial spectrum of x[k] is being split into two complementary parts by the pair of filters, such that each part has half the original bandwidth. This means that the sampling frequency of a[k] and d[k] can be lowered 2 times by a process called *downsampling* without any loss of information. The downsampling is performed by the *decimators* that are merely discarding every other sample appearing at the input. At the output the system produces two vectors half the length of the original sequence. These two vectors combined form together the $\{\alpha_k\}\big|_{k=1}^{N}$ in (2.69).

One can notice that the filter taps for LPF $H(z)$ and HPF $G(z)$ in above example are the same as the refinement coefficients (2.41)-(2.44) for Haar wavelets. This is not just a coincidence and holds not only for Haar wavelets, but for any wavelet family satisfying biorthogonality conditions (2.41)-(2.46). These conditions have a simple equivalent in digital domain called *perfect reconstruction conditions* [6]. Consider system in Fig. 2.13 complemented by a similar setup performing inverse transform (Fig. 2.14).

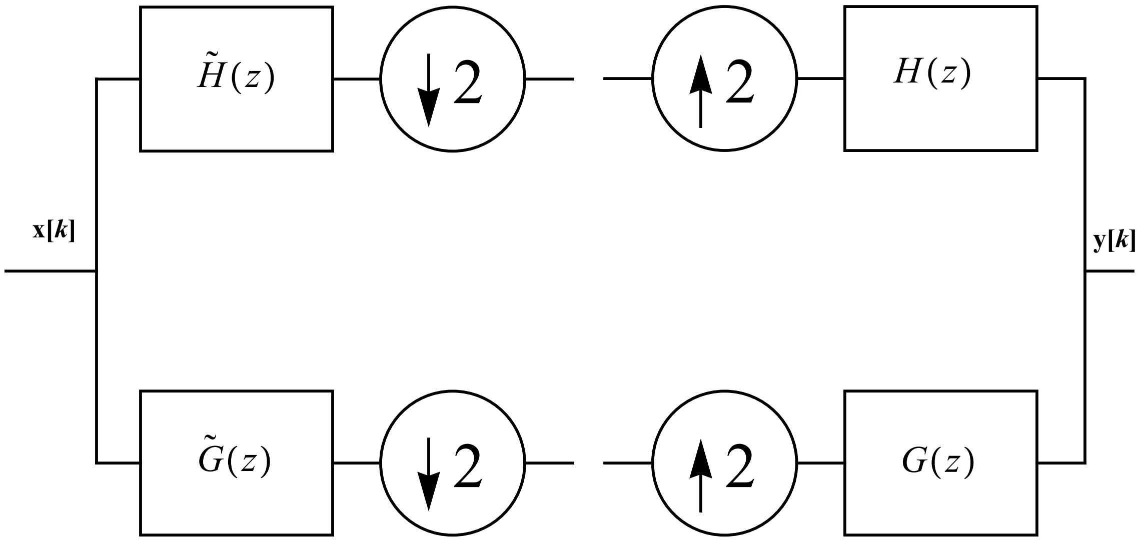


FIGURE 2.14. Perfect reconstruction filter bank.

The inverse transform filter bank consists of two upsamplers introducing zero samples into the sequence and two reconstruction FIR filters. In order for the output sequence y[k] to be identical to the input sequence x[k] with only possible delay, the filter banks must satisfy two conditions:

$$\text{No distortion condition: } \tilde{H}(z)H(z) + \tilde{G}(z)G(z) = 2z^{-1} \qquad (2.70)$$

$$\text{Alias cancellation condition: } \tilde{H}(z)H(-z) + G(z)\tilde{G}(-z) = 0 \qquad (2.71)$$

One can show [6] that filters banks with filter coefficients satisfying (2.41)-(2.46) also satisfy these two conditions written in z domain. Such filters are called *Quadrature Mirror Filters* (QMF).

For numerical methods, it is also convenient to be able to explicitly write transform matrix T for (2.69). If we define filter coefficients for an arbitrary orthogonal wavelet family as $h = [h_0, h_1, h_2, \ldots, h_{M-1}]$ and $g = [g_0, g_1, g_2, \ldots, g_{M-1}]$, then one can write [9]:

$$[\alpha_k] = \begin{bmatrix} a_1 \\ d_1 \\ a_2 \\ d_2 \\ a_3 \\ d_3 \\ \ldots \end{bmatrix} = \begin{bmatrix} h_0 & h_1 & h_2 & \ldots & h_{M-1} & 0 & 0 & 0 & 0 & \ldots \\ g_0 & g_1 & g_2 & \ldots & g_{M-1} & & & & & \\ 0 & 0 & h_0 & h_1 & h_2 & \ldots & h_{M-1} & 0 & 0 & \ldots \\ & & g_0 & g_1 & g_2 & \ldots & g_{M-1} & & & \\ 0 & 0 & 0 & 0 & h_0 & h_1 & h_2 & \ldots & h_{M-1} & \ldots \\ & & & & g_0 & g_1 & g_2 & \ldots & g_{M-1} & \\ & & & & & & \ldots & & & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ \ldots \end{bmatrix} \qquad (2.72)$$

T is a sparse matrix with block diagonal structure. Each subsequent block has two rows representing $\begin{bmatrix} h \\ g \end{bmatrix}$ and is shifted two columns to the right due to the downsampling. It is clear that the cost of computing (2.72) is O(N). Due to orthogonality, the inverse transform matrix is obtained by transposing T:

$$T^{-1} = T^{T} \tag{2.73}$$

In biorthogonal case, forward transform matrix is similar to (2.72)

$$\tilde{T} = \begin{bmatrix} \begin{matrix}\tilde{h}_0 \\ \tilde{g}_0\end{matrix} & \begin{matrix}\tilde{h}_1 \\ \tilde{g}_1\end{matrix} & \begin{matrix}\tilde{h}_2 \\ \tilde{g}_2\end{matrix} & \begin{matrix}\dots \\ \dots\end{matrix} & \begin{matrix}\tilde{h}_{\tilde{M}-1} \\ \tilde{g}_{\tilde{M}-1}\end{matrix} & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & \begin{matrix}\tilde{h}_0 \\ \tilde{g}_0\end{matrix} & \begin{matrix}\tilde{h}_1 \\ \tilde{g}_1\end{matrix} & \begin{matrix}\tilde{h}_2 \\ \tilde{g}_2\end{matrix} & \begin{matrix}\dots \\ \dots\end{matrix} & \begin{matrix}\tilde{h}_{\tilde{M}-1} \\ \tilde{g}_{\tilde{M}-1}\end{matrix} & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & \begin{matrix}\tilde{h}_0 \\ \tilde{g}_0\end{matrix} & \begin{matrix}\tilde{h}_1 \\ \tilde{g}_1\end{matrix} & \begin{matrix}\tilde{h}_2 \\ \tilde{g}_2\end{matrix} & \begin{matrix}\dots \\ \dots\end{matrix} & \begin{matrix}\tilde{h}_{\tilde{M}-1} \\ \tilde{g}_{\tilde{M}-1}\end{matrix} & \dots \\ & & & & & & \dots & & & \end{bmatrix} \tag{2.74}$$

and the backward transform matrix is given by

$$T = \begin{bmatrix} h_0\ g_0 & 0 & 0 & \\ h_1\ g_1 & 0 & 0 & \\ h_2\ g_2 & h_0\ g_0 & 0 & \\ \dots\ \dots & h_1\ g_1 & 0 & \\ h_{M-1}\ g_{M-1} & h_2\ g_2 & h_0\ g_0 & \\ 0 & \dots\ \dots & h_1\ g_1 & \\ 0 & h_{M-1}\ g_{M-1} & h_2\ g_2 & \dots \\ 0 & 0 & \dots\ \dots & \\ 0 & 0 & h_{M-1}\ g_{M-1} & \\ \dots & \dots & \dots & \end{bmatrix} \tag{2.75}$$

To complete wavelet decomposition by incorporating multilevel expansion, we construct a filter bank that doesn't stop after a couple of filters, but follows the structure of a *wavelet decomposition tree*. In such a tree, decomposition is performed successively by identical blocks shown in Fig. 2.13 each acting upon the a[*k*] sequence of the previous stage (Fig. 2.15).

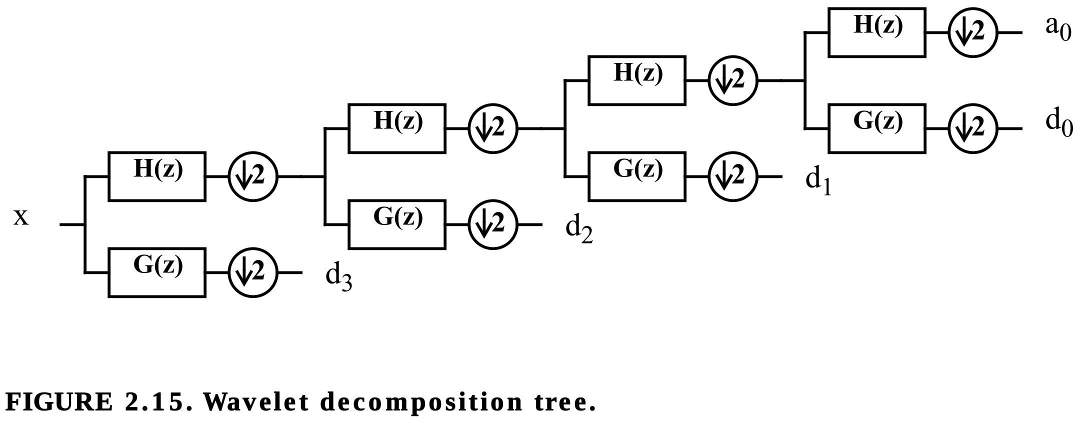


FIGURE 2.15. Wavelet decomposition tree.

Let length of the input sequence be equal to N. The first stage produces two vectors a_3 and d_3 each $N/2$ long. Total computational cost of this stage is $2N\tilde{M}$ where $\tilde{M}$ is length of the decomposition filters (for orthogonal wavelets just substitute M for $\tilde{M}$). Vector a_3 is further processed in the second stage at the cost of $N\tilde{M}$ (half the original because a_3 is half the length of x). Similarly, cost of the third and fourth stages is $N\tilde{M}/2$ and $N\tilde{M}/4$ respectively. Total computational cost of computing the whole decomposition thus becomes

$$C = 2N\tilde{M} + N\tilde{M} + \frac{N\tilde{M}}{2} + \frac{N\tilde{M}}{4} = O(N) \qquad (2.76)$$

This simple calculation has important consequences for the computational cost of wavelet-based algorithms: cost of the fundamental operation of computing a FWT of an arbitrary vector is only O(N), versus O(Nlog$N$) for the FFT. Because of that, computational cost of wavelet algorithms is bounded from below by only O(N).

3. STEADY STATE ANALYSIS OF NONLINEAR CIRCUITS

3.1 PROBLEM BACKGROUND

Steady state analysis of nonlinear circuits represents one of the most computationally challenging problems in EDA. Steady state analysis implies that response of the circuit has to be found at times when all the transients have sufficiently died out [16]. This immediately rules out time marching schemes, especially for high bandwidth circuits, unless a good solution for initial conditions can easily be obtained (shooting methods, [17]). Direct frequency-domain methods are not applicable to the nonlinear circuits either for obvious reasons.

Existing methods for steady state analysis of nonlinear circuits combine both frequency domain and time domain analysis and are generally known as the Harmonic Balance. The essence of this technique is to replace the original Initial Value Problem with a Boundary Value Problem with periodic boundary conditions and to solve the BVP in an appropriate basis that ensures periodicity of the solution.

Harmonic Balance-like methods rely on fast and stable ways of solving nonlinear algebraic equations as well as reasonably fast numerical techniques for going back and forth between time and frequency domain. The speed and accuracy of nonlinear methods primarily depend upon how fast one can factorize the Jacobian matrix for the nonlinear equation.

In this section, we present a newly developed numerical method for steady state analysis of nonlinear circuits. The method is somewhat similar in formulation to the traditional Harmonic Balance technique, but uses wavelets instead of Fourier basis. The new bases allow to reduce density of the Jacobian from a matrix with essentially dense blocks to O(N) bandlimited matrix. The potential of this method leads to significant improvement in computational cost and memory requirements as compared to the traditional Harmonic Balance methods.

3.1.1 Matrix formulation in general form

Consider a lumped component nonlinear circuit that is described by nonlinear Ordinary Differential Equations (ODEs) in time domain. Most often these equations are written in the MNA formulation[1] [18]:

$$C\dot{x} + Gx + f(x) + u = 0 \tag{3.1}$$

Where C and G are $N_x \times N_x$ matrices, x is a column vector of unknown circuit variables and u is a vector of independent sources.

1. State space formulation follows by assuming C to be identity matrix.

For steady state analysis we must either assume that the circuit is under periodic excitation, or that the circuit is autonomous and generates periodic output. In both cases solution vector x is periodic with fundamental frequency corresponding to period τ:

$$x(t+\tau) = x(t) \tag{3.2}$$

Equation (3.1) with boundary conditions (3.2) can be solved by expanding nonlinear ODEs (3.1) into a nonlinear algebraic equation for the expansion coefficients of x according to (2.29) and (2.33). In order for the solution to satisfy boundary conditions (3.2), expansion basis must satisfy these boundary conditions as well. In other words, expansion basis must be periodic. Let us assume that $[x_l]$ is a discrete vector containing values of x sampled in time domain at time points $[t_l]$ and that we have a certain periodic basis $\{v_i\}$ that has a pair of forward T and inverse $\tilde{T}$ discrete transforms associated with it:

$$X = T[x_l],\ [x_l] = \tilde{T}X \tag{3.3}$$

The nonlinear term can be represented in the following form:

$$F(X) = Tf(\tilde{T}X) \tag{3.4}$$

Equation (3.1) then can be written in the transform domain as a nonlinear matrix equation:

$$\hat{C}DX + \hat{G}X + F(X) + U = 0 \tag{3.5}$$

Where $\hat{C}$, D and $\hat{G}$ are $N_tN_x \times N_tN_x$ matrices. $\hat{C}$ and $\hat{G}$ are obtained from C and G respectively by taking their tensor product with a $N_t \times N_t$ identity matrix.

We denote left side of (3.5) as $\Phi(X)$ and write it as

$$\Phi(X) = (\hat{C}D + \hat{G})X + F(X) + U = 0 \tag{3.6}$$

Matrix D in (3.5)-(3.6) is projection of the derivative operator $\frac{d}{dt}$ onto M_n (2.34):

$$[D_{ij}] = \langle \frac{d}{dt}v_i, v_j \rangle \tag{3.7}$$

Solution of (3.6) is usually performed using Newton iterations. Assuming $X^{(0)}$ as the initial guess for X, the linear matrix equation to be solved at each step becomes

$$J(X^{(i)})(X^{(i+1)} - X^{(i)}) = -\Phi(X^{(i)}) \tag{3.8}$$

where $X^{(i)}$ is the solution of i-th iteration, $\Phi(X)$ is defined by (3.6) and $J(X)$ is the Jacobian of $\Phi(X)$:

$$J(X) = [J_{kl}(X)] = \left[\frac{\partial \Phi_k}{\partial X_l}\right], \; k, l = \overline{1, \ldots, (N_tN_x)} \tag{3.9}$$

Substituting (3.6) into (3.9) and applying chain rule, we obtain the following expression for computing the Jacobian [19]:

$$J(X) = \hat{C}D + \hat{G} + T\left[\frac{\partial f_k}{\partial x_l}\right]\tilde{T} \tag{3.10}$$

Jacobian is computed as a sum of three matrix components: $\hat{C}D$, G and $T\left[\frac{\partial f_k}{\partial x_l}\right]\tilde{T}$. Sparsity of the Jacobian becomes equal to the sparsity of the most dense of these 3 components. Matrices $\hat{C}$ and G result from the MNA formulation and typically have rather sparse structure. Matrix of derivatives D will have sparse structure only if chosen basis allows sparse representation of the derivative operator, i.e. most of the elements in (3.7) vanish. This naturally happens if $\{v_i\}$ have local support (local support for basis means local support for it's derivatives and therefore D becomes a bandlimited matrix).

Sparsity of the third component in (3.10) depends primarily on the sparsity of the forward and inverse transform matrices T and $\tilde{T}$ as $\left[\frac{\partial f_k}{\partial x_l}\right]$ for time-invariant systems is just a block matrix consisting of diagonal blocks.

For simplicity, we will first consider a scalar case of (3.1) were both $\hat{C}$ and $\hat{G}$ matrices in (3.6) can safely be assumed as being diagonal.

3.1.2 FOURIER BASIS: HARMONIC BALANCE FORMULATION

For many years now, Fourier basis has been the natural choice for solving the steady-state analysis problem. Fourier basis for solution of (3.1) is usually constructed on an interval that ensures periodicity of the solution, includes $2N_f+1$ basis functions that have the base frequencies that are multiples of the fundamental frequency in the circuit [16]:

$$\{v_i\} = \{1, \cos\omega t, \sin\omega t, \cos 2\omega t, \sin 2\omega t, \ldots, \cos N_f\omega t, \sin N_f\omega t\} \tag{3.11}$$

Because complex exponents are natural eigenfunctions of the derivative operator, derivative matrix D in this basis becomes a diagonal matrix in real Schur form with base frequencies on the main diagonal:

$$D = \omega\begin{bmatrix} 0 & & & & & & & \ldots & & \\ & 0 & 1 & & & & & \ldots & & \\ & -1 & 0 & & & & & \ldots & & \\ & & & 0 & 2 & & & \ldots & & \\ & & & -2 & 0 & & & \ldots & & \\ & & & & & 0 & 3 & \ldots & & \\ & & & & & -3 & 0 & \ldots & & \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ & & & & & & & \ldots & 0 & K \\ & & & & & & & \ldots & -K & 0 \end{bmatrix} \tag{3.12}$$

The transform matrix T has dimensions of $N_t \times (2N_f+1)$ with N_t being the number of time points and N_f being the number of frequencies. This matrix has the following structure:

$$T = \begin{bmatrix} 1 & \cos(\omega t_0) & \sin(\omega t_0) & \ldots & \cos(N_f \omega t_0) & \sin(N_f \omega t_0) \\ 1 & \cos(\omega t_1) & \sin(\omega t_1) & \ldots & \cos(N_f \omega t_1) & \sin(N_f \omega t_1) \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 1 & \cos(\omega t_{N_t - 1}) & \sin(\omega t_{N_t - 1}) & \ldots & \cos(N_f \omega t_{N_t - 1}) & \sin(N_f \omega t_{N_t - 1}) \end{bmatrix} \quad (3.13)$$

If

$$N_t = 2N_f + 1 \quad (3.14)$$

then T is a square matrix which is nonsingular with a proper choice of time sampling points. If more restrictions are imposed on the time sampling points[1], T can also be made orthogonal:

$$\tilde{T} = T^{-1} = T^T \quad (3.15)$$

This matrix clearly is dense which would suggest O(N^2) operations for computing Fourier coefficients in (3.3). This cost can be reduced to O(Nlog$N$) by applying Fast Fourier Transform (FFT) algorithm for computing the T and T^{-1} operators. However, Jacobian in (3.10) invariably becomes a dense matrix which brings cost of solving (3.8) up to O(N^3) at each iteration. In order to reduce the cost of solving (3.10), one must choose a different basis that provides sparse representation for both D and T matrices.

1. Lengthy discussions of different algorithms for the selection of time sampling grid can be found, for example, in [16] and [17] and are really beyond the scope of this thesis.

3.2 WAVELET FORMULATION

Let us consider expansion of a scalar form of (3.1) in wavelet basis in a similar way as traditional Fourier expansion described in Section 3.1.2.

3.2.1 BOUNDARY CONDITIONS

Let expansion basis $\{v_i\}$ be equal to one layer of scaling functions and one layer of wavelets at level J such that $2^J = N_f$:

$$\{v_i\}\Big|_{i=1}^{2N_f} = \{\psi_{i,J}, \varphi_{i,J}\}\Big|_{i=0}^{N_f-1} \tag{3.16}$$

Further, let us assume for simplicity that basis (3.16) is defined on an interval (0, 1) and equation (3.1) is scaled accordingly such that period of the fundamental frequency in the circuit is also equal to 1. To satisfy the boundary condition of $x(0) = x(1)$ we must construct the wavelets in such a way that truncated on an interval basis becomes periodic. Such construction can be easily performed when truncated part of the wavelet (or scaling function for that matter) is not discarded, but appears on the other boundary of the interval (Fig. 3.1).

Periodic basis constructed in such a way naturally enforces periodicity of the solution while retaining all of the properties of wavelets described in Section 2.4.

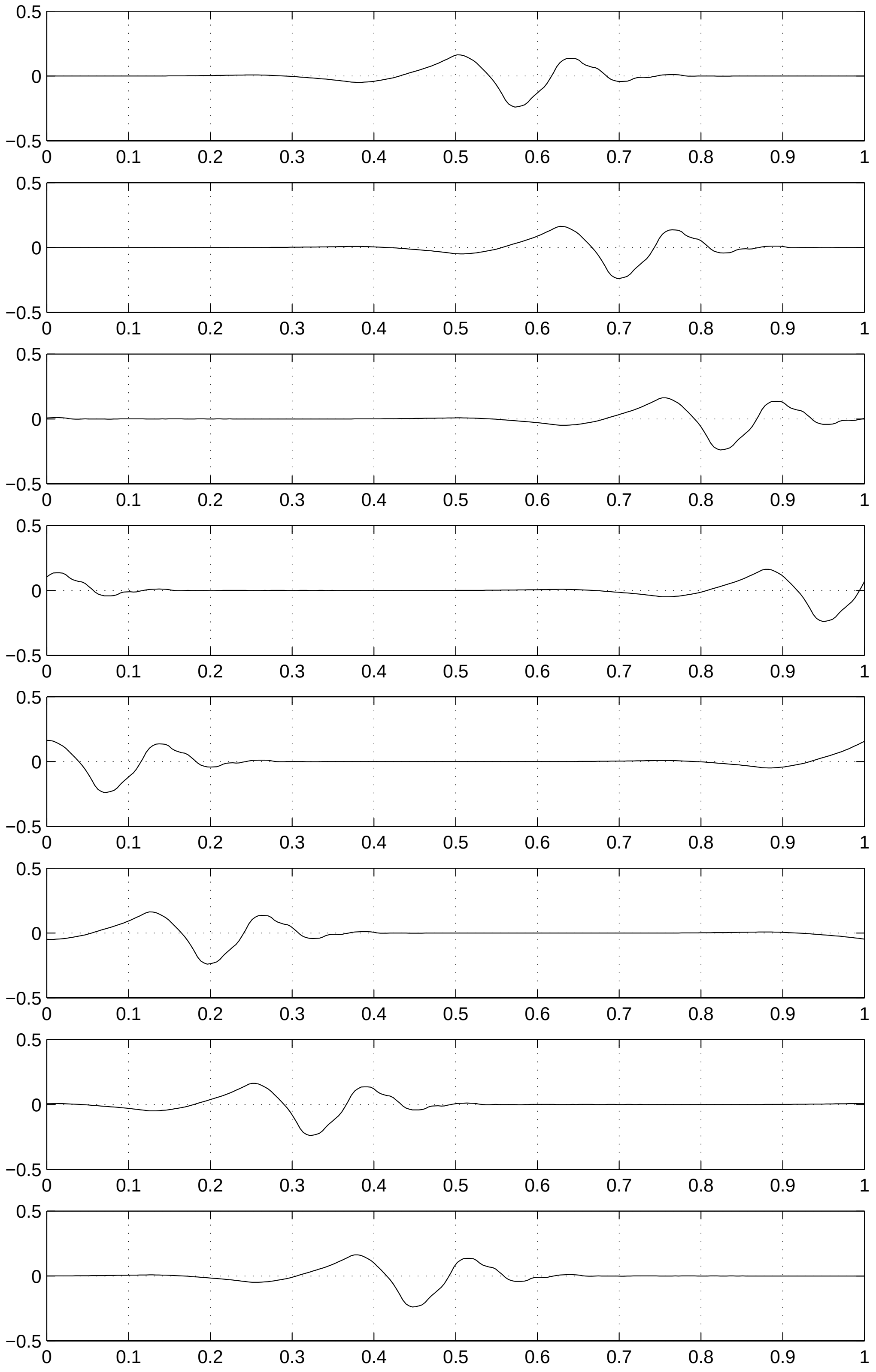


FIGURE 3.1. Periodic wavelet basis on an interval.

3.2.2 TRANSFORM MATRIX

Periodic wavelet basis on an interval (Fig. 3.1) gives rise to a bandlimited transform matrix T that can be obtained from a non-periodic matrix (2.75) by introduction of the “truncated” QMF coefficients into upper right (and, if necessarily, lower left) corners:

$$T = \begin{bmatrix} h_0 \; g_0 & 0 & 0 & \cdots & h_2 \; g_2 \\ h_1 \; g_1 & 0 & 0 & \cdots & \cdots \; \cdots \\ h_2 \; g_2 & h_0 \; g_0 & 0 & \cdots & h_{M-2} \; g_{M-2} \\ \cdots \; \cdots & h_1 \; g_1 & 0 & \cdots & h_{M-1} \; g_{M-1} \\ h_{M-1} \; g_{M-1} & h_2 \; g_2 & h_0 \; g_0 & \cdots & 0 \\ 0 & \cdots \; \cdots & h_1 \; g_1 & \cdots & 0 \\ 0 & h_{M-1} \; g_{M-1} & h_2 \; g_2 & \cdots & 0 \\ 0 & 0 & \cdots \; \cdots & \cdots & 0 \\ 0 & 0 & h_{M-1} \; g_{M-1} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & h_0 \; g_0 \\ 0 & 0 & 0 & \cdots & h_1 \; g_1 \end{bmatrix}$$

(3.17)

Lower left corner coefficients will appear if the QMF taps are aligned around the center of the h vector such that h_0 always appears on the main diagonal [8].

For orthogonal wavelets, inverse transform matrix is obtained from (3.17) via (3.15). For biorthogonal wavelets, $\tilde{T}$ is constructed by augmenting (2.74) in a way similar to described above.

An example of the sparsity pattern for the 64×64 periodized transform matrix constructed with orthogonal Daubechies wavelets with 8 filter coefficients can be observed in Fig. 3.2. Filter coefficients are aligned in the following way: $h = [h_{-4}, h_{-3}, h_{-2}, h_{-1}, h_0, h_1, h_2, h_3]$.

This figure clearly illustrates that for local support wavelets generated by FIR filters, transform matrix remains bandlimited (to a permutation) even in periodic case. With each column containing M nonzero entries, total number of nonzero elements in transform matrix is $N_{NZ} = 2MN_f$ (see also page 42) or O(N). This is already an improvement over traditional Fourier basis which generates a dense transform matrix.

Furthermore, because of the sparsity of the transform matrix and it's band structure, $T\left[\frac{\partial f_k}{\partial x_l}\right]\tilde{T}$ component of (3.10) is also a sparse bandlimited matrix with O(N) nonzero entries (Fig. 3.3).

We will proceed with derivation of the D matrix in wavelet basis to determine the overall sparsity pattern of Jacobian.

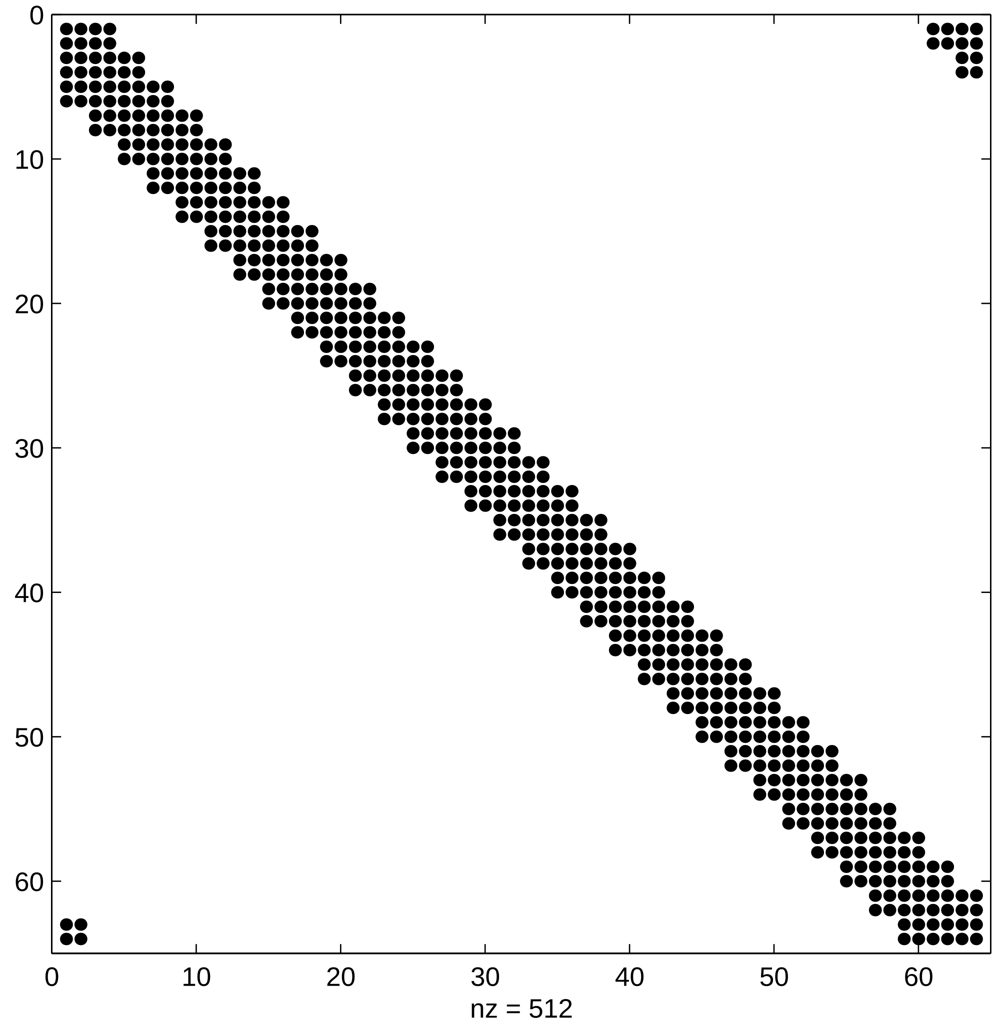


FIGURE 3.2. Sparsity pattern for the periodized transform matrix.

3.2.3 CONNECTION COEFFICIENTS

Derivative matrix D (3.7) contains 4 types of coefficients produced by discretization of derivative operator in wavelet basis (3.16) [20]:

$$\alpha_l = \left\langle \psi\left(\frac{t}{\tau} - \frac{l}{N_t}\right), \frac{\partial}{\partial t}\psi\left(\frac{t}{\tau}\right)\right\rangle \tag{3.18}$$

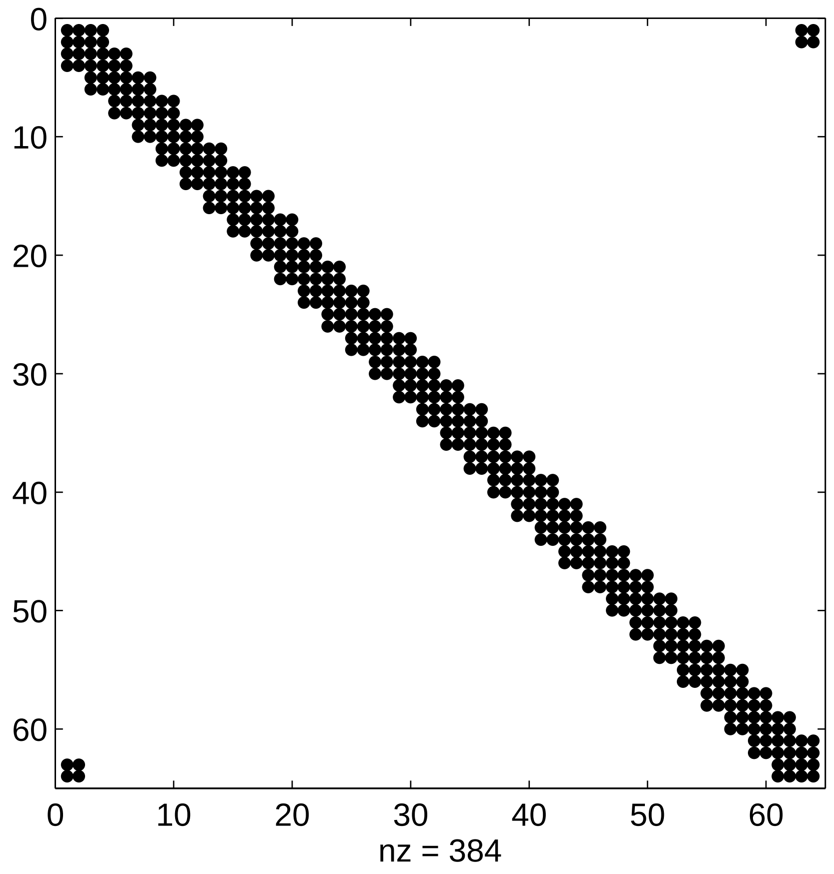


FIGURE 3.3. Sparsity pattern for the $T(\partial f/\partial x)\tilde{T}$ component of Jacobian expanded in a basis of periodic orthogonal Daubechies wavelets of order 2.

$$\beta_l = \left\langle \psi\left(\frac{t}{\tau} - \frac{l}{N_t}\right), \frac{\partial}{\partial t}\varphi\left(\frac{t}{\tau}\right)\right\rangle \tag{3.19}$$

$$\gamma_l = \left\langle \varphi\left(\frac{t}{\tau} - \frac{l}{N_t}\right), \frac{\partial}{\partial t}\psi\left(\frac{t}{\tau}\right)\right\rangle \tag{3.20}$$

$$r_l = \left\langle \varphi\left(\frac{t}{\tau} - \frac{l}{N_t}\right), \frac{\partial}{\partial t}\varphi\left(\frac{t}{\tau}\right)\right\rangle \tag{3.21}$$

where $l = \overline{0, \ldots, N_t - 1}$.

These coefficients obtained by expansion of the derivative operator in a wavelet basis are often called *connection coefficients*.

By substituting (3.18)-(3.21) into the refinement equations (2.41)-(2.44) one can show ([8],[20]) that

$$\alpha_i = 2\sum_{(k)}\sum_{(k')}\tilde{g}_k g_{k'} r_{2i+k-k'} \tag{3.22}$$

$$\beta_i = 2\sum_{(k)}\sum_{(k')}\tilde{g}_k h_{k'} r_{2i+k-k'} \tag{3.23}$$

$$\gamma_i = 2\sum_{(k)}\sum_{(k')}\tilde{h}_k g_{k'} r_{2i+k-k'} \tag{3.24}$$

Therefore, representation of derivative operator in wavelet basis is completely determined by connection coefficients (3.21) obtained from scaling functions only, or in other words, projection the derivative operator on subspace V_J is completely determined by projection on V_0.

For compactly supported (bi)orthogonal wavelets, $\{r_m\}$ is an anti symmetric vector with following properties:

$$r_m \neq 0 \text{ only for } -M+2 \le m \le M-2 \tag{3.25}$$

$$r_0 = 0 \tag{3.26}$$

$$r_{-m} = -r_m \tag{3.27}$$

$$\sum_{(m)} m r_m = -1 \tag{3.28}$$

and, most important:

$$r_m = 2\left[r_{2m} + \frac{1}{2}\sum_{k=1}^{M/2} a_{2k-1}(r_{2m-2k+1} + r_{2m+2k-1})\right] \tag{3.29}$$

where a_i are autocorrelation coefficients of the low pass QMFs:

$$a_i = 2\sum_{m=0}^{M-i-1} \tilde{h}_m h_{m+i},\ i = \overline{1, \ldots, M-1} \tag{3.30}$$

which can be computed with high precision using the following relationships for a wavelet with L vanishing moments [20]:

$$a_{2l-1} = \frac{(-1)^{l-1} C_L}{(L-l)!(L+l-1)!(2l-1)},\ l = \overline{1, \ldots, L} \tag{3.31}$$

where

$$C_L = \left(\frac{(2L-1)!}{(L-1)!4^{L-1}}\right)^2 \tag{3.32}$$

Only odd autocorrelation coefficients have nonzero values. Even coefficients are all equal to zero.

We have to note here that not only a_i are rational numbers by construction, but they only depend on the number of vanishing moments for a particular element and not on the QMF coefficients themselves. Therefore, they can be the same for different wavelets with the same number of vanishing moment.

Linear algebraic system formed by (3.25)-(3.29) is ill conditioned and it's numerical solution is unstable. Fortunately, since the coefficient of this system are rational numbers, it can be solved symbolically. Consequently, basic connection coefficients r_m are also rational numbers by construction and can be computed with any required degree of accuracy.

Connection coefficients for L=1...8 computed according to (3.25)-(3.32) are given in Tables 3.1 and 3.2. It is interesting to observe that for L=1 (Haar wavelets) connection coefficients are equivalent to a well known finite difference discretization scheme. Higher order discretization schemes correspond to wavelets with more vanishing moment.

TABLE 3.1. Connection coefficients r_m for orthogonal wavelets with L=1...5 vanishing moments.

m	L=1	L=2	L=3	L=4	L=5
0	0	0	0	0	0
1	-1/2	-2/3	-272/365	-39296/49553	-957310976/1159104017
2		1/12	53/365	76113/396424	265226398/1159104017
3			-16/1095	-1664/49553	-735232/13780629
4			-1/2920	2645/1189272	17297069/2318208034
5				128/743295	-1386496/5795520085
6				-1/1189272	-563818/10431936153
7					-2048/8113728119
8					-5/18545664272

Matlab code for computing r_m can be found in Appendix A..

TABLE 3.2. Connection coefficients r_m for orthogonal wavelets with L=6...8 vanishing moments.

m	L=6	L=7	L=8
0	0	0	0
1	-3986930636128256/ 4689752620280145	-34141691312970913517142016/ 39300063853302507072666225	-15307411742024122608047464906752/ 17326934462787065887255134938813
2	4850197389074509/ 18759010481120580	3706848761045042521568797/ 13100021284434169024222075	84072878532195990258950029270329/ 277230951404593054196082159021008
3	-1019185340268544/ 14069257860840435	-1181478687077054472781824/ 13100021284434169024222075	-5528889721339826844157784096768/ 51980803388361197661765404816439
4	136429697045009/ 9379505240560290	14265867224679607007907577/ 628801021652840113162659600	8674597455129477747923023172169/ 277230951404593054196082159021008
5	-7449960660992/ 4689752620280145	-152541415888643207462912/ 39300063853302507072666225	-602836894589837888264488353792/ 86634672313935329436275674694065
6	483632604097/ 112554062886723480	1767685682356115678983/ 5240008513773667609688830	857916307369972797746047650035/ 831692854213779162588246477063024
7	78962327552/ 6565653668392203	233086814008971624448/ 55020089394623509901732715	-1328578550813659870682677248/ 17326934462787065887255134938813
8	31567002859/ 75036041924482320	-345875758226176733651/ 209600340550946704387553200	-67976810578396425327419253/ 277230951404593054196082159021008
9	-2719744/ 937950524056029	-25786104557650313216/ 117900191559907521217998675	-6228044283147844191256576/ 155942410165083592985296214449317
10	1743/ 2501201397482744	32878864308626027/ 78600127706605014145332450	99913317392947272157417209/ 1386154757022965270980410795105040
11		-5202857403613184/ 432300702386327577799328475	184824736448048598614016/ 190596279090657724759806484326943
12		-138931281377/ 209600340550946704387553200	603160866987014936359/ 831692854213779162588246477063024
13			-2793278738075222016/ 225250148016231856534316754204569
14			43953990152589/ 277230951404593054196082159021008

3.2.4 DERIVATIVE MATRIX

Having obtained connection coefficients for the expansion of derivative operator in basis of scaling functions (3.21), we can now construct derivative matrix D for (3.7).

We start with constructing matrix R which is projection of the derivative operator onto subspace spanned by scaling functions. Because scaling functions, as well as wavelets, have local support, R is a bandlimited

Toeplitz matrix with it's diagonals filled by r_m [8]. To extend this construction for periodized wavelets, we need to populate the upper left and lower right corners of the matrix as well [21].

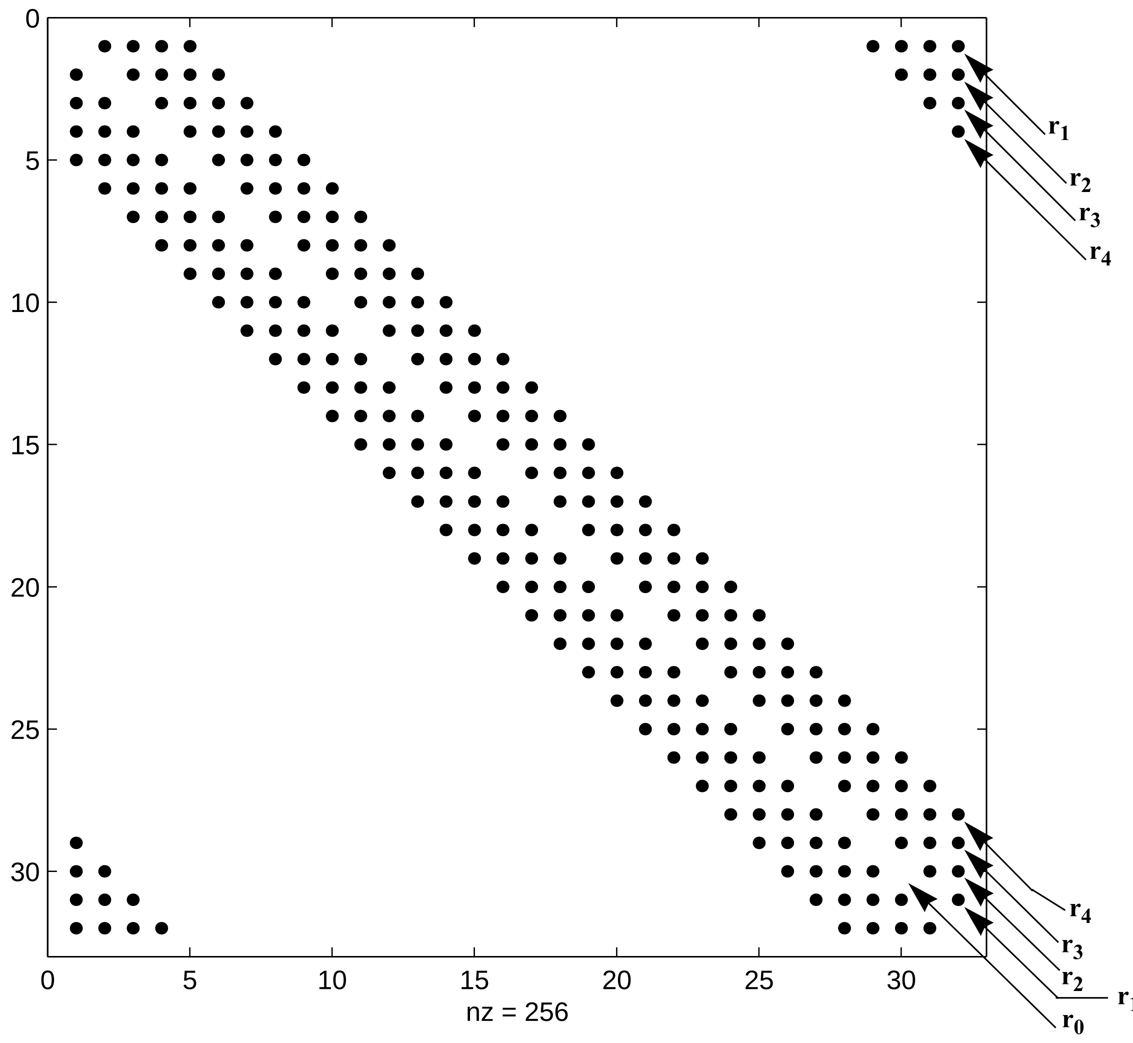


FIGURE 3.4. Derivative matrix *R* for periodized wavelets.

Fig. 3.4 illustrates structure for the derivative matrix *R* constructed with periodized Daubechies wavelets of third order (6 filter coefficients). From here, matrix *D* can be easily obtained using (3.22)-(3.24). These equations describe convolution of the derivative filter with QMFs, which is equivalent to taking a 2-dimensional wavelet transform. In matrix form, we can write it as

$$D = TR\tilde{T} \tag{3.33}$$

where T and $\tilde{T}$ are forward and inverse transform matrices. Sparsity structure of the matrix D will depend on the sparsity structure of transform matrices, which in it's turn depend on the ordering of basis functions. The traditional way (first introduced in [20]) is to place scaling functions first and then wavelets:

$$\{v_i\}\Big|_{i=1}^{2N_f} = \{\varphi_{0,J}, \ldots, \varphi_{N_f-1,J}; \psi_{0,J}, \ldots, \psi_{N_f-1,J}\} \tag{3.34}$$

This will generate matrix D that has four bandlimited quadrants populated by α, β, γ and r as defined in (3.18)-(3.21) (Fig. 3.5). This is convenient for generating representation of the derivative operator in so called non-standard form [8], however such matrix has rather high bandwidth.

We can reduce the bandwidth of matrix D by reordering the basis in such a way, that each scaling function is followed by the overlapping wavelet:

$$\{v_i\}\Big|_{i=1}^{2N_f} = \{\varphi_{0,J}, \psi_{0,J}, \ldots, \varphi_{N_f-1,J}, \psi_{N_f-1,J}\} \tag{3.35}$$

Transform matrices defined by (3.17) in fact correspond to such bases. Because both T and $\tilde{T}$ in this case are bandlimited matrices, as well as

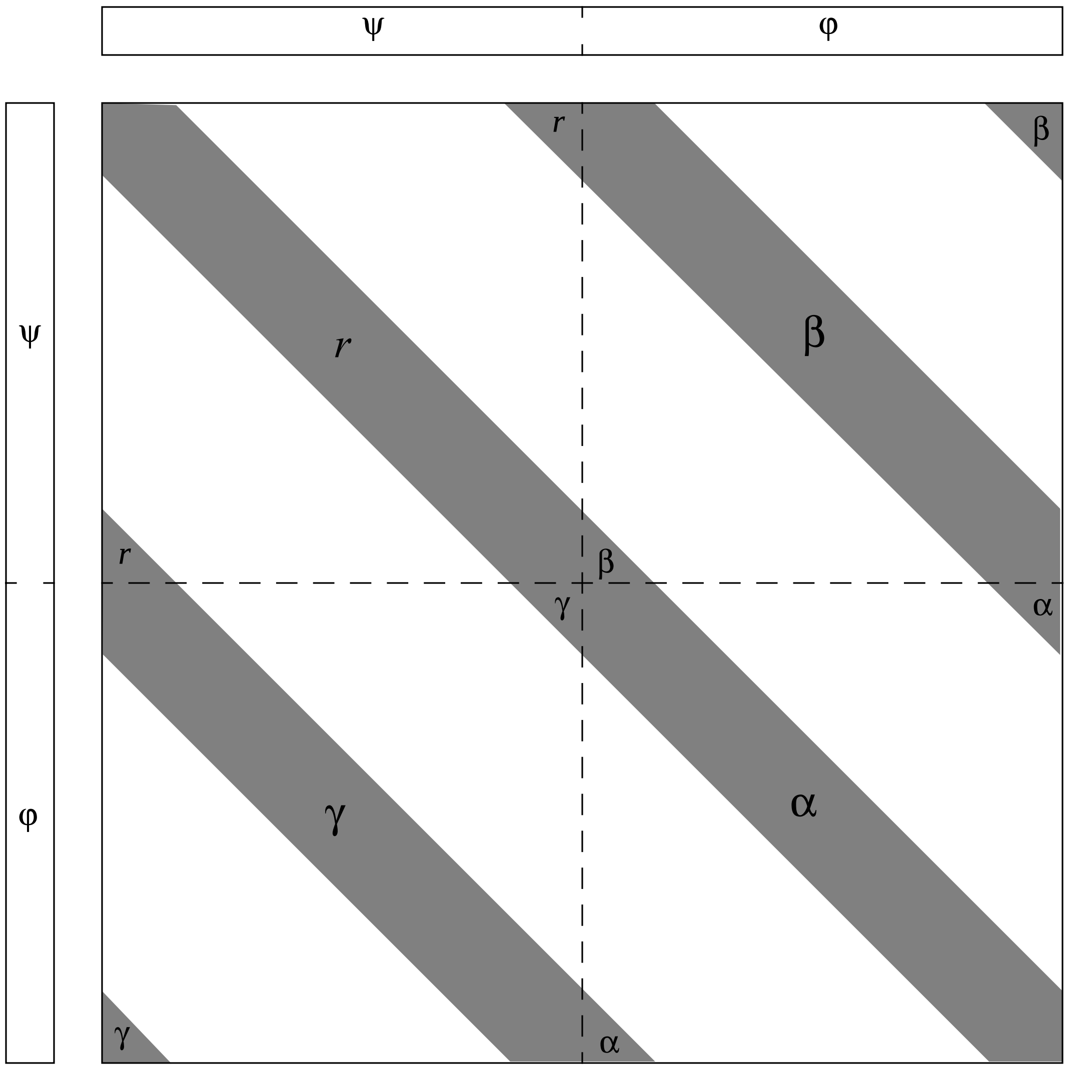


FIGURE 3.5. Structure of matrix *D* obtained in wavelet basis ordered according to (3.34).

R, resulting matrix D is also a bandlimited matrix with O(N) nonzero entries (Fig. 3.6).

Refer back to the Jacobian in equation (3.10). We have established that wavelet expansion leads to construction of sparse bandlimited matrices for all components of the Jacobian. Even though derivative matrix in wavelet basis is not diagonal (as in case of Fourier basis), it has only O(N)

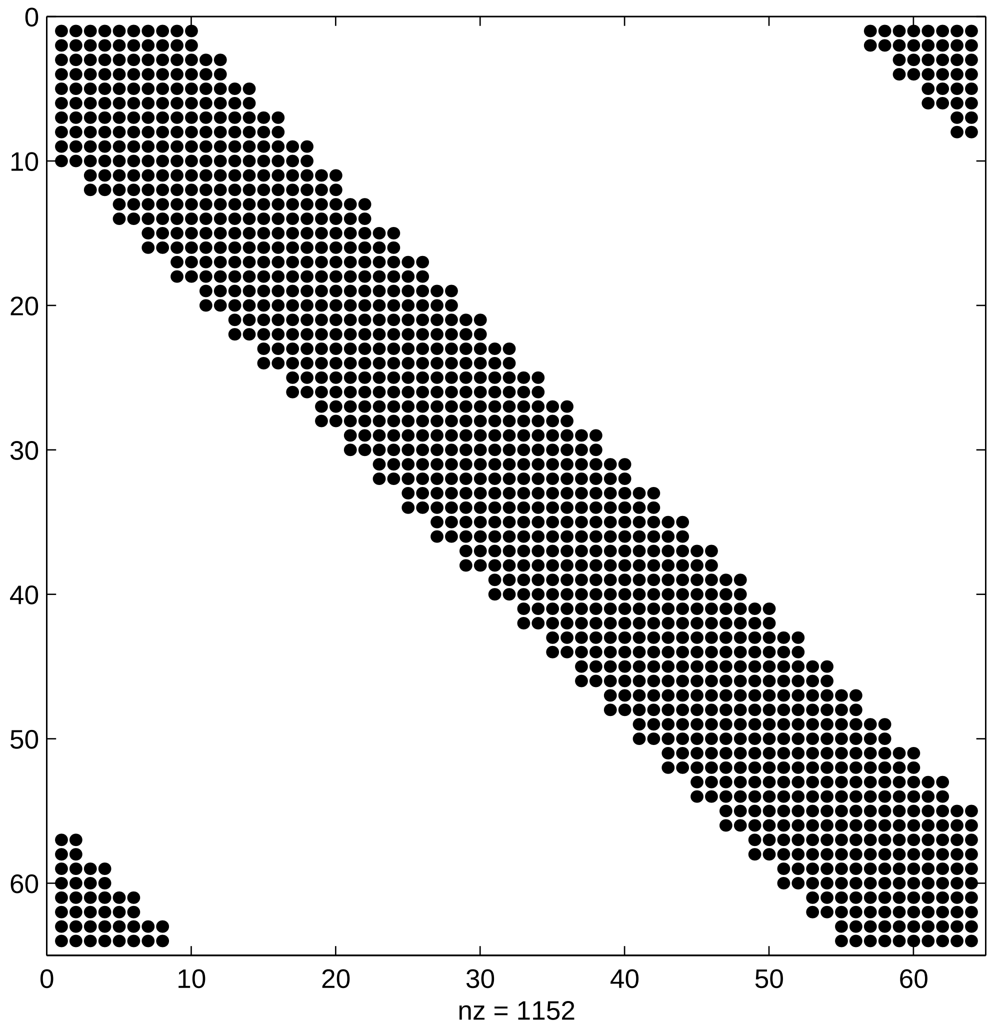


FIGURE 3.6. Sparsity pattern for the 64x64 derivative matrix *D* constructed in basis of order 3 periodized Daubechies wavelets.

nonzero entries. Together with sparse, O(*N*) transform matrix, this results in sparse Jacobian in equation (3.8).

3.3 Analysis of computational complexity

We use two computational cost metrics: number of nonzero elements in the Jacobian matrix and net CPU time required for one LU decomposition and Forward/Backward substitution of the Jacobian. The former is independent of all the platform and implementation issues, is a dominating factor for both memory storage and CPU requirements and provides a good measure of computational resources required to perform the simulation. The latter is highly dependent on the software and hardware implementation of the simulator, but given pretty much state of the art in both, dominates the CPU cost of a Newton's iteration and provides a real world estimate of the CPU time required for the solution. In this section we will derive analytical estimates for the first metric, while reliable data for the second metric can be obtained only experimentally and will be presented in section 3.4.

3.3.1 HARMONIC BALANCE FORMULATION

Let us consider equation (3.1) in scalar form. Provided square Fourier transform is used, T and $\tilde{T}$ matrices in (3.3) are square and dense. Jacobian (3.10) also becomes a dense matrix because of the $T(\partial f/\partial x)\tilde{T}$ component. If we denote order of expansion as N_t, Jacobian is a dense $N_t \times N_t$ matrix that has $O(N_t^2)$ nonzero elements.

Let is generalize this to a vector case. Matrices in (3.6) and (3.10) obtain a block structure with each $N_t \times N_t$ nonzero block corresponding to one nonzero entry in circuit equation matrices (3.1). These nonzero blocks have $O(N_t)$ elements for every nonzero entry in matrices C and G and $O(N_t^2)$ elements for every nonzero entry in $[\partial f_k/\partial x_l]$. Density of Jacobian

(3.10) in this case is dominated by these dense blocks corresponding to nonlinear elements in the circuit. Only the size of these blocks changes with the order of expansion. Overall density of the Jacobian in this case is $N_{NZ} \cong O(\kappa \cdot N_t^2)$, where κ is constant for a given circuit and therefore

$$N_{NZ} \cong O(N_t^2) \tag{3.36}$$

like in scalar case. In it’s turn, order of expansion is linearly proportional to the number of frequencies in truncated set:

$$N_t = 2N_f + 1 \tag{3.37}$$

with 1 accounting for the DC component.

Number of frequencies in truncated set is a critical point for computational cost analysis.

3.3.2 Spectrum truncation issues

Let us denote the highest order of intermodulation products retained in simulation as N_H. Equation (3.11) describes frequency set useful only for analysis of circuits excited by a single tone. For multitone analysis, the set Ω should include harmonics of all the tones as well as all relevant intermodulation products:

$$\Omega = \left\{ \omega | \omega = \left| \sum_{s=1}^{S} k_s \omega_s \right|; \quad k_s \in Z \right\} \tag{3.38}$$

This set is infinite. In order to make the problem computationally solvable, we must truncate this set to one that provides an approximate solution. Truncation schemes are the principal source of errors in steady state analysis, primarily due to the aliasing of truncated components [17].

The simplest truncation scheme (we will refer to it as *trivial truncation*) assumes that all ω_s in (3.38) are commensurate with a single fundamental frequency $\Delta\omega$. Trivial truncation then generates an equidistant frequency grid spanning all the frequencies from 0 to N_H-th harmonic of the highest frequency in ω_s. For example, if tone frequencies are equal to 900 and 910 MHz and $N_H = 10$, the set will span frequencies from 0 to 9100 MHz with step $\Delta\omega = 10$ MHz. If we denote density of the grid as

$$\Delta\hat{\omega} = \frac{\Delta\omega}{\max\{\omega_s\}} \qquad (3.39)$$

then trivial truncation produces a grid that has

$$N_f \cong \mathrm{O}\left(\frac{N_H}{\Delta\hat{\omega}}\right) \qquad (3.40)$$

frequency components. Together with (3.36) and (3.37) this results in the following computational complexity estimation for HB formulation with trivial truncation:

$$N_{NZ} \cong \mathrm{O}\left(\frac{N_H}{\Delta\hat{\omega}}\right)^2 \qquad (3.41)$$

This effectively renders trivial truncation to be unsuitable for all but the simplest and smallest cases.

Another truncation strategy is aimed to retain in Ω_H only the frequencies which carry intermodulation components with orders up to N_H. This strategy gives rise to box

$$\Omega_H = \left\{ \omega | \omega = \left| \sum_{s=1}^{S} k_s \omega_s \right| ; \quad k_s \in Z; \quad |k_s| \le N_H \right\} \tag{3.42}$$

and diamond

$$\Omega_H = \left\{ \omega | \omega = \left| \sum_{s=1}^{S} k_s \omega_s \right| ; \ k_s \in Z ; \ \left| \sum_{s=1}^{S} k_s \right| \le N_H \right\} \tag{3.43}$$

truncation schemes.

In general case, for multitone analysis with S tones and box or diamond truncation, number of frequencies in truncated set is proportional to the volume of a cube in S-dimensional space ([16], p.245):

$$N_f \cong \mathrm{O}(N_H^S) \tag{3.44}$$

However, for a practically interesting case of periodic analysis when all ω_s are commensurate with a single fundamental frequency $\Delta\omega$, N_f grows slower than (3.44) because with increase in N_H frequencies of the new IM products often coincide with already existing in the set. Particularly, if tone frequencies in ω_s are evenly spaced (e.g. 900, 910, 920, ... MHz), set size grows only as

$$N_f \cong \mathrm{O}(S \cdot N_H^2) \tag{3.45}$$

Combining (3.36), (3.37) and (3.45), we conclude that for multitone Harmonic Balance computational cost in terms of the number of nonzero elements in Jacobian is equal to

$$N_{NZ} \cong \mathrm{O}(N_H^4) \tag{3.46}$$

Computational cost in terms of CPU time will actually be slightly higher and also depend on the size and density of the circuit equation (3.1).

3.3.3 Wavelet formulation

Similarly to Harmonic Balance expansion, estimations given in Section 3.2 for wavelet expansion can be generalized to include circuit equations (3.1), where each nonzero element after expansion becomes an $N_t \times N_t$ sparse block, each having $\mathrm{O}(N_t)$ nonzero entries (see Fig. 3.3 and Fig. 3.6). Total number of nonzero entries in wavelet Jacobian becomes

$$N_{NZ} \cong \mathrm{O}(N_t) \tag{3.47}$$

with

$$N_t = 2N_f \tag{3.48}$$

because of the sampling theorem.

With wavelets we use trivial truncation that produces an equidistant uniform frequency grid spanning harmonics and IM components up to required N_H. This, however, is quite a beneficial trade-off as this scheme produces frequency grid with

$$N_f = \mathrm{O}\left(\frac{N_H}{\Delta\hat{\omega}}\right) \tag{3.49}$$

components, where $\Delta\hat{\omega}$ is the relative density if the frequency grid (e.g. for base frequencies 99 and 100 MHz $\Delta\hat{\omega} = 1\%$). Combining (3.47)-(3.49) we conclude that computational cost of wavelet expansion is

$$N_{NZ} = \mathrm{O}(N_H) \tag{3.50}$$

and despite the primitive truncation schemes, with increase in N_H and S wavelet methods very quickly gain significant advantages in computational cost.

Comparison of relative computational complexity is shown in Fig. 3.7. This plot was produced for a scalar case with multitone excitation and closely spaced commensurate tone frequencies (e.g. 1000, 990, 980, 970 MHz, ... for $\Delta\hat{\omega} = 1\%$).

This plot produces some interesting observations. First, computational complexity of Harmonic Balance with diamond truncation does not

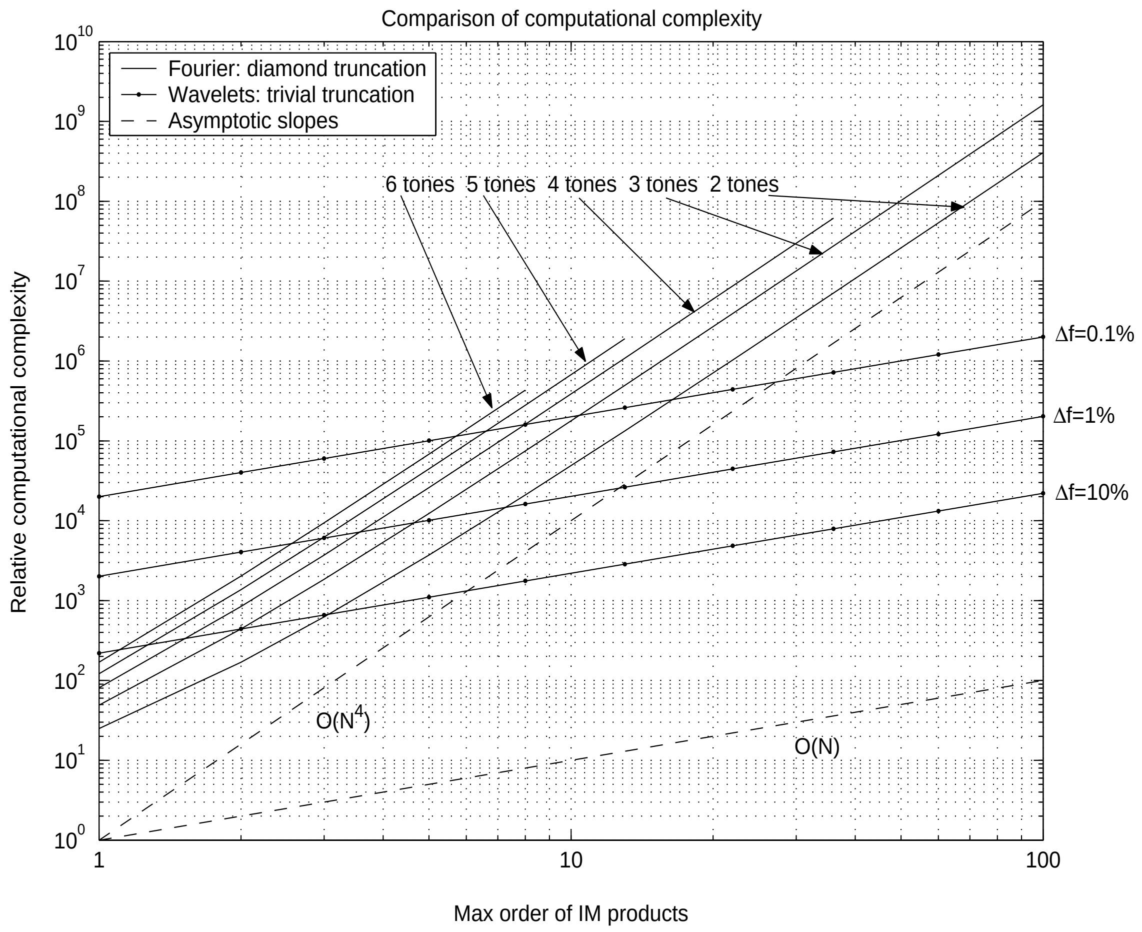


FIGURE 3.7. Comparison of computational complexity in terms of the number of nonzero elements in the Jacobian.

depend on the density of frequency grid. This is quite understandable in view of the fact that Fourier series is a frequency localized basis. Good frequency localization is what makes possible sophisticated truncation schemes in frequency domain. However, this also causes computational cost to be $N_{NZ} \cong O(N_H^4)$ and also to depend on the number of tones. Second, computational cost of wavelet formulation with trivial truncation perfectly follows (3.49) and despite the fact that it depends on the density of frequency grid, it does not depend on the number of tones as long as the newly introduced tones fall into the same grid produced by trivial truncation. This is also quite understandable if we consider the fact that wavelet transform used in the formulation can be represented by a single level filter bank represented in Fig. 2.14. Note, that for (bi)orthogonal wavelets power frequency response of the filter bank satisfies the no distortion condition (2.70), which essentially means that the transform covers all frequency range split into two bands (Fig. 3.8). By comparison, a filter bank associated with Fourier transform has frequency response of a collection of narrowband filters. With trivial truncation, combined frequency response covers the whole frequency range, however, with box or diamond truncation only selected frequencies in the range are covered and introduction of new tones or higher order intermodulation products leads to substantial growth in the number of basis functions (3.45) and, consequently, in computational cost of the solution.

3.4 NUMERICAL RESULTS

All simulations described in this section were performed in Matlab 6.5.0 (R13), running on a SUN Blade-1000 workstation with 900 MHz UltraSPARC-III CPU, 8 MB L2 cache and 5 GB of physical RAM.

Because Matlab environment uses interpreted programming language [28], CPU time was recorded only for the time required to solve the Jacobian matrix (averaged over several Newton's iterations). Recording total simulation time would include all the overhead associated with the inter-

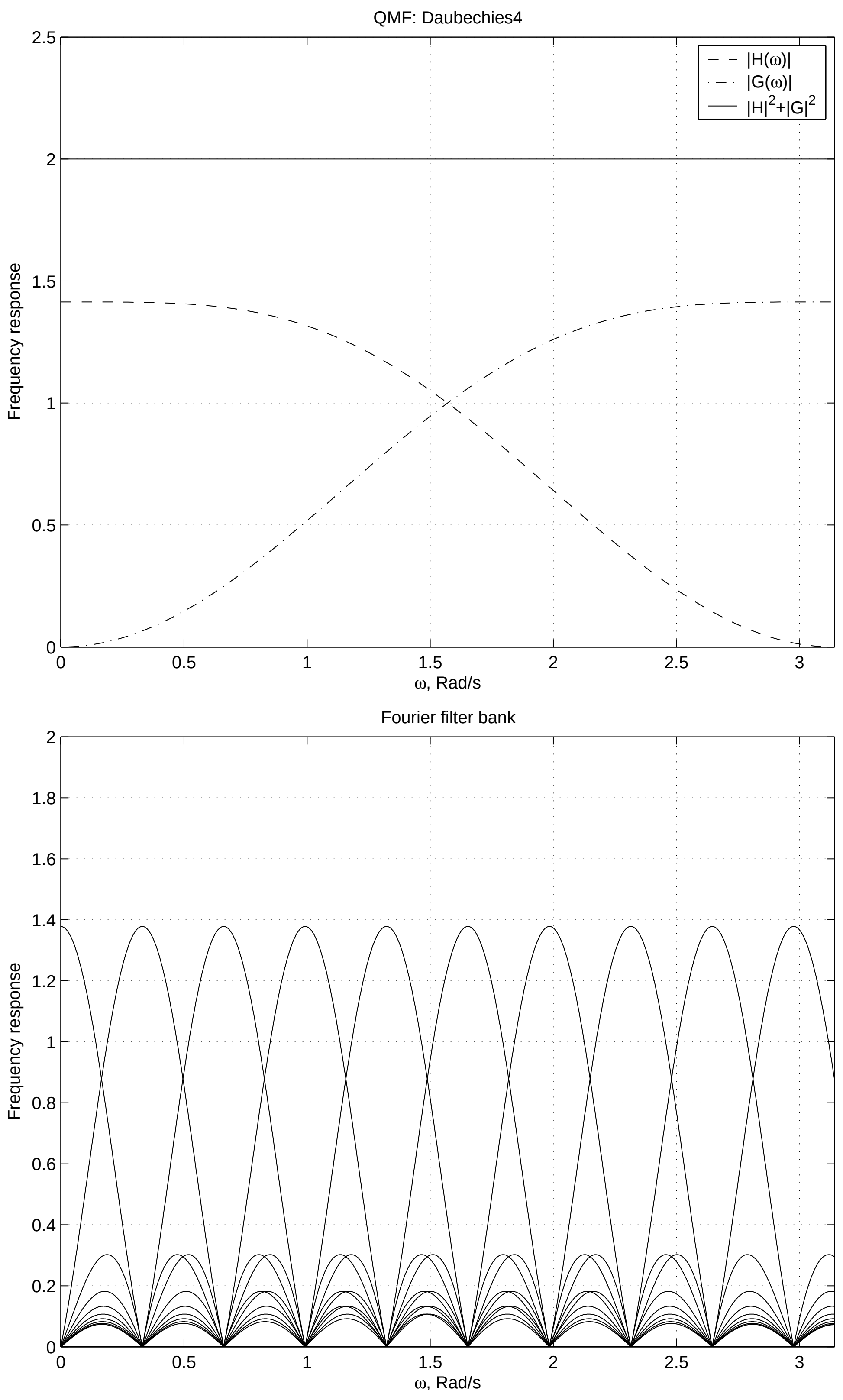


FIGURE 3.8. Frequency response of filter banks corresponding to the wavelet (top) and Fourier (bottom) transforms.

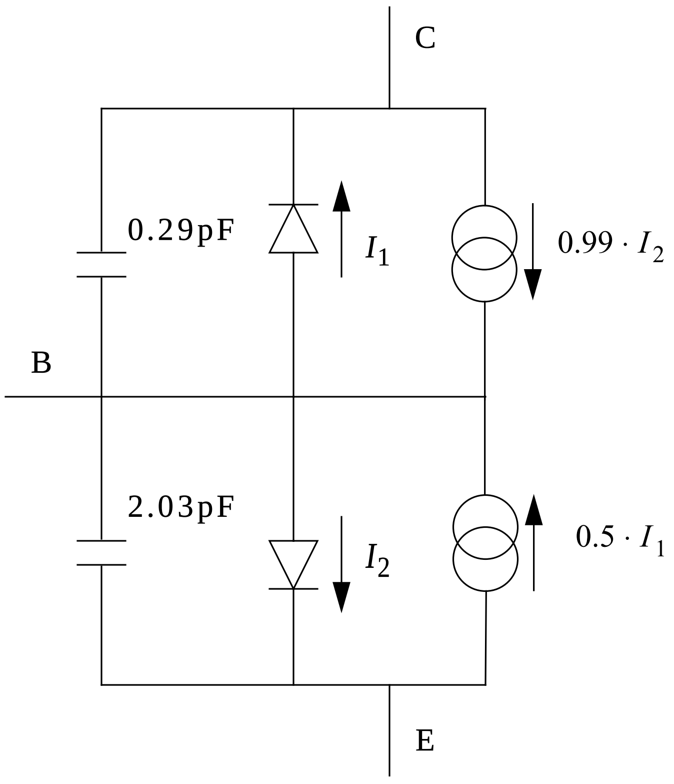
preter and possibly other implementation issues and would produce contaminated and therefore misleading results. Matlab 6.x sparse matrix solver relies on the UMFPACK package ([29], [30]). Matlab's left matrix division operator was used to invoke the matrix solver, which in this case performs LU decomposition using Gaussian elimination with partial pivoting algorithm. By default, the solver performs column approximate minimum degree preordering before performing Gaussian elimination. It was established that explicit utilization of other preordering algorithms is extremely beneficial for the steady state analysis problems. Symmetric approximate minimum degree (SAMD) preordering was used for Jacobians arising from the Fourier series expansion, while for the wavelet expansion it appeared to be possible to use symmetric reverse Cuthill-McKee (SRCM) preordering ([29]).

In both examples diamond truncation was used for Harmonic Balance and trivial truncation for wavelet expansion.

All the results of wavelet expansion presented in this section were obtained with Daubechies wavelets of second order. Some experiments were also performed with Haar wavelets and higher orders of Daubechies wavelets. Haar expansion produced poor results in both accuracy and convergency, while higher order Daubechies wavelets produced essentially the same accuracy and convergence as the second order, but at a slightly higher computational cost.

3.4.1 CASE STUDY: CASCODE LNA

A 900 MHz cascode LNA was considered in the first example. The amplifier (Fig. 3.9) consists of 2 BJTs with DC bias and impedance matching networks. A simple Ebers-Moll injection model (Fig. 3.10) was used for both BJTs. Even though this model probably is not accurate at the higher range of simulated frequencies, the goal of this experiment is to validate wavelet formulation, so as long as the same model is used for both Harmonic Balance and wavelet method, analysis results should match.



$$I_1 = I_s\left(e^{\frac{V_{BC}}{V_T}} - 1\right)$$

$$I_2 = I_s\left(e^{\frac{V_{BE}}{V_T}} - 1\right)$$

$$I_s = 10^{-15}\text{A}$$

$$V_T = 26\times10^{-3}\ \text{V}$$

FIGURE 3.10. Ebers-Moll injection model for BJTs.

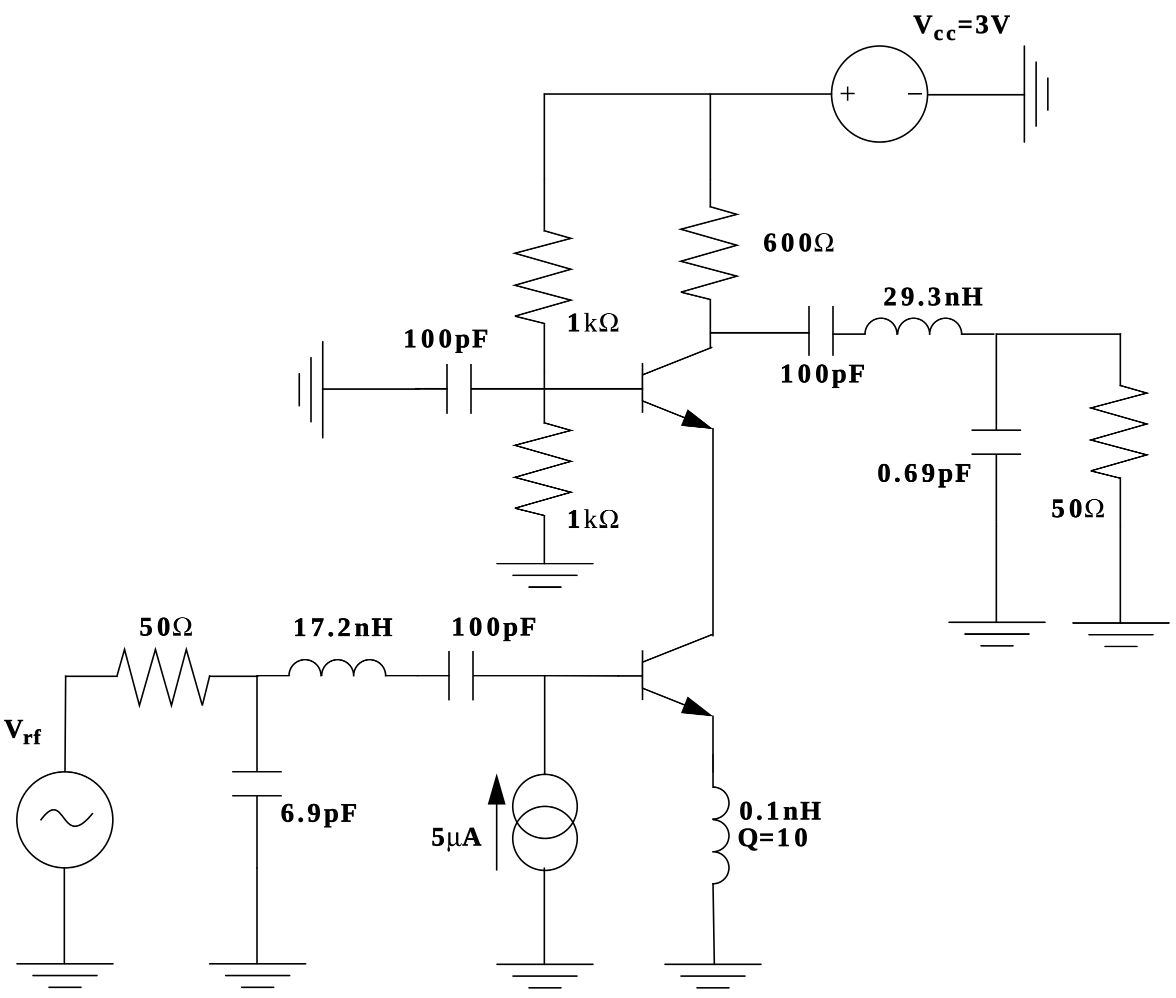


FIGURE 3.9. Cascode LNA circuit.

Under all these assumptions, total size of MNA equations in this example was 25.

The first experiment is a 900 MHz single tone simulation, which is routinely performed during design stage of such amplifiers to determine gain and 3dB compression point. Up to the 16-th order intermodulation products had to be retained for this simulation to ensure convergence and accuracy. Output power delivered to the load at first (900 MHz), second

(1800 MHz) and third (2700 MHz) harmonics versus the input power is plotted in Fig. 3.11. Both methods are in excellent agreement with each other and with simulations performed independently in [27]. Both methods also exhibited essentially the same convergence and computational cost, which is understandable given the small size of the problem (Jacobian size was 775×775 with 16,470 nonzero entries for Fourier series and 800×800 with 10,827 nonzero entries for wavelets).

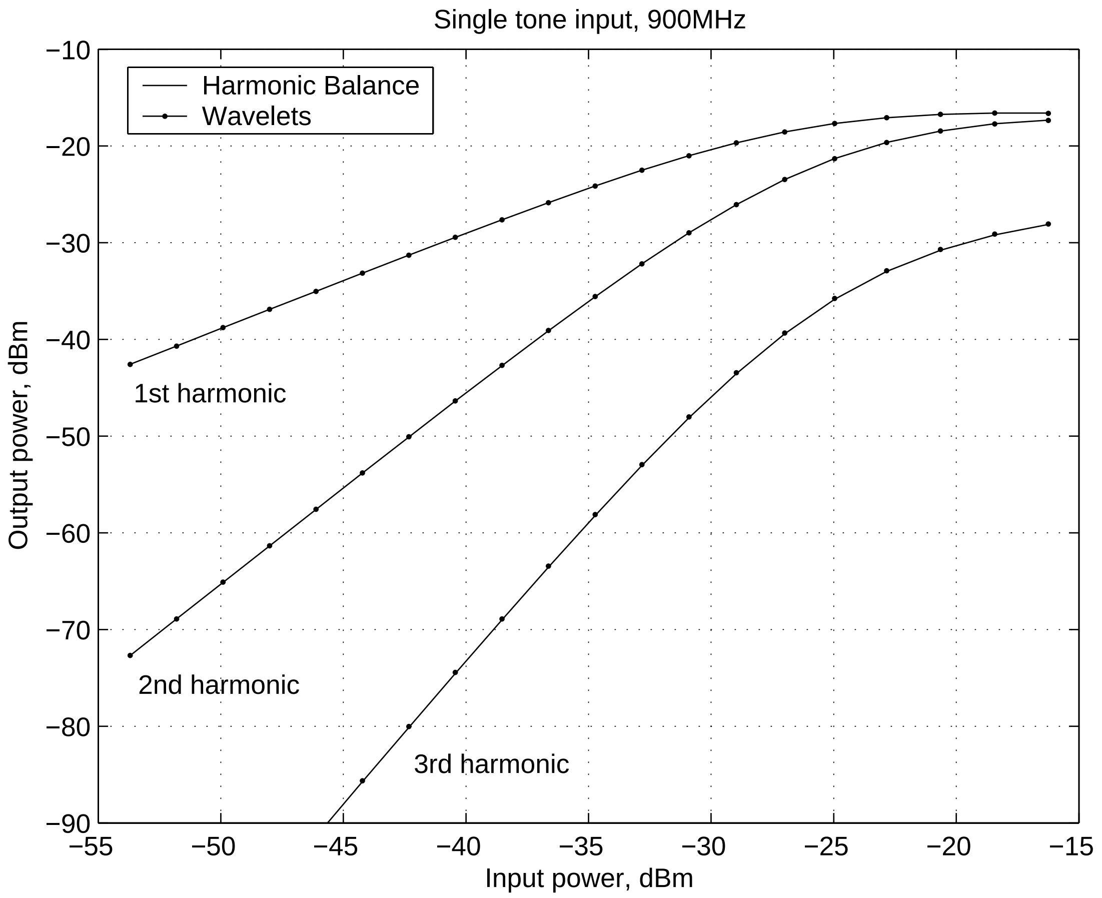


FIGURE 3.11. Single tone input simulation results for the cascode LNA in Fig. 3.9.

Second experiment involves the same circuit under multitone excitation, with two tone input signals of the same power and frequencies of 900 and 910 MHz. Purpose of this experiment is to validate speed and accuracy of the wavelet expansion on computations of the third order in-band intermodulation products at 920 and 930 MHz. Simulation results are shown in Fig. 3.12 and are in excellent correspondence with each other. In each

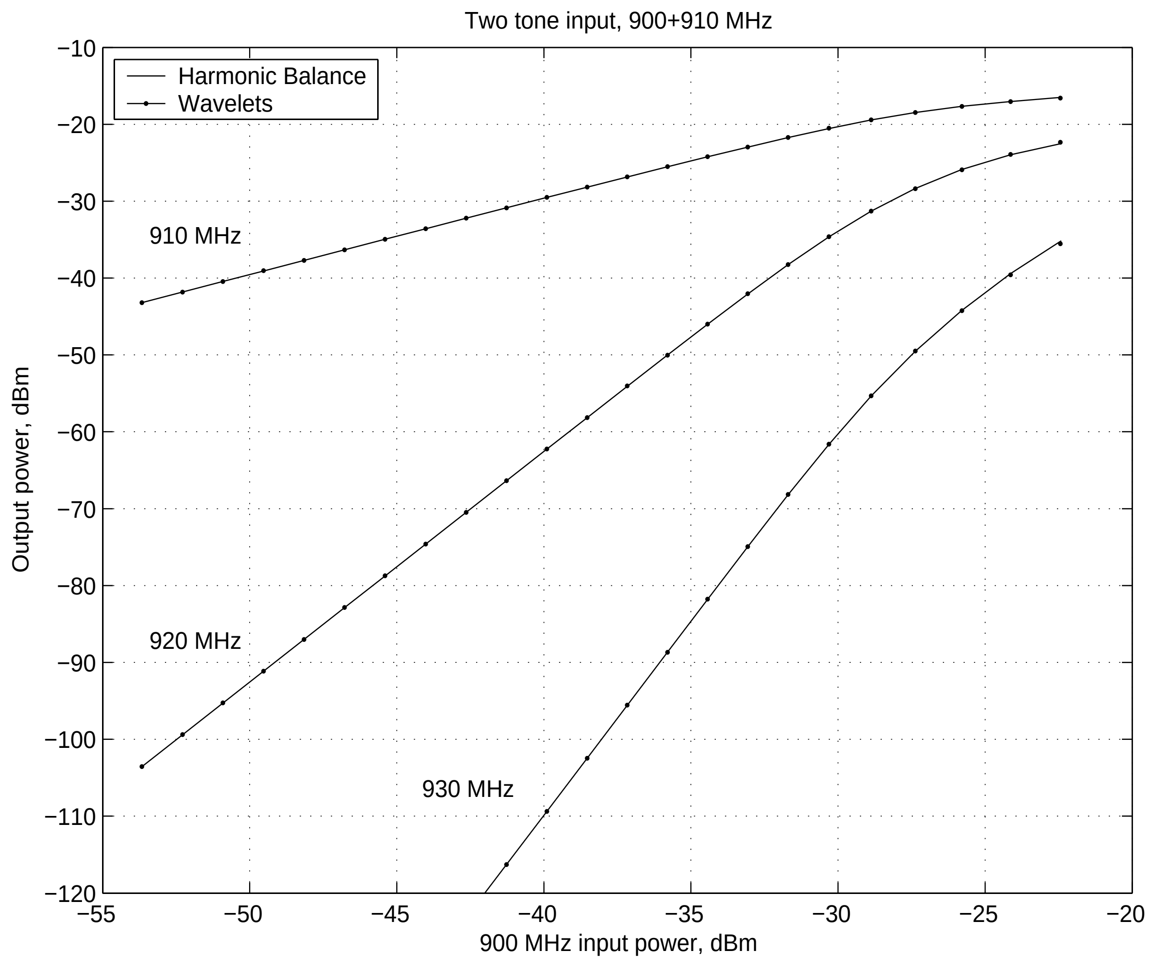


FIGURE 3.12. Two tone input simulation results for the cascode LNA in Fig. 3.9.

case (HB and wavelets) intermodulation products were computed with N_H ranging from 5 to 22 (maximum value for HB given software implementation and available memory). To compare computational complexity of both methods, number of nonzero elements in the Jacobian and average time for one LU decomposition was recorded and is shown as a function of N_H in Fig. 3.13 and Fig. 3.14 respectively. Detailed data for N_H = 6, 12, 18 and 22 can be also found in tables 3.3-3.6.

TABLE 3.3. Computational cost comparison for cascode LNA (Fig. 3.9) at N_H = 6.

	Harmonic Balance	Wavelets
Frequency grid size	43	556
Time grid size	85	1,112
Jacobian size	2125×2125	27800×27800
Number of nonzero elements	114,219	386,976
Sparsity ratio	2.5%	0.05%
Memory storage size, MBytes	1.35	4.6
Average CPU time for preordering, seconds	0.17	0.26
Average time per LU/FBS, seconds	1.8	7.7
Number of Newton iterations	5	5

TABLE 3.4. Computational cost comparison for cascode LNA (Fig. 3.9) at N_H = 12.

	Harmonic Balance	Wavelets
Frequency grid size	157	1,102
Time grid size	313	2,204
Jacobian size	7825×7825	55100×55100
Number of nonzero elements	1,489,284	747,174
Sparsity ratio	2.4%	0.025%
Memory storage size, MBytes	17.5	9.4
Average CPU time for preordering, seconds	2.5	0.5
Average time per LU/FBS, seconds	70	15
Number of Newton iterations	5	5

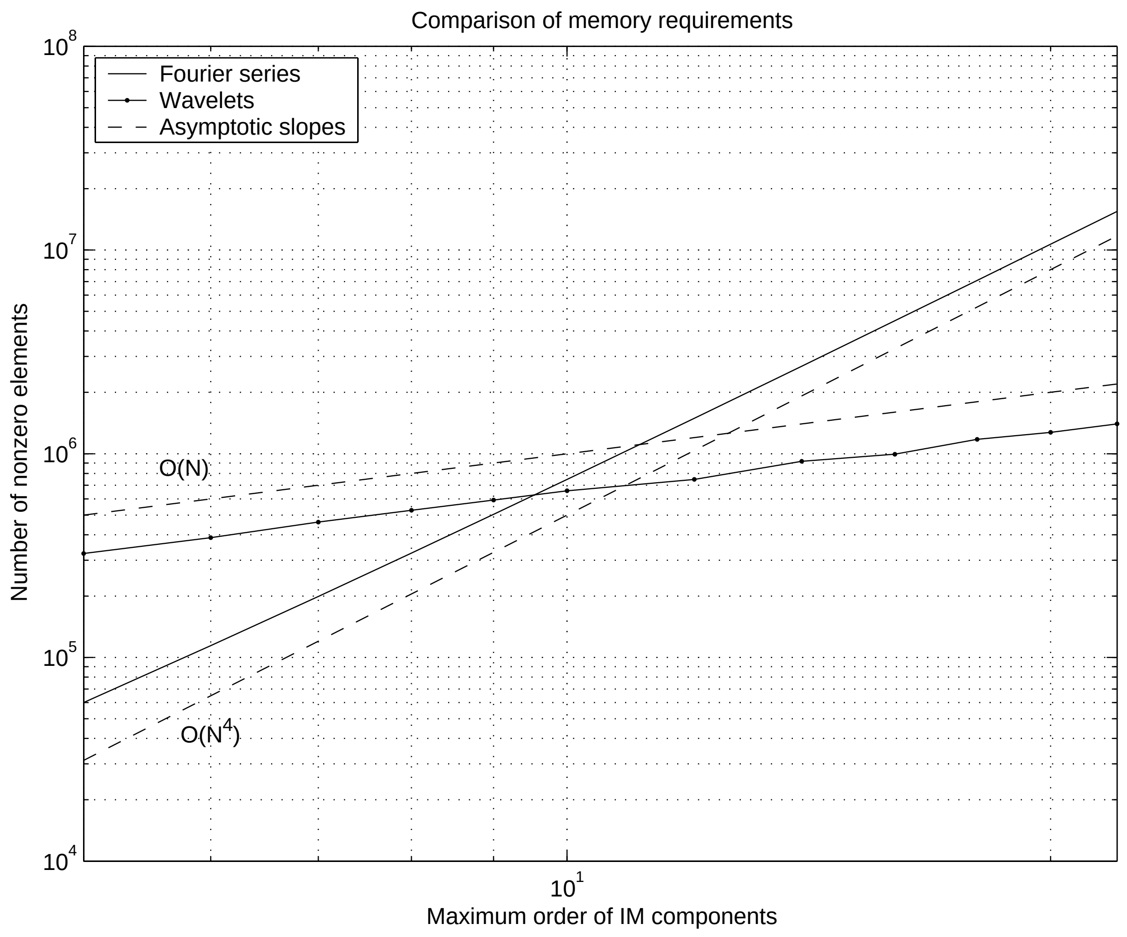


FIGURE 3.13. Number of nonzero elements in Jacobian for the cascode LNA in Fig. 3.9.

TABLE 3.5. Computational cost comparison for cascode LNA (Fig. 3.9) at N_H = 18.

	Harmonic Balance	Wavelets
Frequency grid size	343	1,648
Time grid size	685	3,296
Jacobian size	17125×17125	82400×82400
Number of nonzero elements	7,085,619	1,176,654
Sparsity ratio	2.4%	0.017%
Memory storage size, MBytes	83.1	14.8
Average CPU time for preordering, seconds	77	0.8
Average time per LU/FBS, seconds	840	22
Number of Newton iterations	5	5

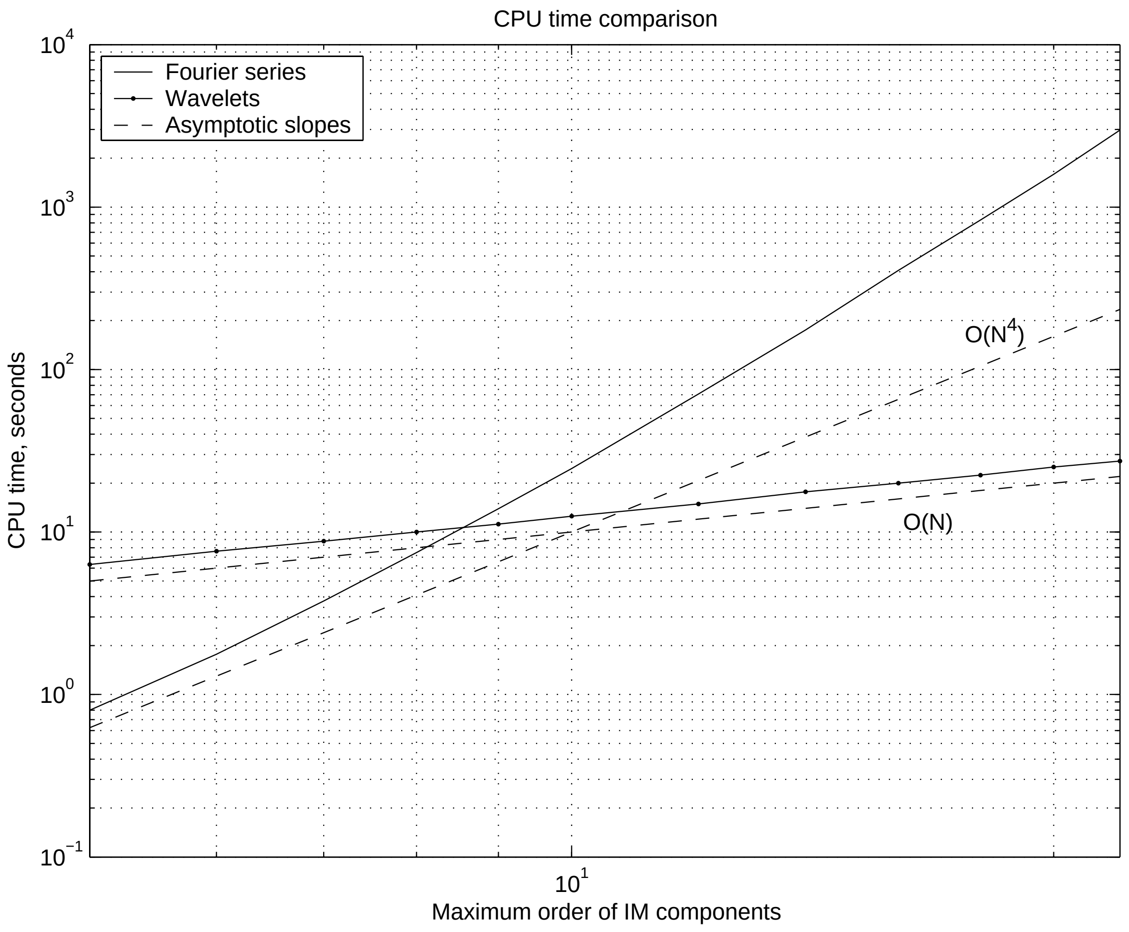


FIGURE 3.14. Average CPU time per LU decomposition of Jacobian for the cascode LNA in Fig. 3.9.

TABLE 3.6. Computational cost comparison for cascode LNA (Fig. 3.9) at N_H = 22.

	Harmonic Balance	**Wavelets**
Frequency grid size	507	2,012
Time grid size	1,013	4,024
Jacobian size	25325×25325	100600×100600
Number of nonzero elements	15,462,403	1,400,370
Sparsity ratio	2.4%	0.014%
Memory storage size, MBytes	181.3	17.6
Average CPU time for preordering, seconds	278	1.0
Average time per LU/FBS, seconds	2,980	28
Number of Newton iterations	5	5

As can be seen from Figures 3.13 and 3.14, experimental data for the comparison of computational cost follows the trends predicted by analysis performed in Section 3.3 (Fig. 3.7). Even though trivial truncation results in a large matrix size for wavelet expansion, these matrices are extremely sparse and wavelet method becomes computationally more favourable for $N_H \geq 10$. The trend of $O(N_H^4)$ derived in (3.46) is just too powerful and quickly overcomes linear cost of wavelet expansion. CPU time rises even a bit faster than that, which is understandable in view of the computational complexity of the Gaussian elimination for large scale sparse matrices arising from MNA equations ([18]) being $O((N_t \times N_x)^{1+\alpha})$ where $0 < \alpha \ll 1$ is a parameter which depends on sparsity ratio of the matrix $N_{NZ}/(N_t N_x)^2$.

Both Fourier series and wavelet expansions produce Jacobian with sparsity pattern similar to the original MNA equation except for the fact that with the Fourier series blocks corresponding to nonlinear elements are dense matrices. Because of this, sparsity ratio of HB Jacobian stays essentially the same with increase in N_H (2.4% in this example), while sparsity ratio of the wavelet Jacobian decreases as $O(1/N_H)$ thus compensating for the increased matrix size. In fact, this compensation allows the CPU time metric to stay $O(N_H)$ even when using a general purpose matrix solver.

An example of the sparsity pattern for Jacobian arising from Fourier series expansion is shown in Fig. 3.15. Note the dense blocks corresponding to the nonlinear elements are dominating the nonzero element count. These blocks account for 98.5% of nonzero elements, thus also dominating computational cost in terms of CPU time.

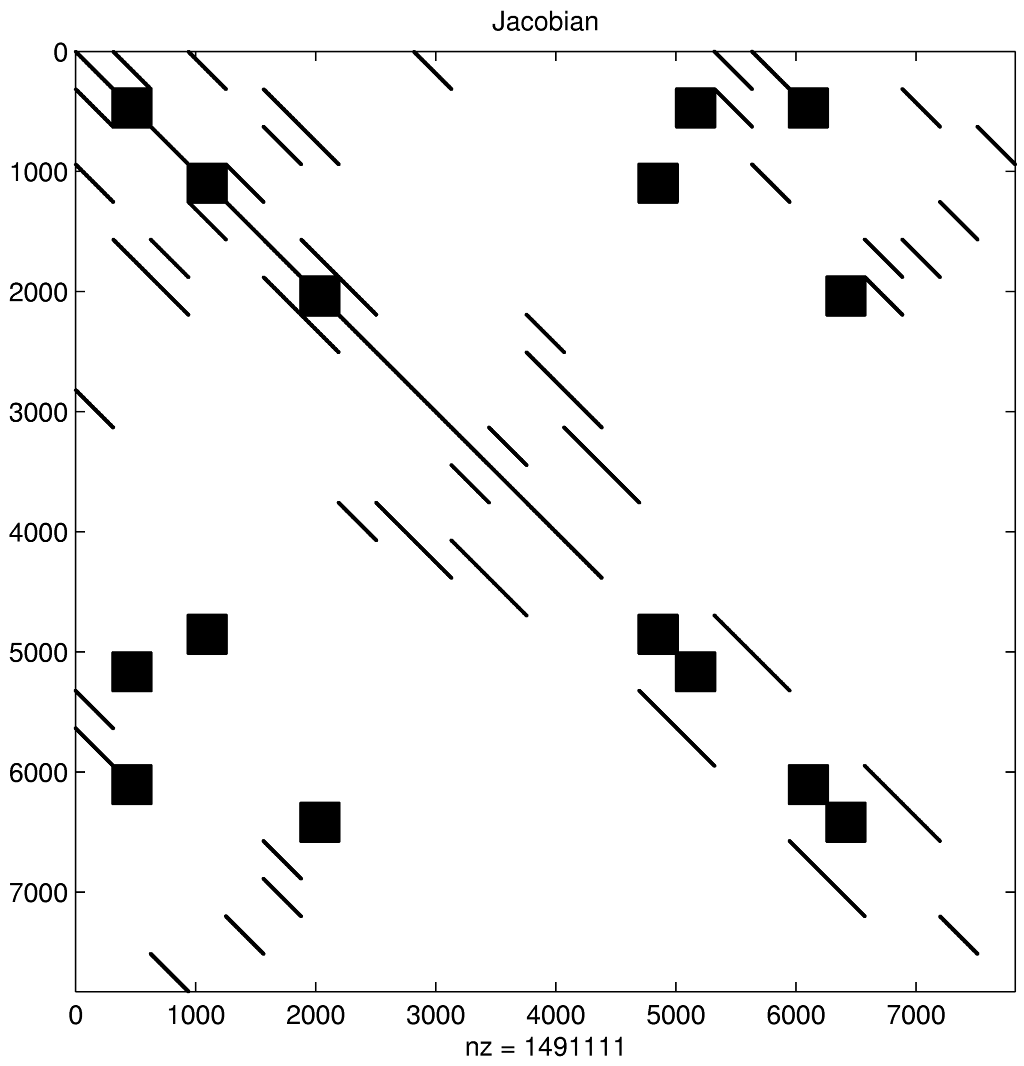


FIGURE 3.15. Sparsity pattern for the Jacobian arising from Fourier series expansion for cascode LNA in Fig. 3.9 with N_H = 12. Dense blocks account for 98.5% of nonzero elements.

Fig. 3.16 show sparsity pattern of this Jacobian after SAMD reordering. By comparison, Fig. 3.17 shows the sparsity pattern obtained after SRCM reordering. RCM algorithm tries to find a reordering that produces a bandlimited structure. As could be expected, SRCM reordering in this case performs quite poorly because of the dense blocks in the sparsity pattern of the original Jacobian. The resulting matrix structure in Fig. 3.17 has fairly high bandwidth, while it is known ([29]) that the band experiences significant fill-in during LU decomposition (Fig. 3.18), leading to high memory and CPU time requirements. On the other hand, minimum degree algorithms try to find such reordering that it would produce a structure with large blocks of contiguous zeros which do not fill in during factorization (Fig. 3.19).

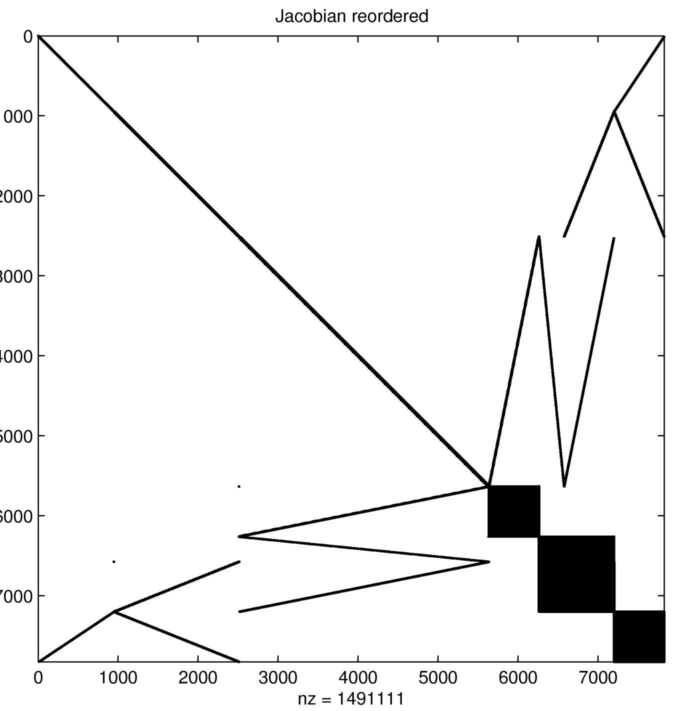


FIGURE 3.16. Jacobian in Fig. 3.15 after symmetric approximate minimum degree reordering.

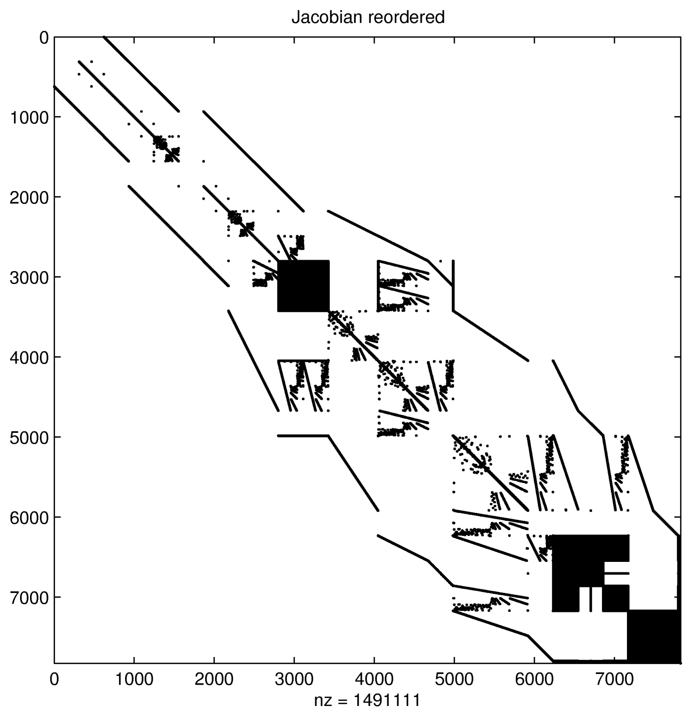


FIGURE 3.17. Jacobian in Fig. 3.15 after symmetric reverse Cuthill-McKee reordering.

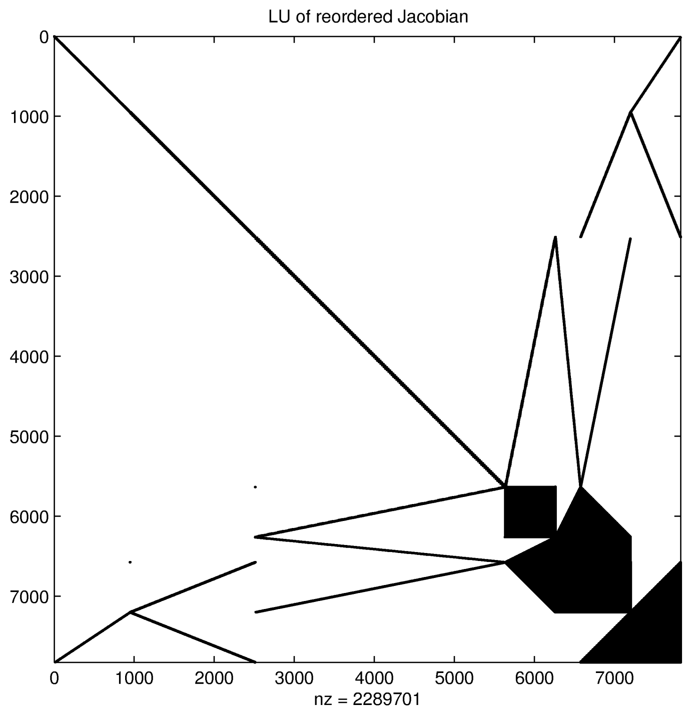


FIGURE 3.18. Sparsity pattern of the LU factors for SAMD-reordered HB Jacobian (Fig. 3.16).

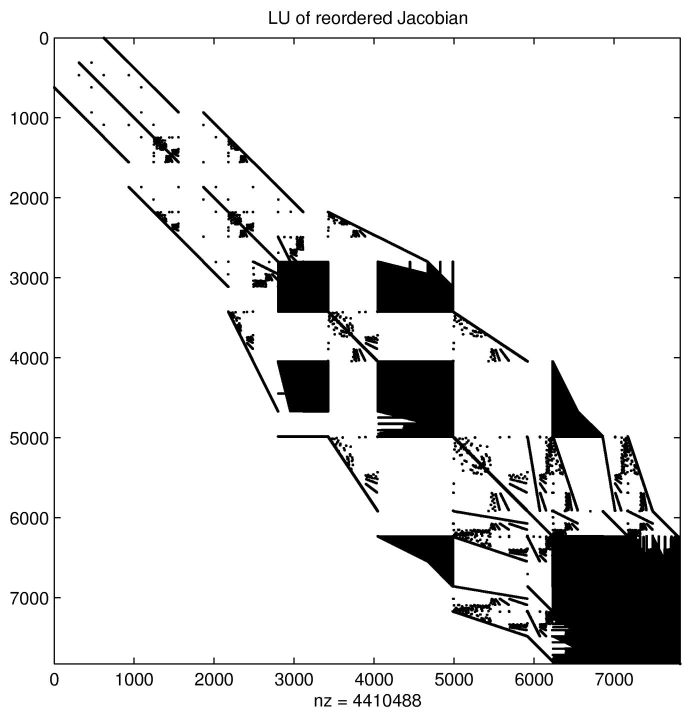


FIGURE 3.19. Sparsity pattern of the LU factors for SRCM-reordered HB Jacobian (Fig. 3.17). Note significant fill-ins within the band that result in almost 2-fold increase in the nonzero elements count (as compared to SAMD in Fig. 3.18).

Compare this to the sparsity patterns of the Jacobian obtained from wavelet expansion (Figures 3.20-3.23). Jacobian (Fig. 3.20) has sparse bandlimited blocks in place of dense blocks in the HB Jacobian (Fig. 3.15). Size of the wavelet Jacobian is much larger, because both were obtained with $N_H = 12$ and wavelet expansion with trivial truncation utilizes larger frequency grid. However, SRCM reordering works extremely well for such matrices and produces an extremely narrowband matrix (Fig. 3.22). Theoretical computational cost for solving narrowband matrices ([32], pp. 149-153) is O(N), or more precisely O(Npq) floating point operations, where p and q are the bandwidths of the upper and lower triangles respectively. For $N = 55100$ and $p = q \approx 200$ this results in computational complexity of approximately 2.2 Gflops. With the computer rated at approximately 700 Mflops/s on dense matrix computations (see Appendix A.3) and a customized bandlimited matrix solver, theoretically we could achieve CPU time of several seconds. Recall (Table 3.4) that solution of this matrix required only 15 seconds of CPU time with a *general purpose matrix solver* and all the *overhead associated with sparse matrix storage*. This illustrates the point that a well-engineered general purpose matrix solver can provide performance that rivals that of the solvers custom tailored for a specific problem.

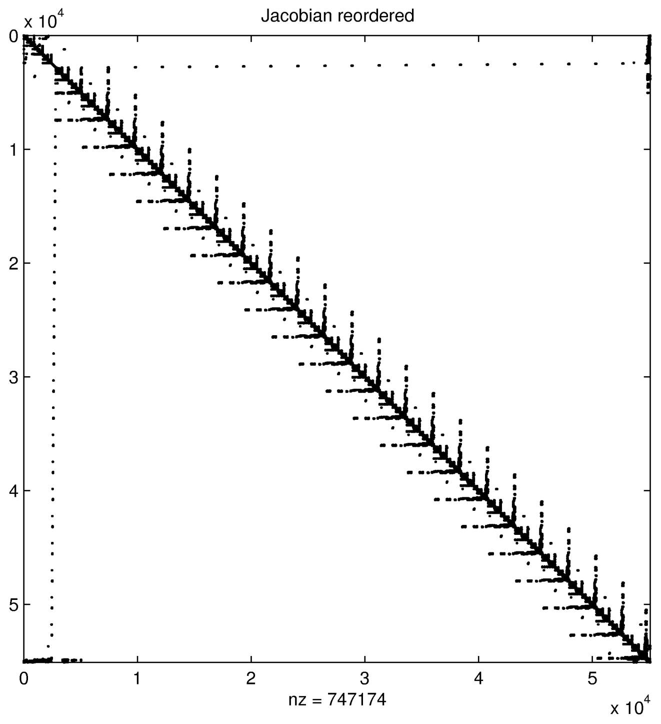


FIGURE 3.21. Jacobian in Fig. 3.20 after symmetric approximate minimum degree reordering.

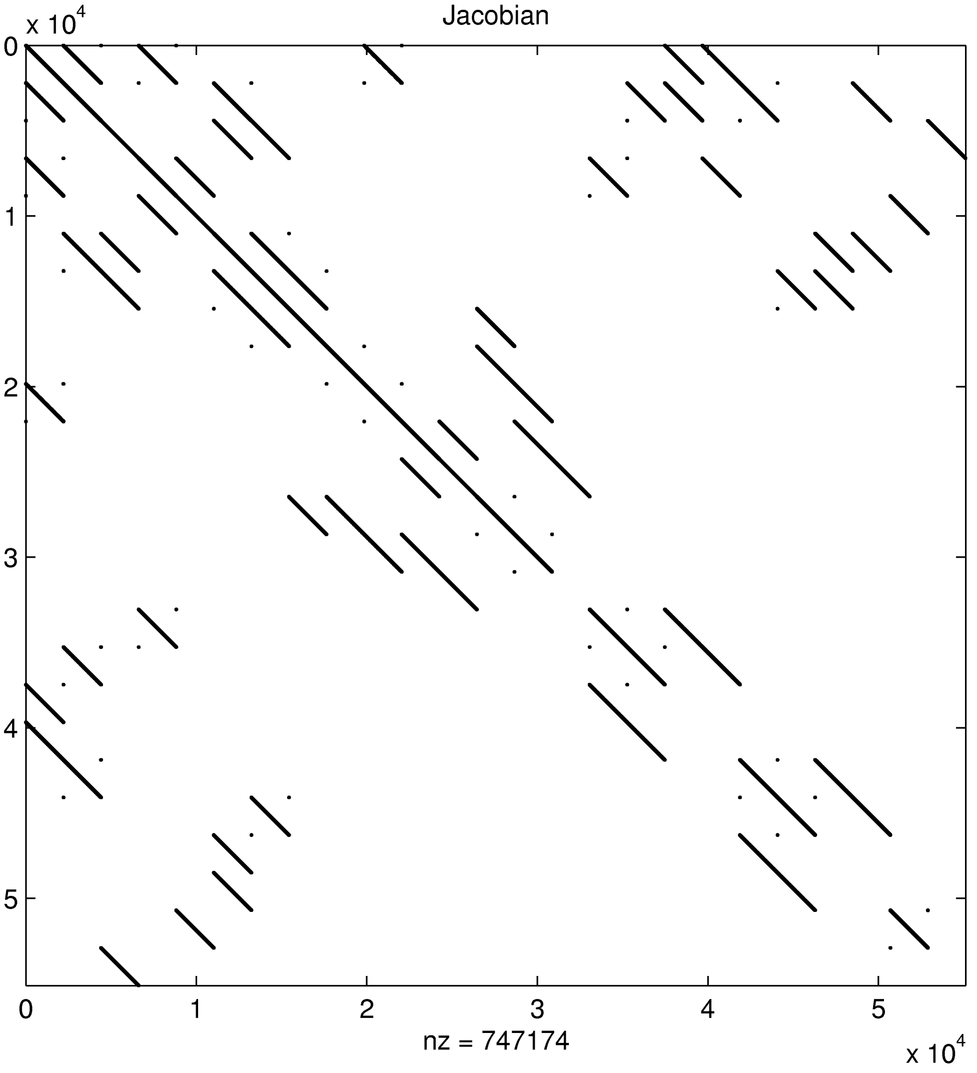


FIGURE 3.20. Sparsity pattern for the Jacobian arising from wavelet expansion for cascode LNA in Fig. 3.9 with $N_H = 12$.

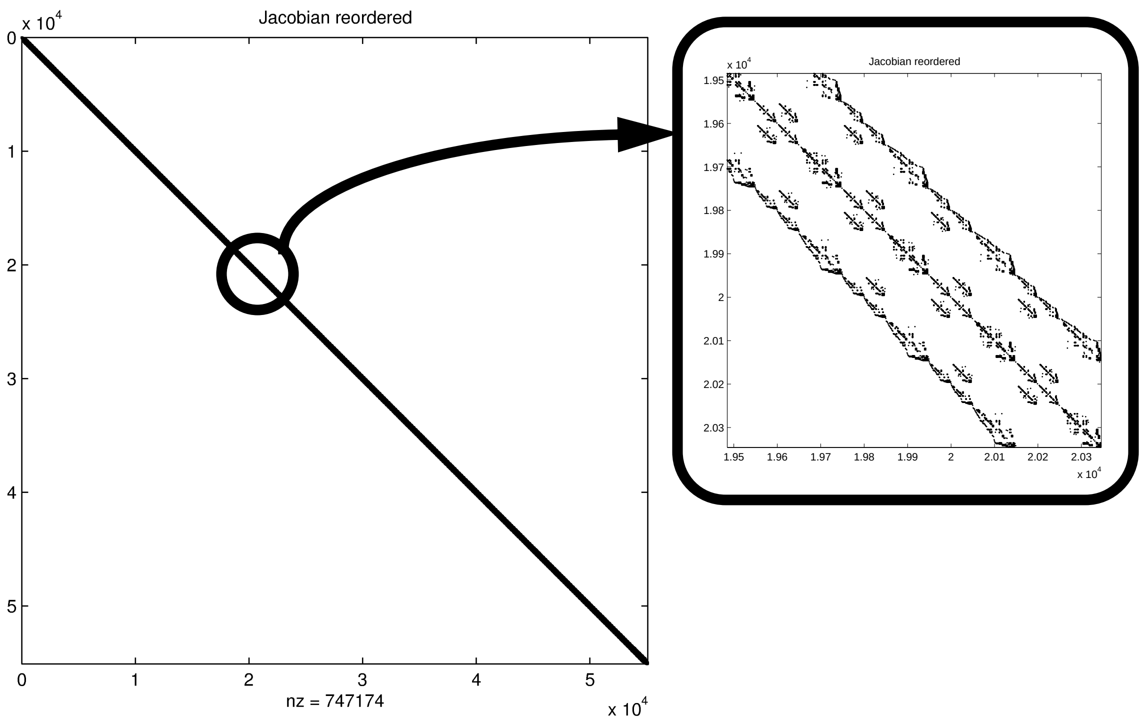


FIGURE 3.22. Jacobian in Fig. 3.20 after symmetric reverse Cuthill-McKee reordering. The matrix becomes bandlimited with a fairly narrow and sparse band.

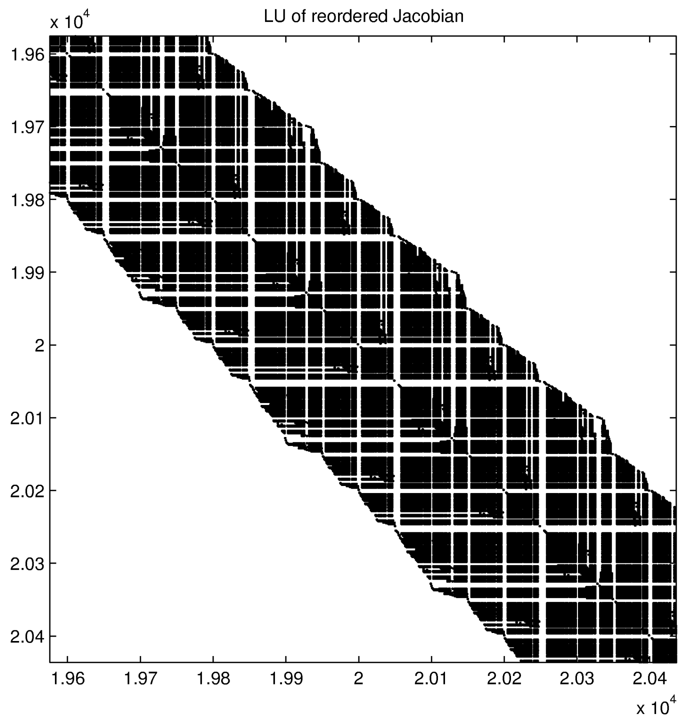


FIGURE 3.23. In-band sparsity pattern of the LU factors for SRCM-reordered wavelet Jacobian (Fig. 3.22).

3.4.2 CASE STUDY: GILBERT CELL MIXER

The second example involves a BJT Gilbert cell mixer circuit that consists of 9 transistors (including 3 as current sources), DC bias and impedance matching networks (Fig. 3.24). BJTs are represented by Ebers-Moll injection models (Fig. 3.10). Transformers are assumed to be ideal 1:1 converters. Under these assumptions total size of the MNA equations (3.1) is equal to 37.

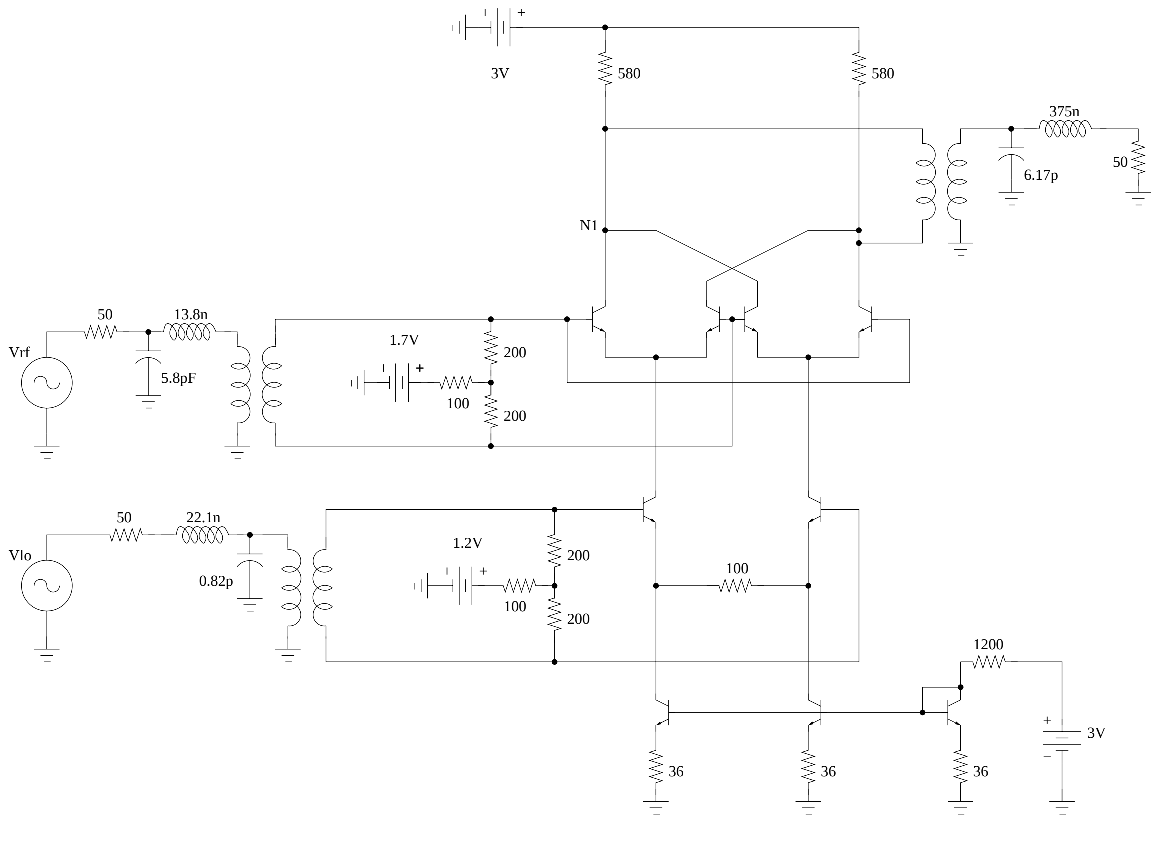


FIGURE 3.24. Gilbert cell mixer circuit

The mixer was configured for down conversion with LO input at 1 GHz, RF input at 900 MHz and IF output at 100 MHz. Inputs and output were matched to 50 Ohms active impedance at their respective frequencies.

The first experiment is a single tone LO input power sweep simulation often performed to estimate mixer operational point for LO bias. Results of the simulation (magnitude of second harmonic of voltage at node $N1$) are shown in Fig. 3.25. As was previously observed for single tone simulations, both HB and wavelet expansion exhibited essentially the same behaviour in terms of accuracy and convergence.

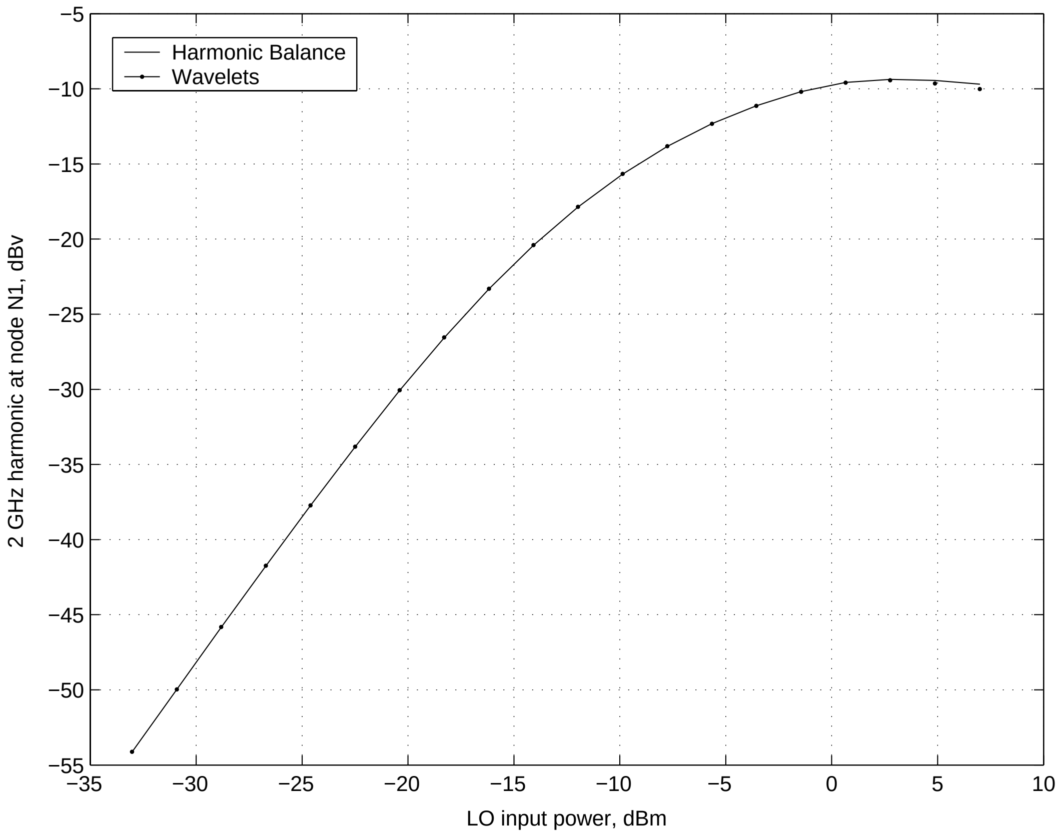


FIGURE 3.25. Single tone LO input power sweep for Gilbert cell mixer circuit in Fig. 3.24.

For the second experiment input LO power was kept constant at +1 dBm, while performing RF input power sweep. IF output power response for this simulation is shown in Fig. 3.26. Fig. 3.27 also illustrates convergence of the Newton iterations from DC solution to operation point P_{LO} = +1 dBm, P_{RF} = -33 dBm with N_H = 9. As can be seen from both plots, both HB and wavelet expansion exhibit essentially the same behaviour in terms of both accuracy and speed of convergence.

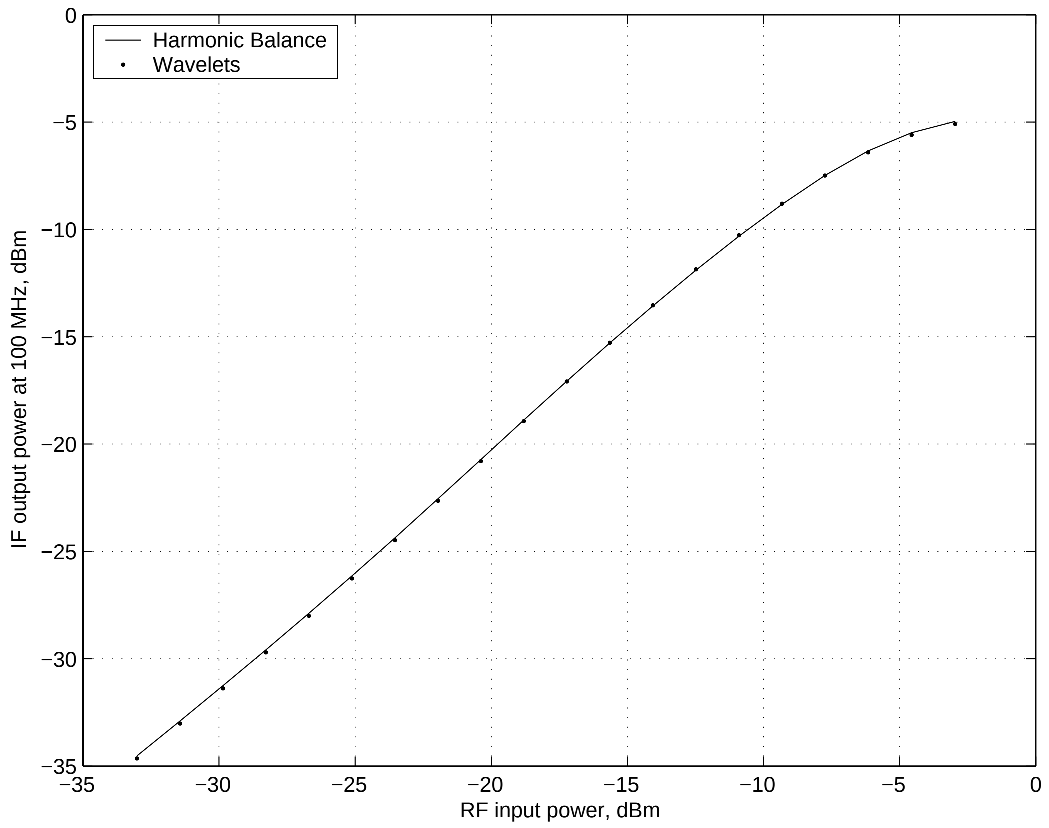


FIGURE 3.26. IF output power at 100 MHz with respect to RF input power for Gilbert cell mixer circuit in Fig. 3.24.

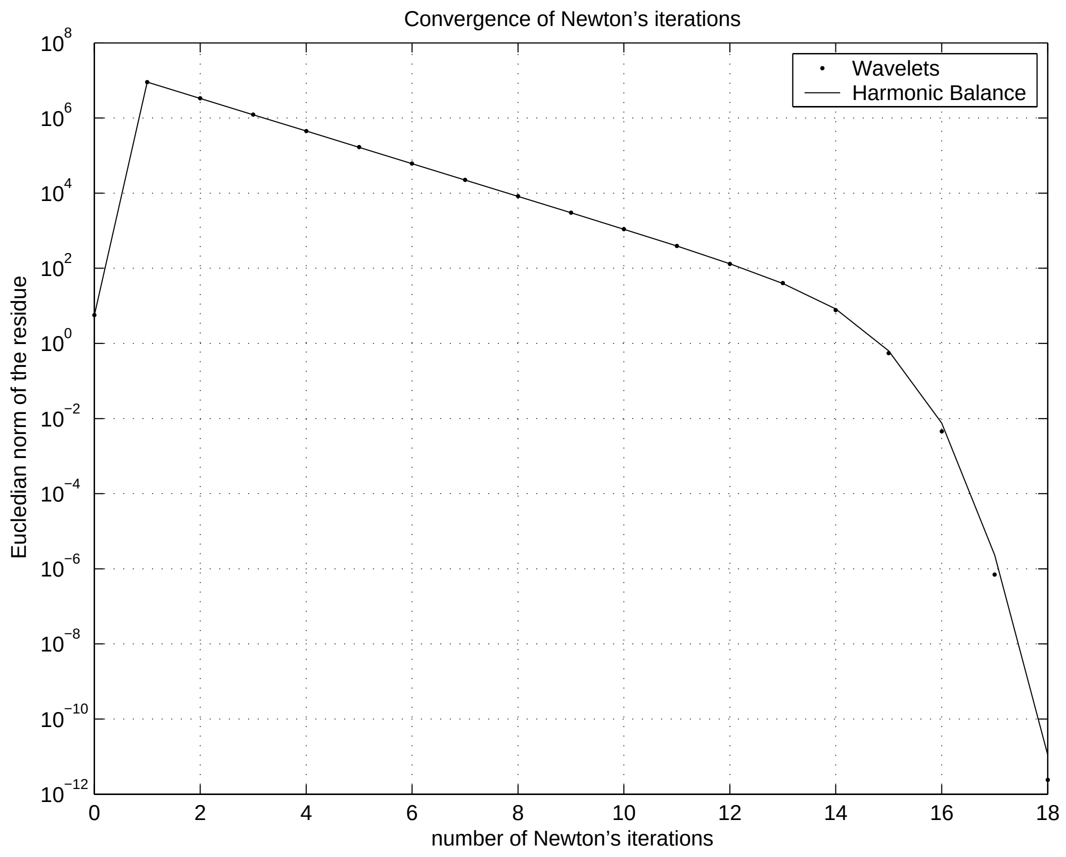


FIGURE 3.27. Convergence of the two tone simulation of the Gilbert cell mixer circuit in Fig. 3.24.

Comparison of the computational cost of the two methods in terms of the number of nonzero elements in the Jacobian and average time per LU decomposition is shown in Figs. 3.28 and 3.29 respectively. Both are in good agreement with the computational cost analysis performed in section 3.3. Because of the circuit configuration frequency grid density in this case is 10% ($\Delta\omega$ is 100 MHz with 900 and 1000 MHz fundamental frequencies), which means that for $N_H \geq 9$ diamond truncation frequency grid becomes "saturated" and diamond truncation degenerates into trivial truncation with $N_f = O(N_H)$ and $N_{NZ} = O(N_H^2)$.

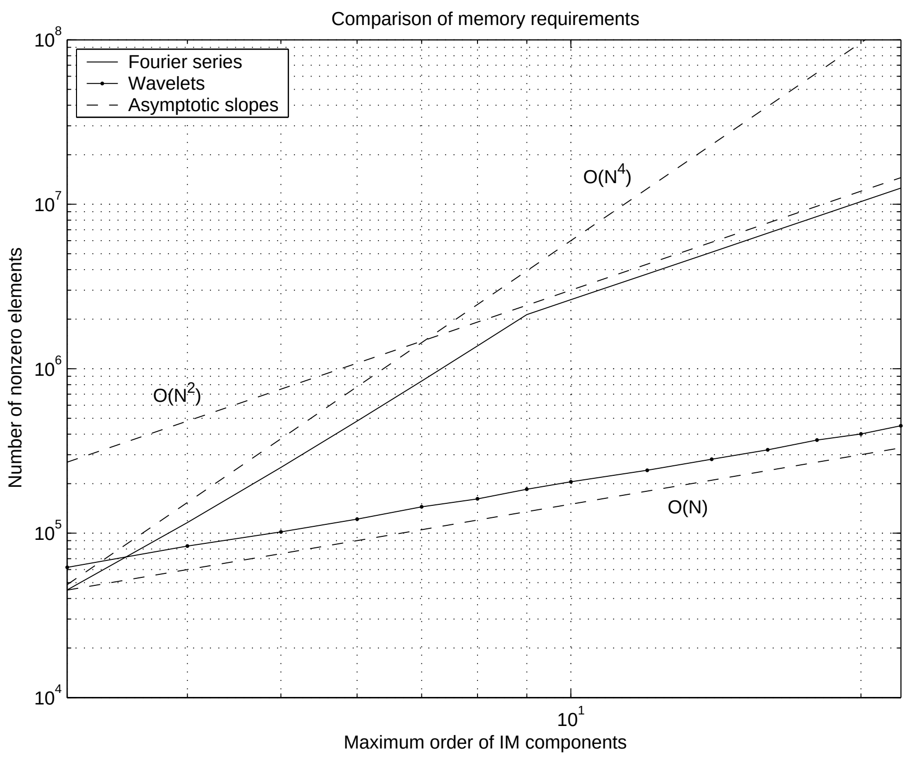


FIGURE 3.28. Number of nonzero elements in the Jacobian for Gilbert cell mixer in Fig. 3.24.

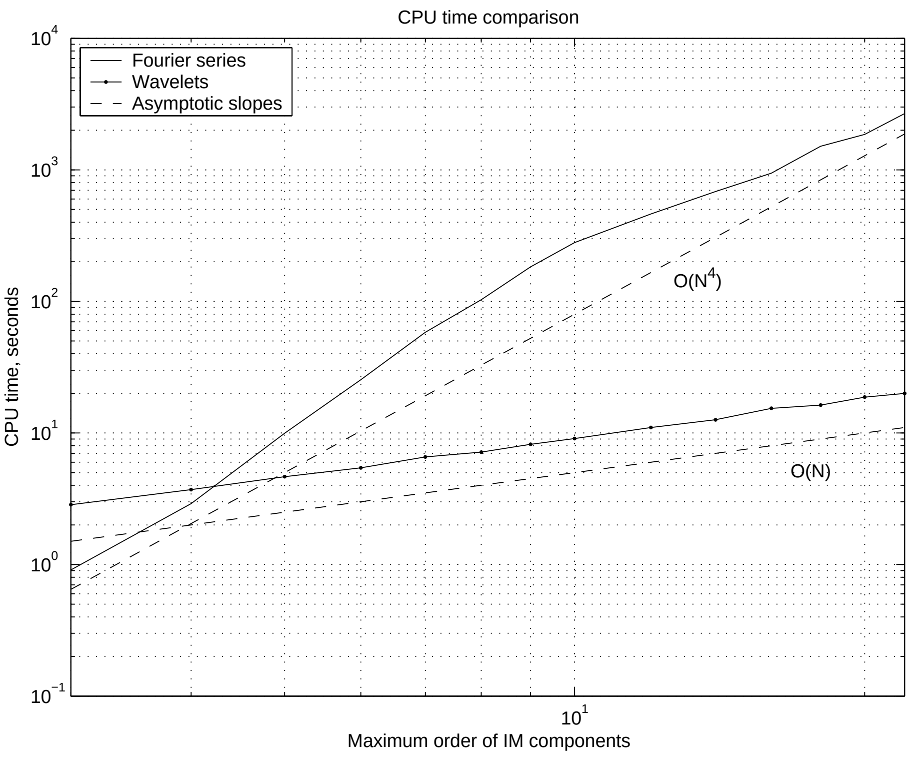


FIGURE 3.29. Average CPU time per LU decomposition of Jacobian for Gilbert cell mixer in Fig. 3.24.

Detailed computational cost data for $N_H = 6$, 9, 12 and 22 can also be found in tables 3.7-3.10.

TABLE 3.7. Computational cost comparison for Gilbert cell mixer (Fig. 3.24) at $N_H = 6$.

	Harmonic Balance	Wavelets
Frequency grid size	43	61
Time grid size	85	122
Jacobian size	6290×6290	9028×9028
Number of nonzero elements	480,135	121,695
Sparsity ratio	1.2%	0.15%
Memory storage size, MBytes	5.7	1.5
Average CPU time for preordering, seconds	0.66	0.07
Average time per LU/FBS, seconds	25.4	5.4
Number of Newton iterations	8	8

TABLE 3.8. Computational cost comparison for Gilbert cell mixer (Fig. 3.24) at $N_H = 9$.

	Harmonic Balance	Wavelets
Frequency grid size	91	91
Time grid size	181	182
Jacobian size	13394×13394	13468×13468
Number of nonzero elements	2,134,819	185,016
Sparsity ratio	1.2%	0.1%
Memory storage size, MBytes	25.1	2.3
Average CPU time for preordering, seconds	4.5	3.4
Average time per LU/FBS, seconds	185	8.2
Number of Newton iterations	8	8

TABLE 3.9. Computational cost comparison for Gilbert cell mixer (Fig. 3.24) at N_H = 12.

	Harmonic Balance	Wavelets
Frequency grid size	121	121
Time grid size	241	242
Jacobian size	17834×17834	17908×17908
Number of nonzero elements	3,767,955	241,356
Sparsity ratio	1.2%	0.075%
Memory storage size, MBytes	44.2	3.1
Average CPU time for preordering, seconds	9.4	5.6
Average time per LU/FBS, seconds	440	11
Number of Newton iterations	8	8

TABLE 3.10. Computational cost comparison for Gilbert cell mixer (Fig. 3.24) at N_H = 22.

	Harmonic Balance	Wavelets
Frequency grid size	221	221
Time grid size	441	442
Jacobian size	32634×32634	32708×32708
Number of nonzero elements	12,539,661	449,475
Sparsity ratio	1.2%	0.042%
Memory storage size, MBytes	147.2	5.7
Average CPU time for preordering, seconds	48	17
Average time per LU/FBS, seconds	2,720	20
Number of Newton iterations	8	8

Sparsity patterns for HB Jacobian, it's SAMD and SRCM reorderings and their LU factors are shown in Figs. 3.30-3.34 and the same for wavelet Jacobian in Figs. 3.35-3.38. This corroborates observation made in the previous section for cascode LNA that SRCM performs extremely well on wavelet Jacobians, while HB Jacobians represent quite a problem for both with MD algorithm providing better results.

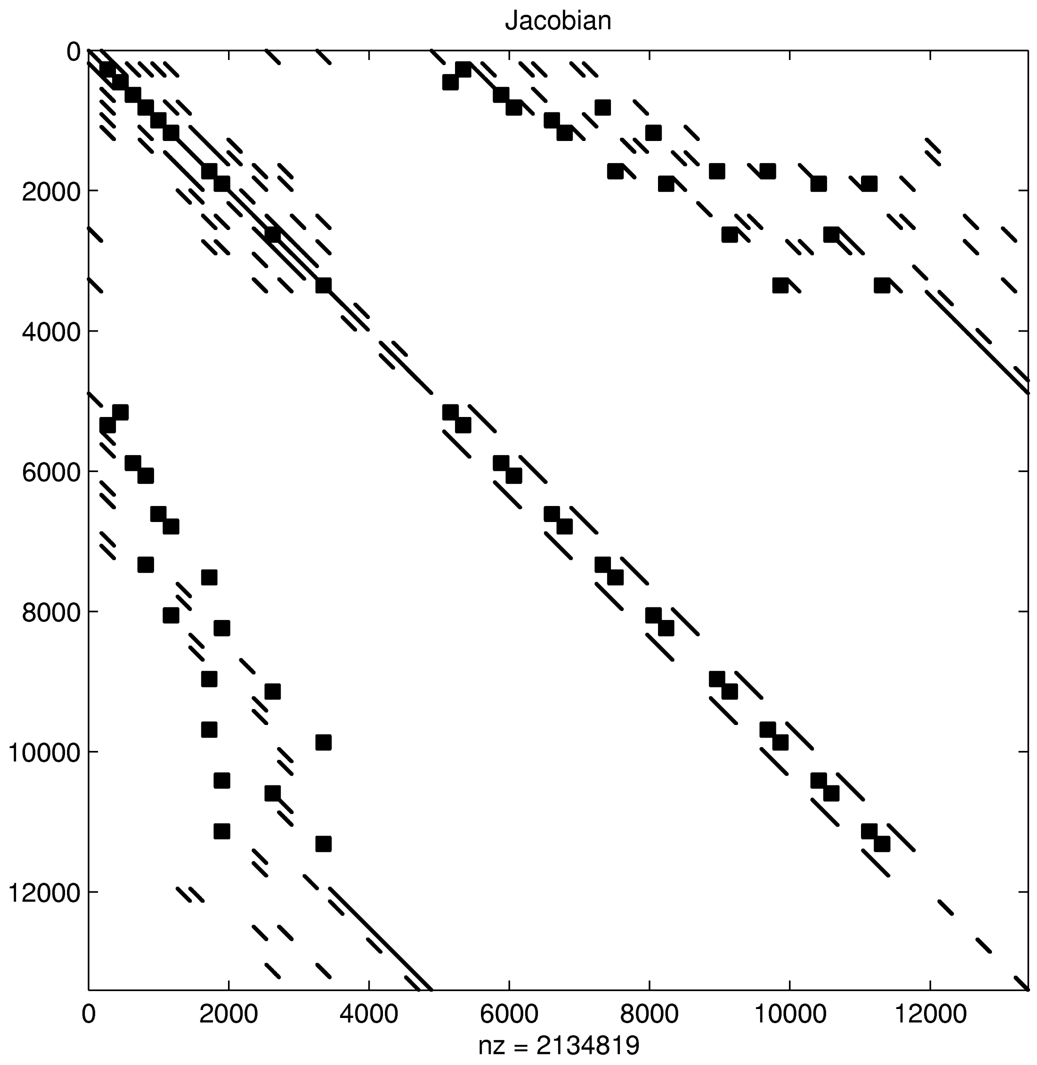


FIGURE 3.30. Sparsity pattern for the Jacobian arising from Fourier series expansion for Gilbert cell mixer in Fig. 3.24 with N_H=9. Dense blocks account for 98.2% of nonzero elements.

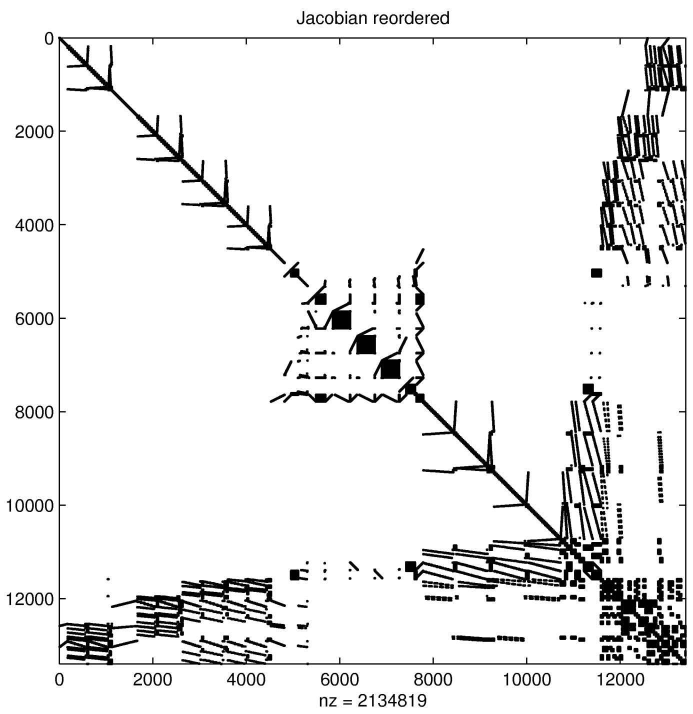


FIGURE 3.31. Jacobian in Fig. 3.30 after symmetric approximate minimum degree reordering.

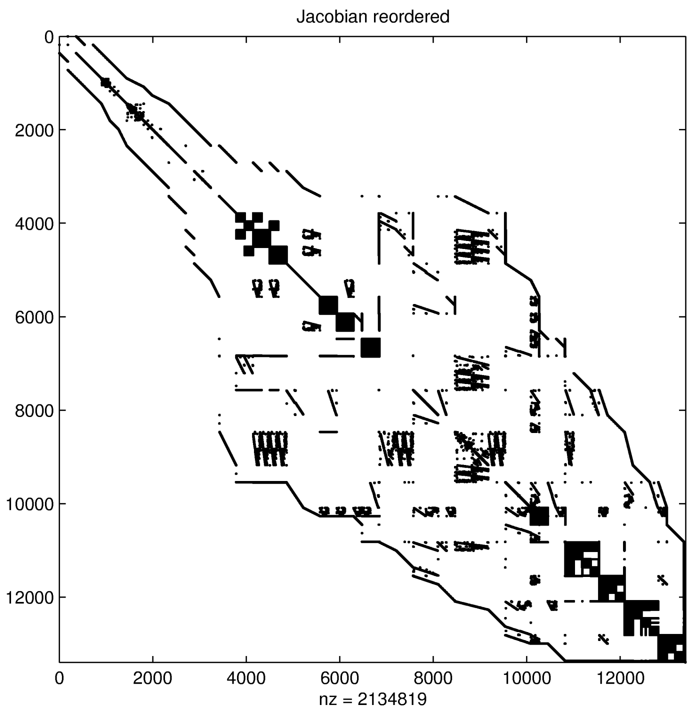


FIGURE 3.32. Jacobian in Fig. 3.30 after symmetric reverse Cuthill-McKee reordering.

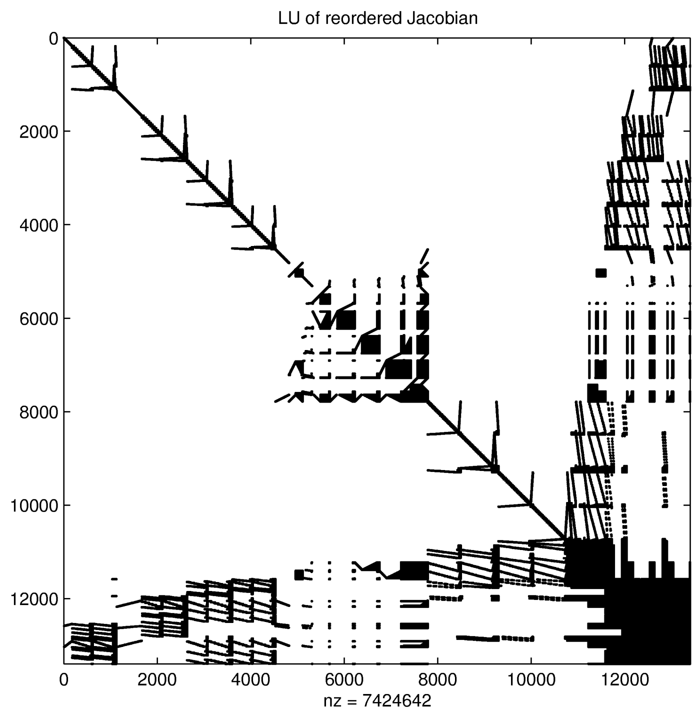


FIGURE 3.33. Sparsity pattern of the LU factors for SAMD-reordered HB Jacobian (Fig. 3.31).

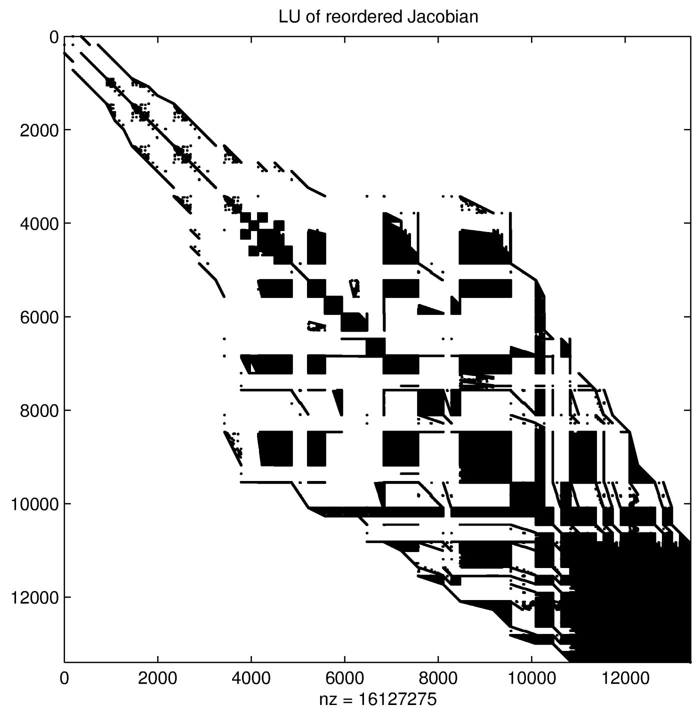


FIGURE 3.34. Sparsity pattern of the LU factors for SRCM-reordered HB Jacobian (Fig. 3.32).

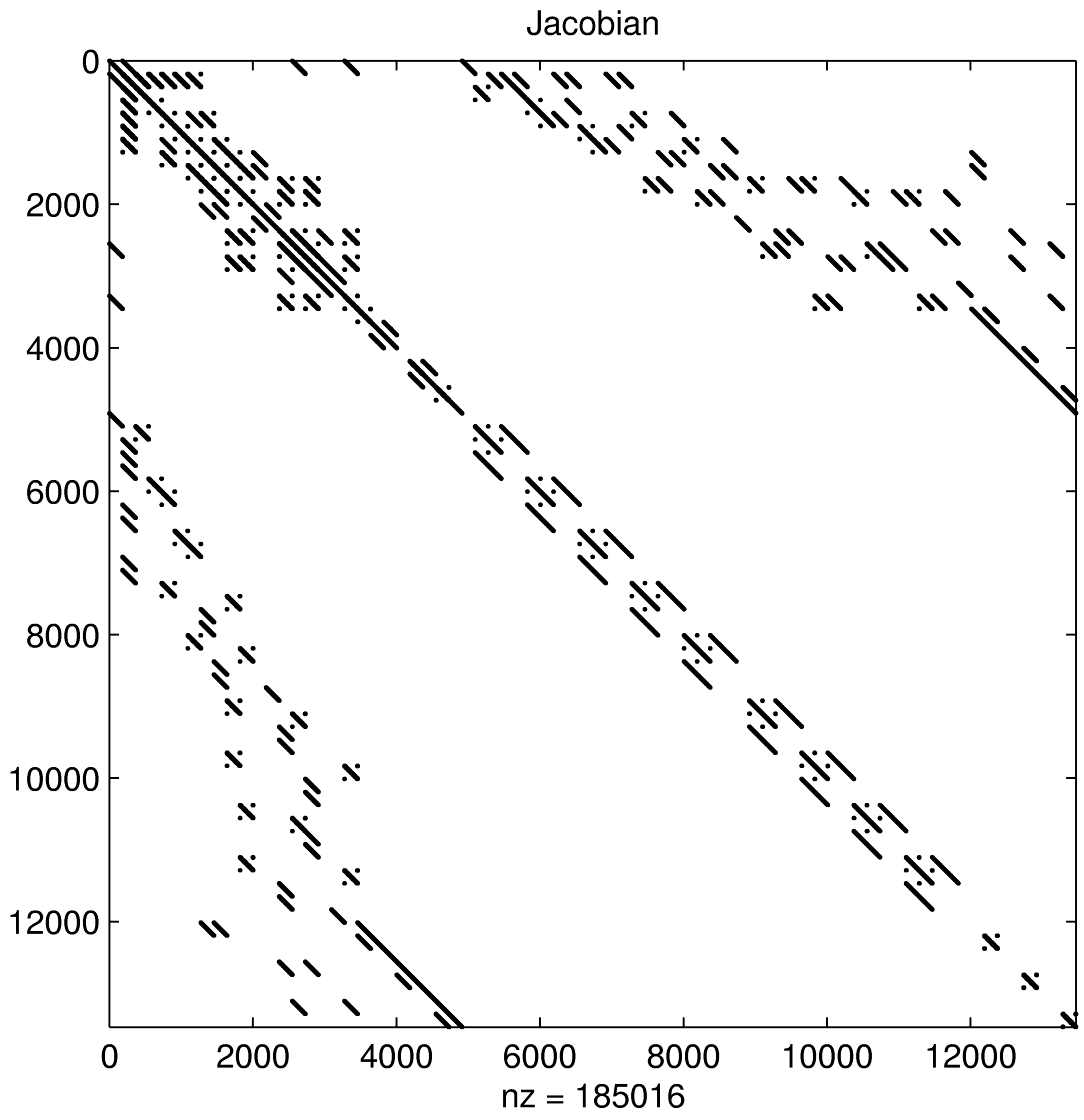


FIGURE 3.35. Sparsity pattern for the Jacobian arising from the wavelet expansion for Gilbert cell mixer in Fig. 3.24 with N_H=9.

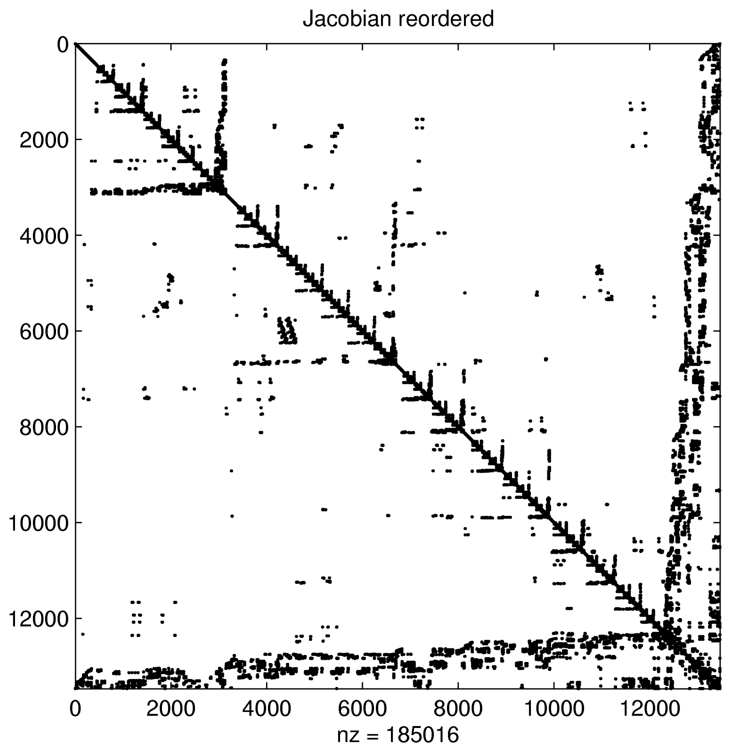


FIGURE 3.36. Jacobian in Fig. 3.35 after symmetric approximate minimum degree reordering.

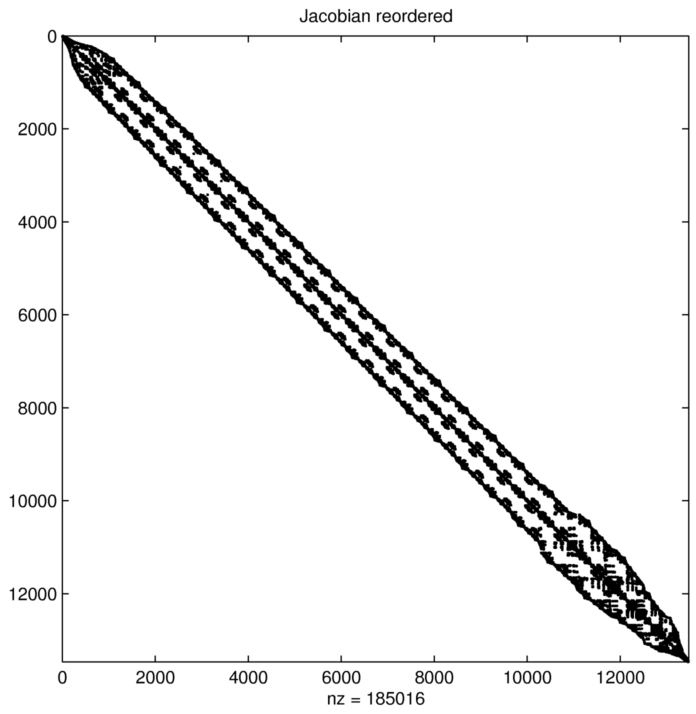


FIGURE 3.37. Jacobian in Fig. 3.35 after symmetric reverse Cuthill-McKee reordering.

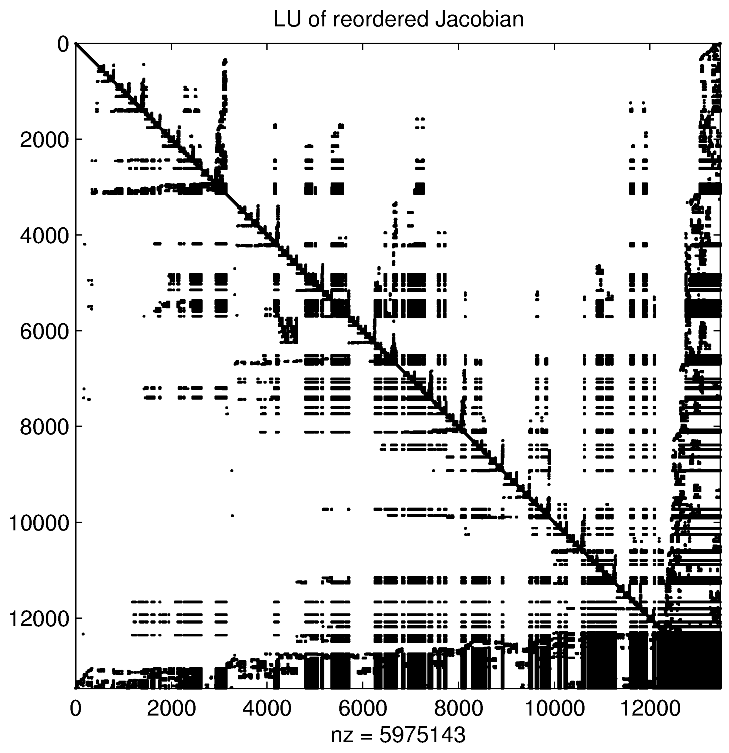


FIGURE 3.38. Sparsity pattern of the LU factors for SAMD-reordered wavelet Jacobian in Fig. 3.36.

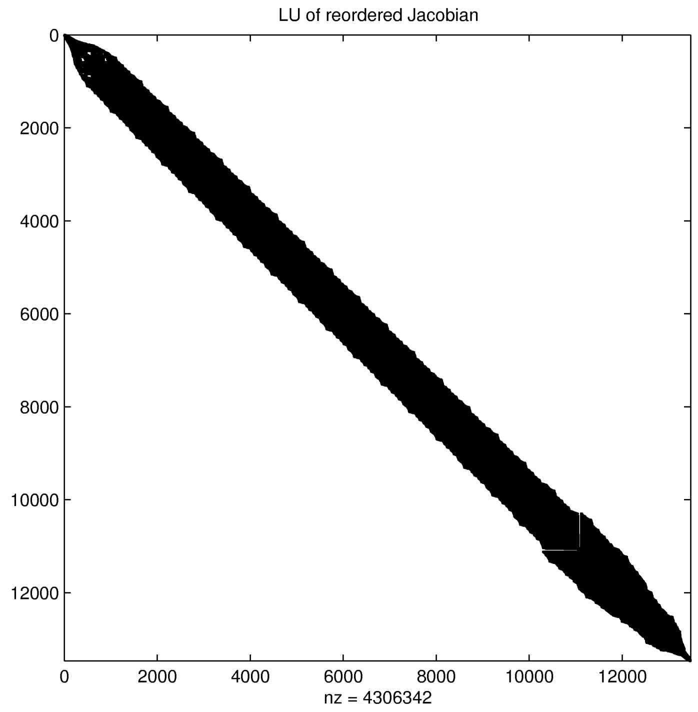


FIGURE 3.39. Sparsity pattern of the LU factors for SRCM-reordered wavelet Jacobian in Fig. 3.37.

3.5 CONCLUDING REMARKS

In this chapter we have proposed a new method for steady state analysis of nonlinear circuits under periodic excitations. The new method is similar to the well known technique of Harmonic Balance, but uses wavelets instead of Fourier series as expansion basis. The following features of the new method have been established:

- The new method retains accuracy and speed of convergence of the traditional technique.
- Use of wavelets allows for significant increase in sparsity of the Jacobian matrix and, consequently, in computational cost of the analysis, in terms of both storage requirements and CPU time.
- The new method has O(N) computational complexity and thus is particularly advantageous for large scale simulations.
- With respect to the maximum order of IM components retained in the simulation, the new method has also O(N_H) computational complexity, compared to O(N_H^4) for traditional Harmonic Balance. This makes the new method particularly suitable for simulation of highly nonlinear circuits.
- Computational complexity of the new method for multitone simulations does not depend on the number of tones, which makes it particularly suitable for multitone simulations with large number of tones.

- Computational complexity of the new method does depend of the density of the frequency grid, i.e. on the least common denominator of the fundamental frequencies. Therefore it is advantageous for simulation of wideband circuits where frequencies of the test tones can be moved around further to fall on a lower density regular spaced grid.

The new method has a full potential of producing a major impact on steady state analysis of large scale nonlinear circuits, as well as other systems described by nonlinear differential equations.

4. Other applications

In this chapter we will consider the other applications of wavelets to the problems encountered in EDA. Transient analysis of nonlinear circuits continues the lumped parameter circuit analysis started in Chapter 3, while the rest of the chapter is dedicated to the distributed parameter circuits.

4.1 Transient analysis of nonlinear circuits

In this section we review the traditional time marching schemes and then proceed with comparison to the wavelet methods. Simple analysis shows that in transient analysis it is very difficult for the wavelet methods to compete with traditional techniques. Numerical data from the literature supports this point of view.

4.1.1 Time marching methods

Transient analysis of nonlinear networks with lumped parameters is traditionally performed in the time domain using time marching numerical integration methods. The circuit is usually described by a set of coupled nonlinear differential and nonlinear algebraic equations in time domain (same as (3.1)):

$$C\dot{x} + Gx + f(x) + u = 0 \tag{4.1}$$

where C and G are $N_x \times N_x$ matrices containing imaginary and real parts of the nodal admittances, x is a $N_x \times 1$ vector of the unknowns, $f(x)$ is a $N_x \times 1$ vector containing nonlinear terms and u is a $N_x \times 1$ vector of independent sources. This is accompanied by a vector of initial conditions

$$x_0 = x(0) \tag{4.2}$$

thus creating an Initial Value Problem (IVP).

Because matrix C is in general case a singular matrix [18], the usual form of IVP ODEs

$$\dot{x} = -C^{-1}(Gx + f(x) + u) \tag{4.3}$$

does not exist. However, it is possible to separate differential equations in (4.1) from the algebraic equations, writing it in state space formulation

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned} \tag{4.4}$$

State space formulation leads to smaller number of unknowns, but is less convenient to implement for large circuits, produces denser matrices and often is not beneficial in terms of computational cost.

Equation (4.1) can be discretised using any of the available numerical integration techniques, producing a set of nonlinear algebraic equations that has to be solved at each time point. For example, Backward Euler (BE) formula with time step $h = t_{n+1} - t_n$ and $x_n = x(t_n)$ produces the following equation:

$$\Phi(x_{n+1}) = \frac{1}{h}C(x_{n+1} - x_n) + Gx_{n+1} + f(x_{n+1}) + u(t_{n+1}) = 0 \tag{4.5}$$

which can be solved for x_{n+1} using Newton's iterations:

$$J(x_{n+1}^{(i+1)})(x_{n+1}^{(i+1)} - x_{n+1}^{(i)}) = -\Phi(x_{n+1}^{(i)}) \tag{4.6}$$

In this case, Jacobian matrix J is given by

$$J(x) = \frac{\partial\Phi}{\partial x} = \left(\frac{1}{h}C + G\right) + \left[\frac{\partial f}{\partial x}\right] \tag{4.7}$$

and it is an $N_c \times N_c$ sparse matrix, which sparsity is completely determined by the structure of the original equation (4.1). If h is sufficiently small, a good initial guess for $x_{n+1}^{(0)}$ can be easily obtained by extrapolating x_n, which provides quadratic convergence for Newton's iterations (4.6).

Similar equations can be written for other numerical integration schemes, but they all share the fact that Jacobian to be solved at each iteration is an $N_c \times N_c$ sparse matrix and it's sparsity is completely determined by the structure of (4.1), i.e. by the topology of the underlying circuit. Time marching methods are rather well developed and their software implementation is very efficient.

4.1.2 WAVELET EXPANSION

Wavelet expansion for transient analysis is essentially similar to the one for steady state analysis (introduced in Chapter 3), except for the boundary conditions. In steady state analysis, periodic boundary conditions (3.2) had to be imposed on the expansion basis. With transient analysis being an IVP, there is usually no *a priori* information on the state of the unknowns at the end of analysis interval $t \in (0, t_{max})$. As so, expansion basis has to be able to approximate any value of $x(t_{max})$ that is reasonable under the circumstances. This can be achieved by using wavelets on an interval with boundary adapted basis functions to preserve (bi)orthogonality [8]. Construction of the transform matrices T and $\tilde{T}$ is slightly different from that of described in Chapter 3, but they still are bandlimited matrices. Apart from that, solution is similar to the steady state case. The expanded equation, as before, is

$$\Phi(X) = (\hat{C}D + \hat{G})X + F(X) + U = 0 \tag{4.8}$$

where $\hat{C}$, D and $\hat{G}$ are $N_tN_x \times N_tN_x$ matrices and N_t is the number of basis functions, also equal to the number of time points. Equation (4.8) is solved by performing iterations and at each iteration Jacobian matrix to be solved is

$$J(X) = \hat{C}D + \hat{G} + T\left[\frac{\partial f_k}{\partial x_l}\right]\tilde{T} \tag{4.9}$$

which is also an $N_tN_x \times N_tN_x$ sparse matrix with sparsity structure similar to the Jacobian matrix used in steady state analysis. Although this matrix is only O(N_t) dense, it is easy to see that it is N_t times bigger than the Jacobian for time marching methods which imposes a computational cost penalty on expansion techniques. Another disadvantage of expansion techniques is that the initial guess now has to be obtained not at a single time point, but at all time points at once. This is quite a difficult task by itself.

Wavelet methods for transient simulation have been researched by Zhou et al. in [36]-[39] and independently by Steer and Christoffersen in [26], [40] and [41]. A wavelet collocation method described in [38] features B-spline-based biorthogonal wavelets, uniform error distribution and O(h^4) convergence (versus O(h^2) for popular time marching methods like Runge-Kutta). However only linear circuits were considered in that publication. Adaptive wavelet scheme for nonlinear circuits was described in [38] where higher scale wavelets are introduced only at those locations where it is necessary to maintain accuracy and convergence of the scheme. However, no direct comparison of computational cost with time marching schemes was provided and numerical results (not surprisingly) showed that CPU time increases dramatically with the increase in the length of the time interval (e.g. two times increase in interval length resulted in 5-6 times increase in CPU time).

These results are further corroborated by Steer and Christoffersen for state space formulation. For example, direct comparison between wavelet collocation expansion and time marching schemes (Backward Euler) was made in [40] and [41]. Backward Euler formula follows from (4.8) by assuming transform matrices $T = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ and $\tilde{T} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$. This excludes inaccuracies in CPU time measurement that may result from different implementations, although puts time marching method at a slight disadvantage as Jacobian in this case is twice as large as for the traditional BE formulation (4.7). In conclusion, it was noted that

> “Although this implementation of a time marching transient analysis is inefficient, it is several times faster than the simulation using wavelets. This is because the solution of the nonlinear system in wavelet transient presents two problems: the initial guess in the Newton method is not close to the solution and the number of nonlinear unknowns grows quickly with the window resolution despite the state-variable reduction.”

We can conclude from the above that although wavelet methods for transient analysis show certain potential, particularly in view of adaptive techniques, it will be very difficult for them to compete against well engineered and well established time marching schemes. Advantage of the steady state wavelet methods over Harmonic Balance is primarily due to the significant increase in sparsity of the Jacobian, and that's even before adaptive schemes are introduced. In transient analysis there is no such advantage for wavelet methods in sight and a lot of research into improvement of wavelet adaptive schemes has to be done before they can compete with time marching methods.

4.2 TRANSMISSION LINE MACROMODELLING

With continuously increasing clock speeds and chip density, analysis of interconnects for signal integrity validation becomes extremely important [57], [58]. Distributed parameter interconnects in the form of multiconductor transmission lines under quasi-TEM assumption are described by a set of Partial Differential Equations (PDEs) in time-spatial domain known as the Telegrapher's equations [57]:

$$\frac{\partial}{\partial z}\begin{bmatrix} v(z,t) \\ i(z,t) \end{bmatrix} = -\left(\begin{bmatrix} 0 & R(z) \\ G(z) & 0 \end{bmatrix} + \begin{bmatrix} 0 & L(z) \\ C(z) & 0 \end{bmatrix}\frac{\partial}{\partial t}\right)\begin{bmatrix} v(z,t) \\ i(z,t) \end{bmatrix} \tag{4.10}$$

Where v and i are $N_c \times 1$ vectors of voltages and currents that are functions of both time *t* and position along the line *z*; *R, L, C* and *G* are $N_c \times N_c$ matrices of per-unit-length parameters and N_c is the number of conductors not including the ground. For transmission lines with non-translation-invariant cross-sections (non-uniform lines), per-unit-length parameter matrices are functions of *z*. In case when frequency dependent parameters (e.g., due to skin effect) have to be taken into account, matrices *R, L, C* and *G* can also be functions of frequency.

By denoting length of the line as d and taking Laplace transform of (4.10) in time domain, we can solve (4.10) in frequency domain for terminal voltages and currents and write a set of linear algebraic equations coupling them [42] in terms of y-parameters:

$$\begin{bmatrix} i(0) \\ -i(d) \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix} \begin{bmatrix} v(0) \\ v(d) \end{bmatrix} \tag{4.11}$$

or hybrid ABCD parameters:

$$\begin{bmatrix} T_{11} & -I \\ T_{21} & 0 \end{bmatrix} \begin{bmatrix} v(0) \\ v(d) \end{bmatrix} + \begin{bmatrix} T_{12} & 0 \\ T_{22} & -I \end{bmatrix} \begin{bmatrix} i(0) \\ i(d) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{4.12}$$

where I is the $N_c \times N_c$ identity matrix.

The problem with such formulation is that the circuit equations for the nonlinear lumped parameter network

$$C\dot{x} + Gx + f(x) + u = 0 \tag{4.13}$$

are inherently nonlinear and thus can not be represented in frequency domain, while transmission line stamps (4.11) and (4.12) are in frequency domain and do not have a direct representation in terms of time domain ODEs.

The goal of constructing a macromodel is to convert (4.10) to a set of ordinary differential equations (ODEs) in time domain, such that they can be coupled with equations (4.13) for the nonlinear lumped parameter network and solved together in a circuit simulator. This can be performed by expanding equations (4.10) in a suitable basis along the spatial domain. Results of this approach published in the literature include expansion in Chebyshev polynomial basis [44] and biorthogonal wavelet bases on an interval [45]-[47]. Paper [48] also explores time domain expansion of (4.10) in terms of orthogonal Daubechies basis, which is only suitable for uniform transmission lines. Both time domain and space domain wavelet expansion are further explored in [49], while [50] combines time domain expansion in wavelet terms with Finite Element-based spatial domain expansion to handle nonuniform transmission lines.

However, in order to avoid stability problems during simulations, a transmission line macromodel has to be passive *by construction* or a proof of passivity has to exist for it [42]. Transmission lines are essentially passive systems. Transmission lines can be represented by non-passive, but stable by themselves models, but when such models are coupled with various terminating networks they can become unstable and produce non-physical solutions.

Wavelet methods are potentially more efficient than other expansion methods, particularly for highly nonuniform transmission lines, but until a proof of passivity (e.g. like in [43] for rational matrix approximation) is found, they probably will not find any significant use in EDA tools. The fact that Finite Difference expansion macromodels are passive by construction is encouraging (in view of FD being equivalent to discretisation in Haar basis), but more research is needed in this direction.

4.3 CAPACITANCE EXTRACTION

4.3.1 PROBLEM BACKGROUND

Interconnect parameter (matrices *R*, *L*, *C* and *G* in (4.10)) extraction problem represents one of the most computationally expensive steps of current high-speed circuit design process. Calculating capacitance matrices as well as matrices of other physical interconnect parameters is the foundation of interconnect analysis. Configuration of interest includes multiconductor transmission lines in multilayered dielectric media [70] (Fig. 4.1). The transmission line consists of a number of conductors of arbitrary shapes. Some of the conductors can be of a finite cross section, while others are infinitesimally thin. Conductors can be touching a dielectric interface, straddling a dielectric interface or be totally embedded in a single dielectric layer. Configuration is assumed to be uniform in the direction of *z* axis. Dielectric interfaces are all assumed to be parallel to the *x*-*z* plane, which is usually the case for on-chip interconnects, as well as PCB level interconnects. This configuration can be bounded by ground planes on either side, any two or three sides or on all four sides.

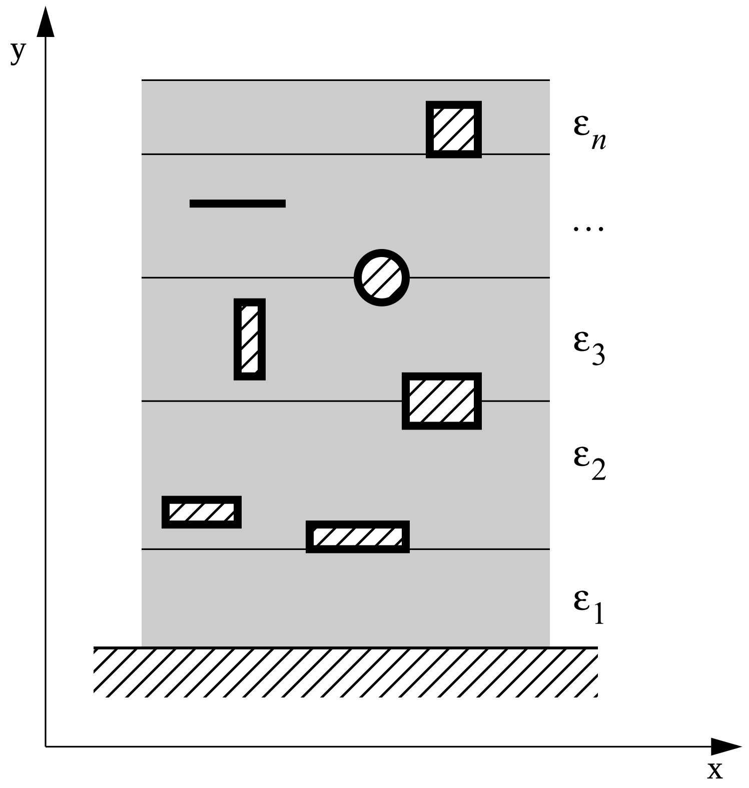


FIGURE 4.1. Multiconductor transmission line in arbitrary multilayered dielectric media.

Existing methods of capacitance calculations can be loosely split into two categories.

The first category involves calculation of electrostatic field by solving the Laplace equation:

$$\nabla^2\phi = 0 \tag{4.14}$$

together with associated boundary conditions. This is solved by Finite Elements Method (FEM) or Finite Differences Method (FDM). These discretisation techniques typically result in sparse matrices, but the dimension of these matrices dramatically grows with the geometrical complexity and condition number of the matrix deteriorates accordingly. Because of these disadvantages, this method is not widely used.

The second category involves solution of the integral equation for superficial charge density:

$$\int_{\Omega} \sigma(r')G(r, r')dr' = v(r) \qquad r, r' \in \Omega \tag{4.15}$$

Where G is the kernel of linear integral equation (Green's function associated with (4.14)), σ is superficial charge density and Ω is the domain of interest (in this case - surface of the conductors). Discretisation of (4.15) is usually performed by Method of Moments (MoM) or its variations, such as Boundary Elements Method (BEM) ([1], [2]). Stiffness matrices resulting from MoM/BEM are typically smaller than the ones for FEM/FDM, but in general these are full matrices and for practical 2D and 3D problems, computational cost O(N^3) of solving such matrices is often prohibitive for use on day-to-day basis in CAD tools. This prohibits today state-of-the-arts VLSI physical parameter extraction software from using inline field solvers in favour of precalculated libraries for typical geometries with empirical matching and calibration procedures for mapping actual geometry into library patterns [59]. Existing general purpose field solvers simply are not flexible enough to provide an efficient trade off of accuracy for execution time [60]. In fact, if a BEM equation does not converge to a suitable accuracy, expansion basis should be updated and a new MoM matrix should be assembled. Denseness of this matrix as well as non-compactness of the kernel renders iterative matrix solvers to be unreasonably expensive to use [61].

A great deal of research has been done in accelerating BEM techniques. Examples of such fast algorithms are: applications of fast *n*-body problem algorithms including multipole accelerated algorithm [62] and Appel's hierarchical algorithm [65]. These algorithms have O(N) cost per iteration for $1/\|x - x'\|$ kernels, but exhibit significant slowdown for hypersingular (e.g. $\log(1/\|x - x'\|)$) operators and therefore are not so effective for practical problems on bounded domains (layered dielectrics, ground planes, etc.) where in this case EM interfaces should be explicitly taken into account by calculating bounded charge densities. Singular Values Decomposition (SVD) accelerated algorithm of Kapur et al. [63] is efficient and independent of kernel type, but in general case it still requires O(N^2) operations for constructing the matrix and O($N\log N$) operations per iteration for solving it. These methods use Galerkin discretisation in pulse functions basis. Precorrected-FFT algorithm [64] is similar to SVD algorithm in terms of efficiency and is based on collocation technique.

The other main reason for the high computational cost of capacitance extraction with traditional approaches is the need to solve for the field distribution or superficial charge density on the whole domain of interest (as it is always the case with pulse expansion basis and collocation methods). However, knowing only the *total surface charge (per conductor or per unit length)* is quite sufficient for calculating capacitance matrix. In this section a new approach is presented that couples this idea with wavelet analysis and features sparse representation of single- and double-layer potential and hypersingular kernels, accurate calculations of total charge without obtaining charge density per se and easy trade of speed for accuracy.

Proposed algorithm is based on the recent advances in approximation theory which allow construction of orthogonal wavelet bases on $\boldsymbol{L}^2(\Re)$ and $\boldsymbol{L}^2(0, 1)$ that also have limited support in both time/spatial and frequency domain (Section 2.4). First achievements of wavelet applications with respect to numerical analysis were in connection with sparse representation of Calderon-Zygmund type kernels, preconditioning of stiffness matrices arising from discretisation of pseudodifferential equations and fast matrix-vector multiplication algorithms ([11], [12], [13], [14]). Later advances also dealt with construction of wavelets on an interval, operator wavelets diagonalising certain types of operators [15], and quite recently, with discretisation of singular and hypersingular kernels and convergence of sparse systems obtained for such types of kernels ([54], [55], [56]).

The first attempt of approaching capacitance extraction problem from the wavelet point of view was done by Wang et. al. [67] with the main focus on handling arbitrary geometries by combination of boundary element and conformal mapping techniques. Here we extend that technique with *principal emphasis on the computational efficiency*. The principal differences of the proposed approach from the previously published wavelet-based technique are:

- Wavelet expansion together with the proposed matrix thresholding strategy results in *extremely sparse matrix*, which leads to substantial gain in CPU time and storage requirements.
- Wavelets by themselves have zero average and directly contribute only to the charge distribution on conductor and not to the total charge. This allows us to *compute total conductor surface charge from the scaling function coefficients without paying high costs of computing exact charge distribution*.
- Iterative construction of the equation matrix in wavelet basis allows *easy and efficient trade of speed for accuracy*.

Together with simultaneous utilization of other key acceleration ideas [62]-[65], these concepts provide potential for an “inline” capacitance extraction engine for very large scale problems.

4.3.2 Wavelet expansion of integral equations

We start with construction of orthonormal periodical wavelet basis on $\boldsymbol{L}^2(0, 1)$ [67]. Define $\varphi_{j,k}(x)$ to be scaling function and $\psi_{j,k}(x)$ to be wavelet function (section 2.3.5).

For wavelets with local support periodical basis on the interval can be easily created by choosing $\varphi_{0,0}$ such, that

$$\operatorname{supp}\varphi_{0,0} = (0, 1), \tag{4.16}$$

where **supp** is support operator, and simply deleting wavelet functions and scaling functions outside this interval [67]. In this case, utilization of periodic wavelets does not sacrifice orthogonality and all the multiresolution properties defined in section 2.3.5 as well as regularity and moment properties outlined in section 2.4.2. The only difference is that we start multiresolution analysis from V_0 and go up to V_J. For convenience, we renumber wavelet basis in the following manner:

$$u_1 = \varphi_{0,0},\ u_2 = \psi_{0,0},\ u_3 = \psi_{1,0},\ u_4 = \psi_{1,1},\ \ldots,\ u_{2^J} = \psi_{J, 2^{J-1}}. \tag{4.17}$$

This basis can be further scaled

$$\operatorname{supp}\varphi_{0,0;i} = \Omega_i \tag{4.18}$$

and normalized, such that it covers every spatial subdomain $\Omega_i, i = \overline{1 \ldots I}$, where I is the total number of elementary conductors, each subdomain being surface of a conductor subject to calculation of capacitance. For conductors of a regular shape (rectangular, circular, etc.) number of elementary conductors is equal to the number of physical conductors. Otherwise, it will depend on decomposition of conductor shape for applying conformal mapping technique [67].

Let us define inner product (section 2.1.2) as:

$$\langle f, g\rangle = \int f(x)g(x)dx \tag{4.19}$$

For the capacitance extraction problem, linear operator in equation (2.27) takes the form integral operator on an interval:

$$Lf = \int_{\Omega_i} f(r')G(r, r')dr' \tag{4.20}$$

where G is defined as in (4.15).

For integral operators in general, matrix in equation (2.33) is a full matrix [61], with the cost of assembling such matrix itself being of O(N^2) and cost of solving equation being of O(N^3). This cost can be substantially reduced by choosing orthogonal wavelets with compact support as basis functions and successive thresholding of stiffness matrix that results in sparse matrix format [14].

4.3.3 ELECTROSTATIC KERNELS AND THRESHOLDING TECHNIQUES

Consider discretization of an electrostatic problem kernel (Fig. 4.2):

$$G(x, x') = \frac{1}{|x - x'|} \tag{4.21}$$

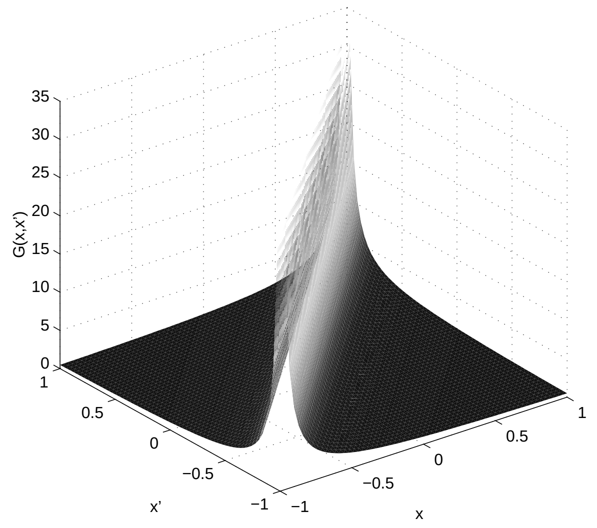


FIGURE 4.2. Typical electrostatic problem kernel.

on interval (-1,1) in Daubechies basis (Fig. 4.3) by means of Galerkin expansion. This kernel is a typical representative of the class of kernels arising from electrostatic problems, such as capacitance extraction. From EM point of view this kernel represents electrostatic potential due to a single point charge in free space. Integral equation kernels for arbitrary geometry can be represented by a general linear combination of spatial dilations of this kernel. This kernel has a pronounced singularity along $x = x'$ and it's Galerkin-wavelet stiffness matrix (Fig. 4.4 - note log scale on the vertical axis) has numerically significant elements only when corresponding pair of basis functions is at scales and locations in the vicinity of this singularity. This is quite understandable, considering local support and vanishing moments of Daubechies wavelets (section 2.4.2). Outside of these locations, elements of the stiffness matrix are orders of magnitude lower and can be successfully discarded with losing very little information about the original operator (Fig. 4.5).

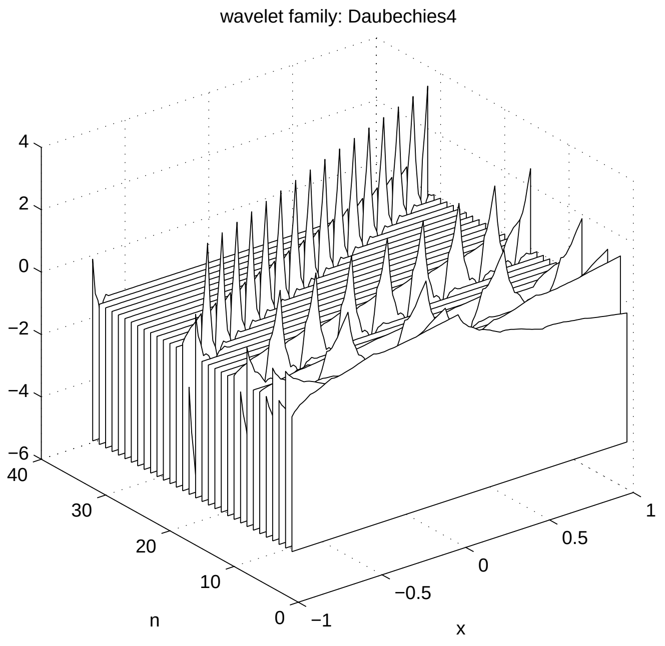


FIGURE 4.3. Daubechies second order basis on an interval (resolution levels 0 to 4).

However, for electrostatic problems on bounded domains, such as conductors embedded in multilayered dielectrics and/or between ground planes, the straightforward approach of using Green's function (4.21) if

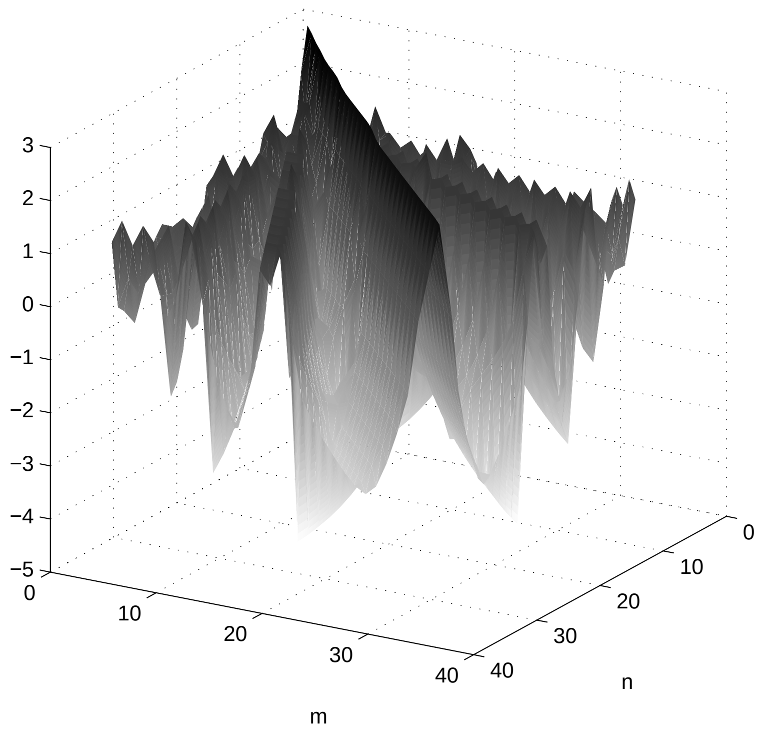


FIGURE 4.4. The stiffness matrix for singular electrostatic kernel (Fig. 4.2) discretised in Daubechies basis (Fig. 4.3). Note log scale on vertical axis.

simple, is not the most advantageous. With multiple dielectric-dielectric and/or dielectric-conductor interfaces free charges experience multiple reflections. Equivalent Green’s function becomes a weighted sum of (4.21) and exhibits significantly more pronounced singularity and slower decay. A usual workaround is to use simple Green’s function (4.21) and take into account interfaces by explicitly computing bounded charge densities. Such approach leads to significant increase in number of unknowns for a problem that is already large dimensioned. The other approach is to use generalized Green’s functions that incorporate all boundary conditions thus effectively representing bounded charge in implicit way. Generalized Green’s functions [68], [69] with decay slower than $1/\|x-x'\|$ belong to the

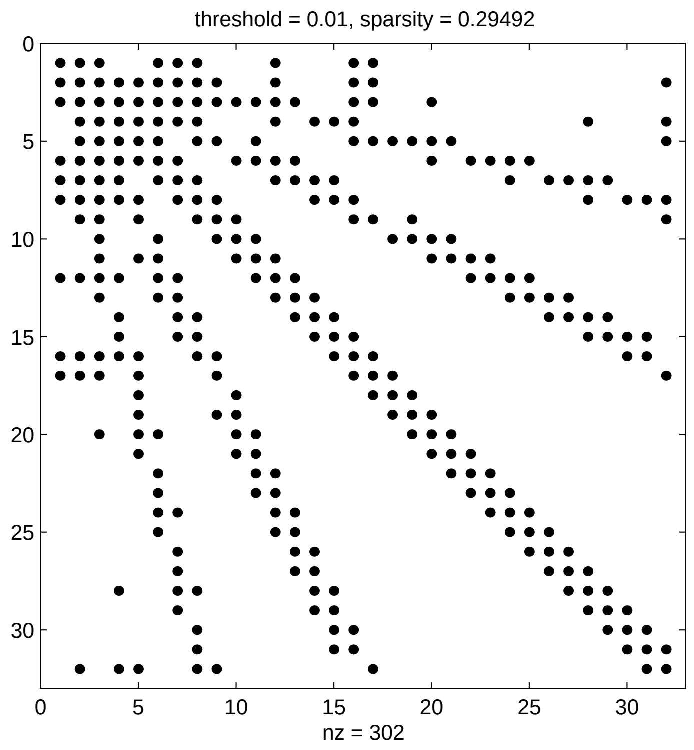


FIGURE 4.5. Sparsity pattern for the stiffness matrix (Fig. 4.4) compressed at threshold 10^{-2}.

class of *hypersingular* kernels [54] that presents a very difficult numerical problem to solve (quickly deteriorating matrix condition numbers result in slow convergence). Fortunately, wavelet treatment of hypersingular operators was proven to be as efficient as for other types of integral operators. We proceed with illustrating this with an example of hypersingular kernel that also arises from an electrostatic problem.

Consider an electrostatic problem kernel (Fig. 4.6):

$$G(x, x') = \ln\left(\frac{1}{|x - x'|}\right) \tag{4.22}$$

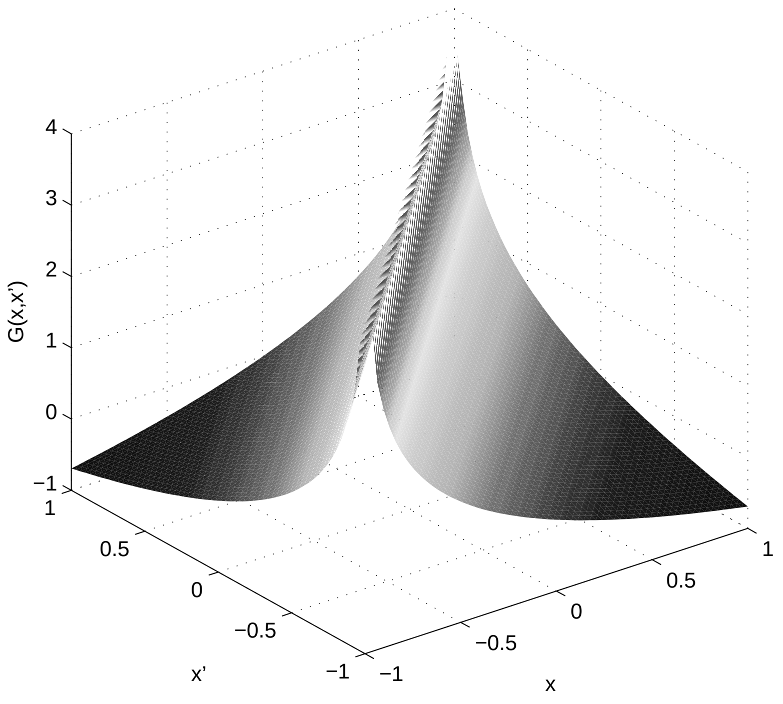


FIGURE 4.6. Hypersingular electrostatic kernel.

This kernel is a representative of class of hypersingular kernels that also corresponds to the electrostatic potential due to a charge filament in free space. Stiffness matrix obtained from Galerkin discretisation in Daubechies basis is shown in Fig. 4.7 and sparsity pattern of compressed matrix in Fig. 4.8. Comparing these results with the ones for the ordinary singular kernel (4.21) we confirm results published in [54] stating that hypersingular operators can be successfully discretised in wavelet bases with thresholding that leads to a sparse matrix. An important consequence of this is that wavelet algorithms can take full advantage of generalized Green's functions for capacitance extraction calculations in multilayered dielectrics.

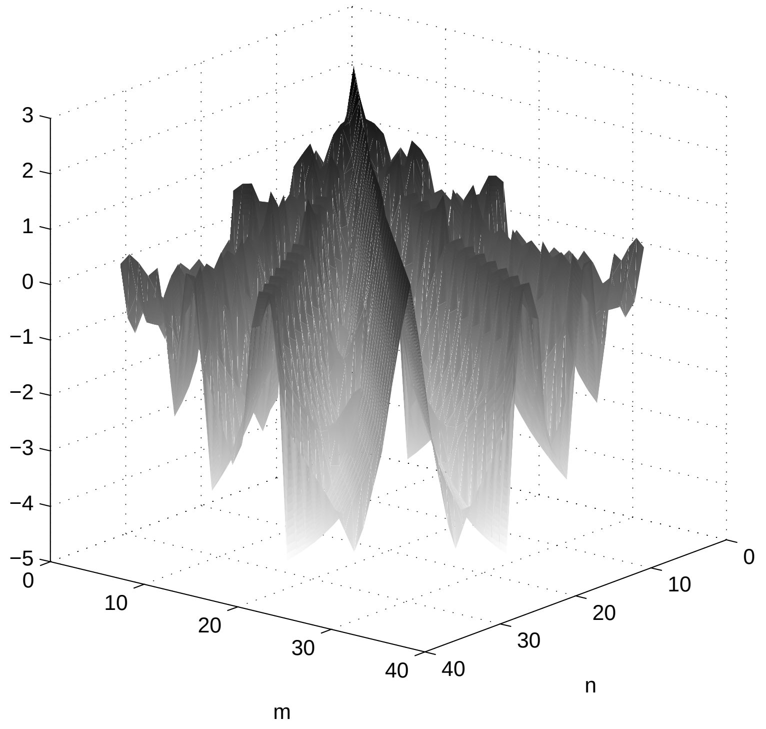


FIGURE 4.7. The stiffness matrix for the hypersingular kernel (Fig. 4.6) discretised in the Daubechies basis (Fig. 4.3). Note log scale on vertical axis.

The usual way (e.g. [67]) of thresholding stiffness matrix A is to choose a largest by absolute value element a_{max} and discard all the elements that are less than εa_{max} by absolute value. Fig. 4.8 shows an example of typical sparsity pattern obtained by means of such thresholding technique. Conventional approach to thresholding described above (we will call it *soft* thresholding) exhibits the same rate of convergence as the uncompressed system [54], but the computational gain from transition to sparse format is moderate. Despite the widespread claims in mathematical literature

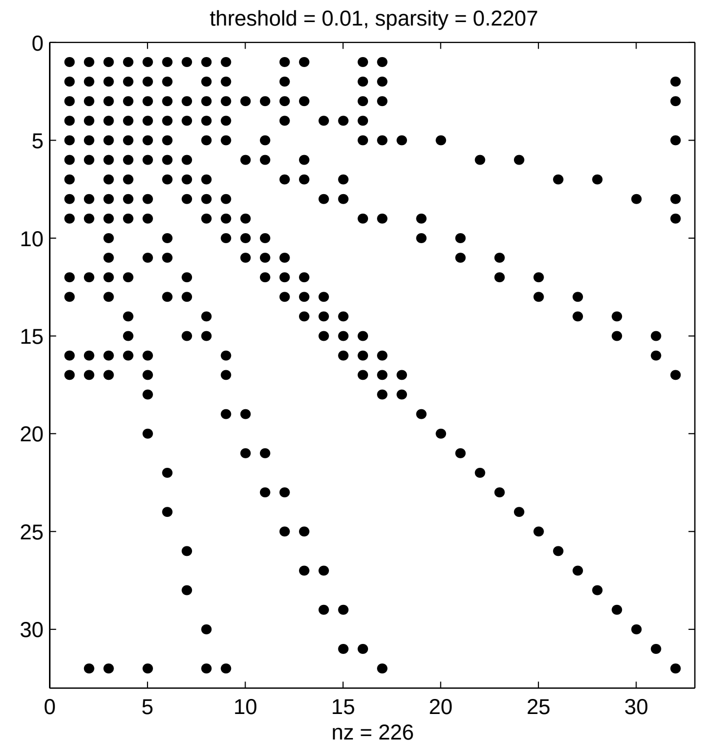


FIGURE 4.8. Sparsity pattern arising from the stiffness matrix for the hypersingular kernel (Fig. 4.7). Compressed at threshold 10^{-2}.

that soft thresholding produces an O(N) sparse matrix, Wagner and Chew showed in [53] that for a wide range of scattering EM problems sparsity of the stiffness matrix obtained with pulse expansion and subsequent wavelet transform and thresholding is $O(\beta N^2)$, where β is a very weak function of the problem size.

Refer to Fig. 4.9. Six plots in Figs. 4.9a, 4.9b and 4.9c represent sparsity patterns (left column) and accuracy vs computational cost (right column) plots. Accuracy is represented as absolute value (in percent) of error for integral of approximate solution (2.25) plotted versus computational cost (in flops) required to obtain such accuracy at increasing orders of expansion. Integral equation with kernel (4.22) has an analytical solution, that makes error estimation a straightforward and accurate process. Fig. 4.9a and 4.9b contain data for soft thresholding with parameter $\varepsilon = 10^{-3}$ and $\varepsilon = 10^{-2}$ correspondingly. One can see that for $\varepsilon = 10^{-3}$ computational gain is practically non-existent and barely reaches an order of magnitude for $\varepsilon = 10^{-2}$. Accuracy stays within several percent which is suitable for our application. With further increase of thresholding parameter, compressed matrix often becomes singular and requires special treatment.

In order to dramatically reduce computational cost, we apply aggressive thresholding technique (we call it *hard* thresholding) with the goal to obtain maximum sparsity with guaranteed absence of singularity in compressed stiffness matrix (Fig. 4.9c). To obtain this, we discard all the elements in stiffness matrix with absolute value less than threshold computed as

$$\gamma \min\{|a_{nn}|\} \tag{4.23}$$

where $0 < \gamma \le 1$ is a relaxation coefficient (typically is equal to 1) that gives an extra degree of freedom in handling stiff systems. Hard thresholding does not necessarily exhibit uniform convergence, but provides reasonable accuracy (better than 10%), bounded by the accuracy of uncompressed system, for the capacitance extraction. Resulting matrix is extremely sparse and almost diagonal that allows to achieve huge computational gain (2 to 3 orders of magnitude) with very little respect to the

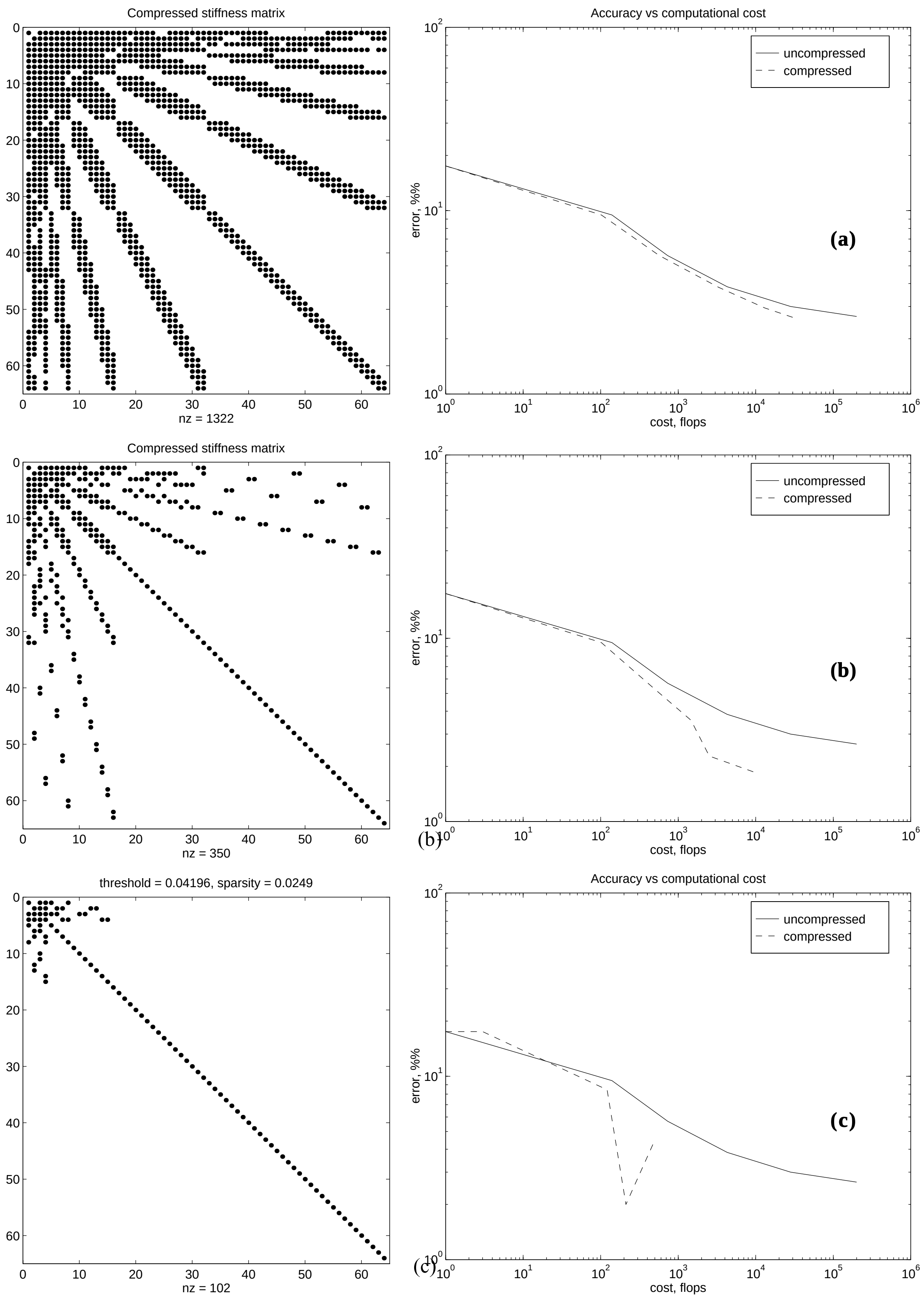


FIGURE 4.9. Sparsity patterns (left) and convergence (right: note log scale) for different thresholding techniques. (a) Soft thresholding. (b) Soft thresholding. (c) Hard thresholding.

order of expansion. Should convergence of solution for hard thresholded matrix prove to be unsatisfactory, relaxation coefficient γ can be lowered at any time, new nonzero elements introduced into matrix and another iteration performed. Note, that results of a previous thresholding iteration are completely reused here and we can start with sparse matrix represented in Fig. 4.9c and gradually move back to the case of Fig. 4.9b terminating calculations as soon as necessary accuracy is obtained.

4.3.4 CALCULATING CHARGE DENSITY VERSUS CALCULATING CHARGE

The next important foundation of the proposed technique is the ability, by virtue of multiresolution analysis, to obtain per-unit-length charge values directly from wavelet expansion coefficients without performing inverse wavelet transform and integrating charge density.

When the capacitance extraction problem is approached via solution of integral equation (4.15), as it is in our case, superficial charge density σ has to be integrated over the surface of respectful conductors. This results in matrix of per-unit-length charges $[Q_{ij}]$:

$$Q_{ij} = \int_{\Omega_i} \sigma(r)|_{V_j = 1,\, V_{j'} = 0,\, j \neq j'}\, dr \qquad (4.24)$$

that is, of charges induced on *i*-th conductor by forcing unit potential on *j*-th conductor and zero potential on the others. Capacitance matrix is then obtained by inverting matrix of charges:

$$C = Q^{-1} \tag{4.25}$$

Consider approximation of σ in wavelet basis (4.17):

$$\hat{\sigma} \cong \mathrm{Proj}_{V_0}\sigma + \sum_{j=1}^{J} \mathrm{Proj}_{W_j}\sigma \tag{4.26}$$

Together with (2.68), (4.16) and orthonormality of the wavelet basis it immediately follows that

$$Q_{ij} = \int_{\Omega_i} \hat{\sigma}(r)\,dr = \alpha_{0,0;i}\int_{\Omega_i}\varphi_{0,0;i}(r)dr + \sum_{j=-1}^{J}\sum_{(k)}\alpha_{j,k;i}\int_{\Omega_i}\psi_{j,k;i}(r)dr = \alpha_{0,0;i}. \tag{4.27}$$

In other words, *all the wavelet coefficients at scales higher than 0 contain no information about total charge on a conductor*[1]. It means that, for the purpose of capacitance extraction, we only need to compute scaling function coefficients with required accuracy, while the wavelet function coefficients are necessary to satisfy equation (2.33) just *in energetic sense*. In

1. We should mention here that wavelet bases are obviously not the only ones that provide separation of average value from the rest of the solution. Another example of such basis is the complete Fourier basis on $[0, 2\pi]$. This basis, however, does not form multiresolution analysis and does not have vanishing moments, i.e. kernels discretised in this basis give rise to essentially dense matrix. In some cases, some kind of thresholding can be successfully applied to such matrix, but in general we can not expect that it would always be the case.

fact, hard thresholding considered above is a direct consequence of this observation: since we are not interested in particular charge density distribution, we can be more aggressive in thresholding those matrix entries that correspond to wavelet functions with zero average. The number of scaling function coefficients for which we are looking for is significantly less than total order of expansion and is comparable to the number of elementary conductors. This is one of the key points of the proposed approach.

4.3.5 The iterative wavelet algorithm for capacitance extraction

We propose the following algorithm for efficient capacitance extraction in wavelet domain.

1. Choose wavelet basis.

2. Choose highest level of expansion J and compute main diagonal of the stiffness matrix.

3. Compute first approximation of solution for strictly diagonal matrix. Set thresholding relaxation coefficient $\gamma = 1$.

4. Compute threshold as per (4.23).

5. Set j=1.

6. If we already have matrix elements computed at this level, go to 8.

7. Compute matrix coefficients at level *j*. Threshold a priori insignificant elements while computing. Store the rest of coefficients for future use.

8. Threshold matrix elements at level *j* with γ.

9. Solve the new matrix equation.

10. Compute capacitance matrix.

11. Estimate convergence of solution. Stop if satisfactory.

12. Proceed to the next scale of expansion: $j=j+1$.

13. If $j<J$, go to 7.

14. Set lower value for γ.

15. Go to 4.

4.3.6 COMPUTATIONAL COMPLEXITY AND IMPLEMENTATION ISSUES

As seen from the previous sections, intelligent choice of wavelet basis for capacitance extraction would imply expectations of significant stiffness matrix coefficients corresponding to scaling functions and linking scaling functions between each other as well as with wavelet functions. All the other coefficients should be as small as possible in order to obtain better sparsity after compression.

Following considerations for wavelet properties should be taken into account while choosing expansion basis:

- local support;
- orthogonality;
- highest number of vanishing moments for a wavelet function with given support width;
- non-vanishing moments for scaling function.

Daubechies wavelets satisfy all of these requirements and as so make up an optimal basis for the capacitance extraction.

As we have mentioned before, decreasing support width and increasing number of vanishing moments give rise to better compression of stiffness matrix. This is a general rule of the thumb that might impose certain difficulties while choosing particular order of Daubechies wavelets. Recall [5], that Daubechies wavelets of order N have support width of $2N$-1, N vanishing moments for wavelet function and no vanishing moments for scaling function. One can not simultaneously decrease support width and increase number of vanishing moments. In fact, it was proven that this particular wavelet has the highest number of vanishing moments for a given support width. Fortunately, hard thresholding that we use for compression of stiffness matrix is not so sensitive to the order of Daubechies wavelets (Fig. 4.10). For the purpose of capacitance matrix computations, simple order 1 Daubechies wavelets, also known as Haar wavelets, provide similar convergence as the higher order wavelets (see Table 4.1).

TABLE 4.1. Comparison of convergence for thresholded matrices obtained with different wavelets.

	matrix Fig. 4.10 (left) obtained with Haar wavelets				matrix Fig. 4.10 (right) obtained with Daubechies order 5 wavelets			
γ[a]	nz[b]	matrix condition number	charge density RMS error[c]	PUL charge error[d]	nz[b]	matrix condition number	charge density RMS error[c]	PUL charge error[d]
0.003	2,232	60.5278	5.6%	3.4%	1,560	46.4895	5.1%	3.3%
0.001	1,640	62.1117	5.8%	3.3%	1,212	46.5984	5.4%	3.3%
0.03	1,060	63.5899	8.3%	3.3%	778	46.2896	5.9%	3.3%
0.1	616	50.7706	25%	3.0%	334	41.2817	12%	3.6%
0.3	196	31.2428	33%	2.6%	160	30.4702	29%	3.7%
1.0	102	28.5450	45%	4.8%	86	28.2838	39%	6.2%

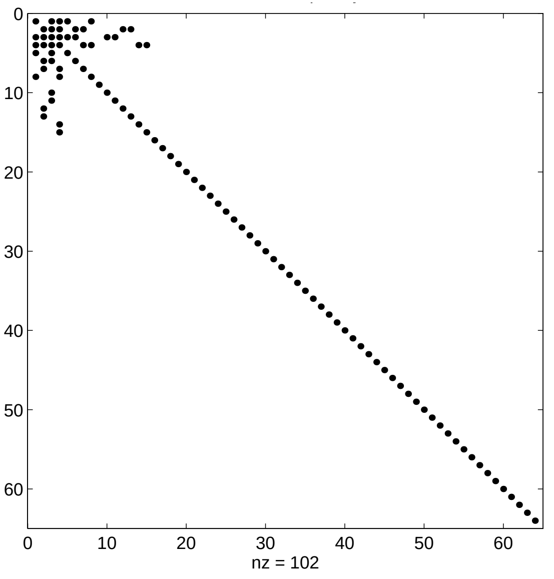
a. Relaxation coefficient for hard thresholding.

b. Number of nonzero entries in 64x64 sparse matrix.

c. Calculated as the ratio of Euclidian norm of difference between numerically computed charge density and closed form solution for charge density related to the Euclidian norm of the latter.

d. Calculated as the error of numerically computed per-unit-length charge related to the closed form solution.

Solving equation (4.15) for the integral of σ rather than for charge density itself also relaxes convergence criteria. Dahmen *et. al.* showed in [54] that a matrix equation arising from discretisation of (hyper)singular integral equations in wavelet bases can exhibit the same speed of convergence with matrix compressed to O(N) nonzero entries as the original uncompressed matrix provided that *proper wavelet basis is chosen*. This essentially means that, in order to achieve O(N) computational costs, a special



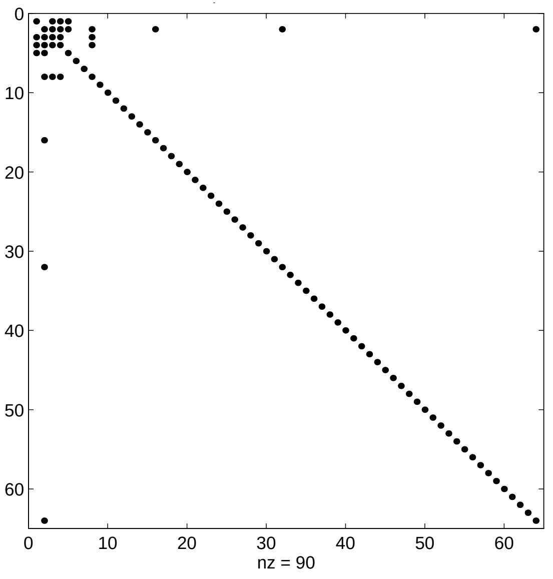


FIGURE 4.10. Side-by-side comparison of sparsity patterns for the compressed stiffness matrix obtained from Haar (left) and Daubechies order 5 (right) wavelets; hard thresholding in both cases.

wavelet family should be constructed for each particular kernel. This is true for convergence criteria based on Euclidian norm of the solution. However, in our case convergence criteria can be based on a much more relaxed error condition for the integral of solution, which also relaxes requirements for the choice of basis. For the purpose of capacitance extraction, there is no need to construct special wavelet family for each type of Green's function as ordinary Daubechies wavelets appear to do the job.

This relaxation of convergence criteria allows to obtain good approximation of the total charge on conductor without accurately computing surface charge density. An illustration of this phenomena can be found on Fig. 4.11, which plots normalized surface charge density as a solution of uncompressed matrix, compressed with hard thresholding and closed form expression. One can see that while compressed solution approximates surface charge density in a rather erratic manner (RMS error 36%), total charge on the conductor is approximated within 0.64% from the closed form solution. The uncompressed solution gives good approximation for both charge density and total charge, but this comes at significantly higher computational cost of solving a full matrix.

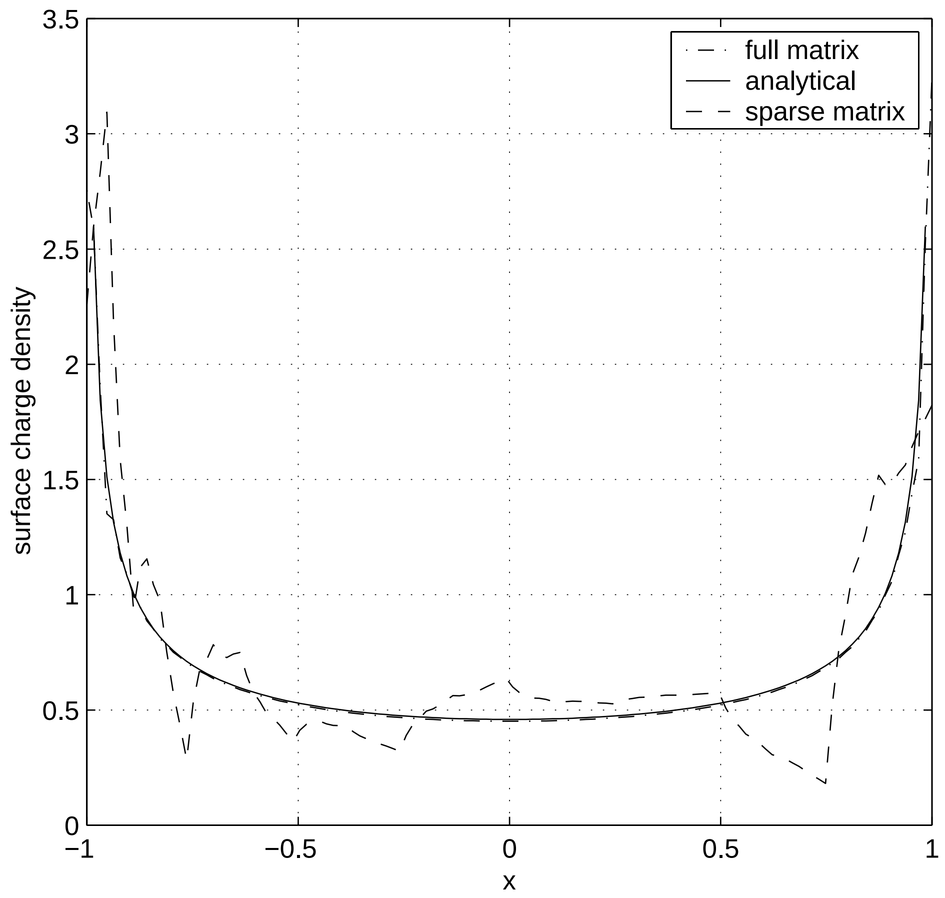


FIGURE 4.11. Normalized surface charge density on a microstrip approximated with a solution of the uncompressed and compressed system.

Now let us consider the implementation issues. At Step 9 of the algorithm, one may use any kind of general purpose matrix solver. However, the proposed method will most benefit from custom solvers. Using general purpose solvers leads to an observation that, with aggressive thresholding, combined cost of matrix preordering (minimum degree), LU decomposition and forward/backward substitution is still less than the cost of just a single iteration for generalized minimum residue method. In fact, one can see (Fig. 4.10) that sparsity pattern of the Galerkin-wavelet matrices is very specific, which makes preordering nearly trivial. This might not always be the case, however, for extremely large problems, where iterative techniques may gain advantage. A custom iterative solver optimized for minimization of residue in terms of capacitance matrix, not just the Euclidian norm of solution vector, should further improve efficiency of large scale capacitance extraction. This statement is based on the fact that a good initial guess for main diagonal of capacitance matrix can be made at the expense of O(N) operations and even better guess for the whole capacitance matrix at the expense of O(L^3). Together with possibility of effective preconditioning [14] resulting in matrix condition number (and, consequently, number of iterations) to be independent of N, this provides potential for a very fast matrix solver. Similarly to this, a custom preordering technique will be highly beneficial for LU/FBS approach. This technique should take into account block structure of the sparse stiffness matrix (Fig. 4.12) as well as very special sparsity pattern of each block (Fig. 4.10).

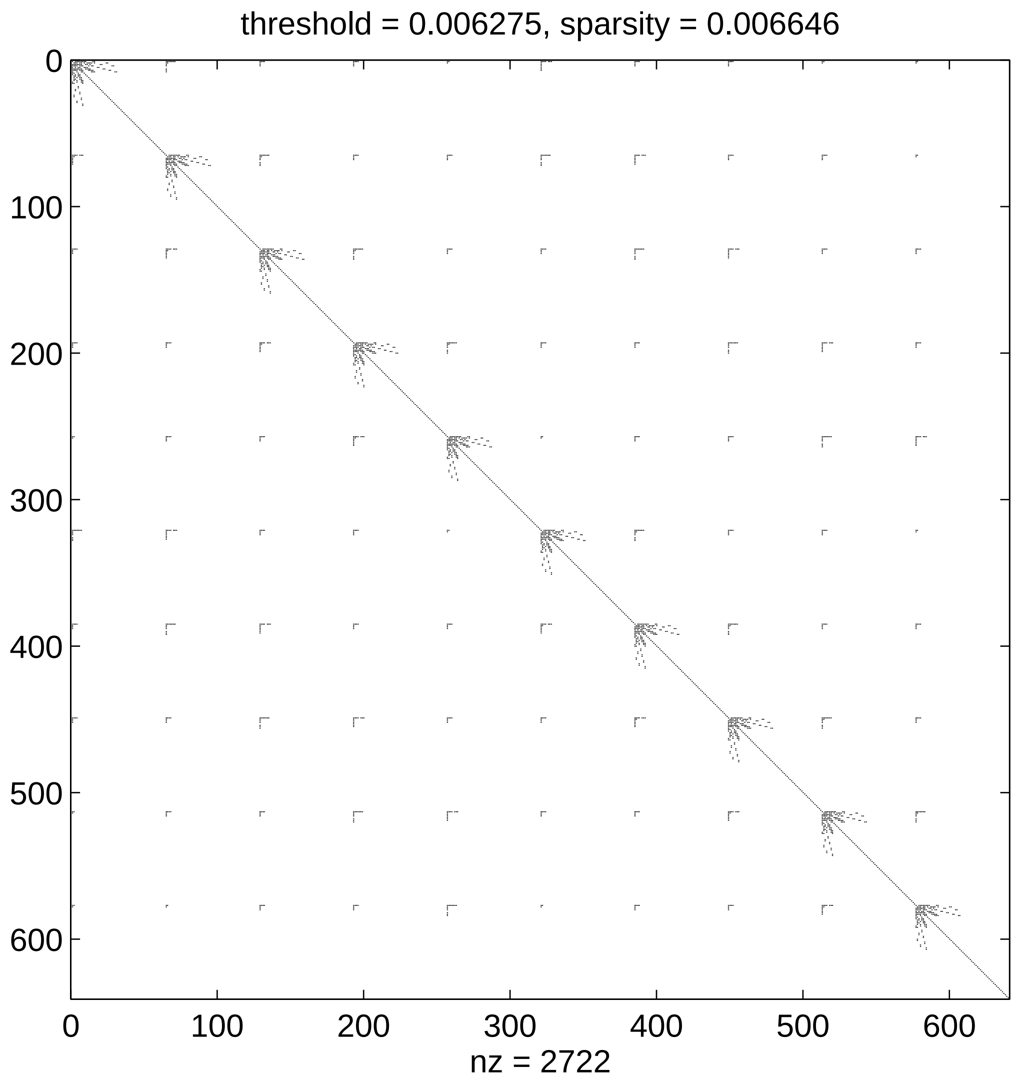


FIGURE 4.12. Typical sparsity pattern for the stiffness matrix obtained from Wavelet-Galerkin expansion: 10 microstrips, 64 Haar wavelets per microstrip. 0.2-relaxed hard thresholding.

4.3.7 NUMERICAL RESULTS

We proceed with experimental study of convergence speed of the proposed technique on a moderately sized examples. For the sake of convenience, these examples are limited to uniform multiconductor transmission lines in multilayered dielectric media. Comprehensive discussion of applicable technique for handling conductors of arbitrary shapes [67] was done before and are not the key point here. We do not provide side-by-side comparisons with other extraction software as implementation issues can significantly affect required CPU time.

All numerical experiments presented in Section 4.3 were performed on a Sun SPARCstation with 143 MHz UltraSPARC CPU, 0.5 MB of L2 cache and 128 MB of 71.5 MHz physical RAM. The equivalent CPU speed of this workstation was found to be approximately 19 Mflops (see Section A.2 on page 158).

First sample configuration incorporates 10 infinitely thin microstrips arranged on 2 levels and embedded in a dielectric slab shielded from all sides (Fig. 4.13). 2D spatial Green's functions for this configuration were derived in [69, equations (53)-(56)]. For the reasons stated above, we use Galerkin expansion with Haar wavelet basis as the simplest one. Order of expansion is increased in dyadic manner in the range of 4 to 64 wavelets per microstrip, such that another complete level of wavelets can be added. As such problems no longer have a closed-form solution, we are forced to use a numerical solution as a reference. This reference solution was obtained from a high-order Haar-Galerkin expansion (at 256 wavelets per microstrip) *without compression* of stiffness matrix and with Green's function evaluated at the same fine grid. Capacitance matrix computation error was estimated in two ways: error of matrix 1-norm and 2-norm. 1-norm error gives estimation of accuracy for the self and mutual capacitances, while the 2-norm error bounds accuracy of the largest eigenvalue of capacitance matrix. Finally, we made comparison against MoM method with traditional basis of pulse functions. As pulse functions also happen to be scaling functions for the Haar wavelets, the same procedure as described in section 4.3.5 was applied to generate MoM stiffness matrix with setting initial order of expansion more than the highest order of expansion (j=J-1 so that the algorithm would stop after generating a full set of scaling functions only) and thresholding parameter γ=0. A standard LU/FBS matrix solver (Gaussian elimination with partial pivoting) was used for compressed, uncompressed and reference solution (with non-symmetric minimum degree preordering in the sparse case).

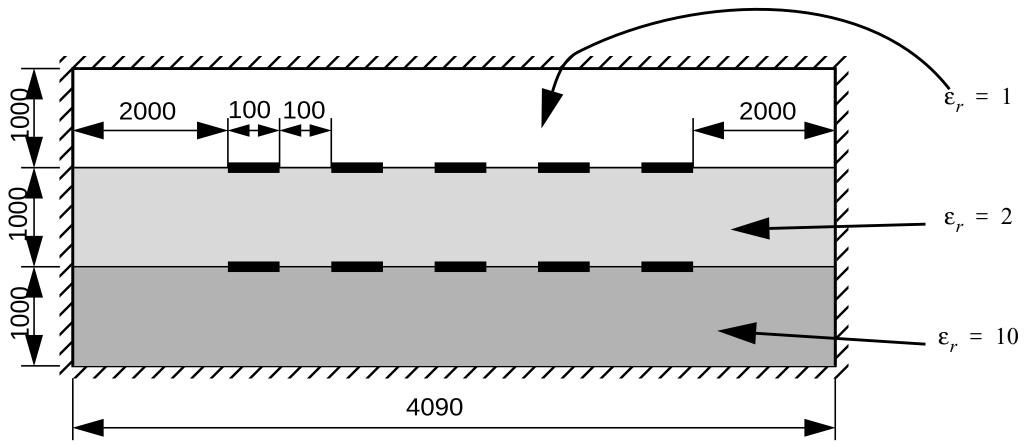


FIGURE 4.13. Configuration for Example 1. All dimensions are in micrometers.

Fig. 4.12 shows sparsity pattern obtained after compression of stiffness matrix using hard thresholding technique with relaxation coefficient $\gamma = 0.2$ at the highest order of resolution $N = 64 \times 10$. Note that the matrix has essentially block structure with each on-diagonal block representing a microstrip and each off-diagonal block representing coupling between microstrips. All the coefficients corresponding to scaling functions are retained, while the absolute majority of coefficients associated with wavelet functions are successfully discarded, resulting in sparsity ratio of better than 0.7% and good accuracy for the capacitance extraction. Condition number of the compressed matrix also was improved by approximately 2 times compared to uncompressed one. Initial guess for capacitance matrix obtained at a cost of $O(L^3)$ was within 30% error for any given element of the capacitance matrix. Storage requirements at expansion level providing 0.01% accuracy were 3.27 MB for the uncompressed matrix and only 43.5 kB for the sparse case, which makes a 75-fold improvement.

Fig. 4.14 portrays results of consecutive Haar-Galerkin computations at increasing levels of resolution. Computational cost is presented in terms of number of floating point operations. This results illustrate two notable conclusions: first, Wavelet-Galerkin method with appropriately relaxed hard thresholding results in capacitance matrix calculated with very reasonable accuracy and uniform convergence, and second, computational cost for these calculations remains *extremely low* - less than 10^6 flops, compared to the range of 10^7-10^9 flops for traditional basis with the same level of accuracy. That makes computational gain of up to 3 orders of magnitude for even a moderately sized problem. This example also illustrates essential equivalency of the error criteria based on matrix 1- and 2-norms.

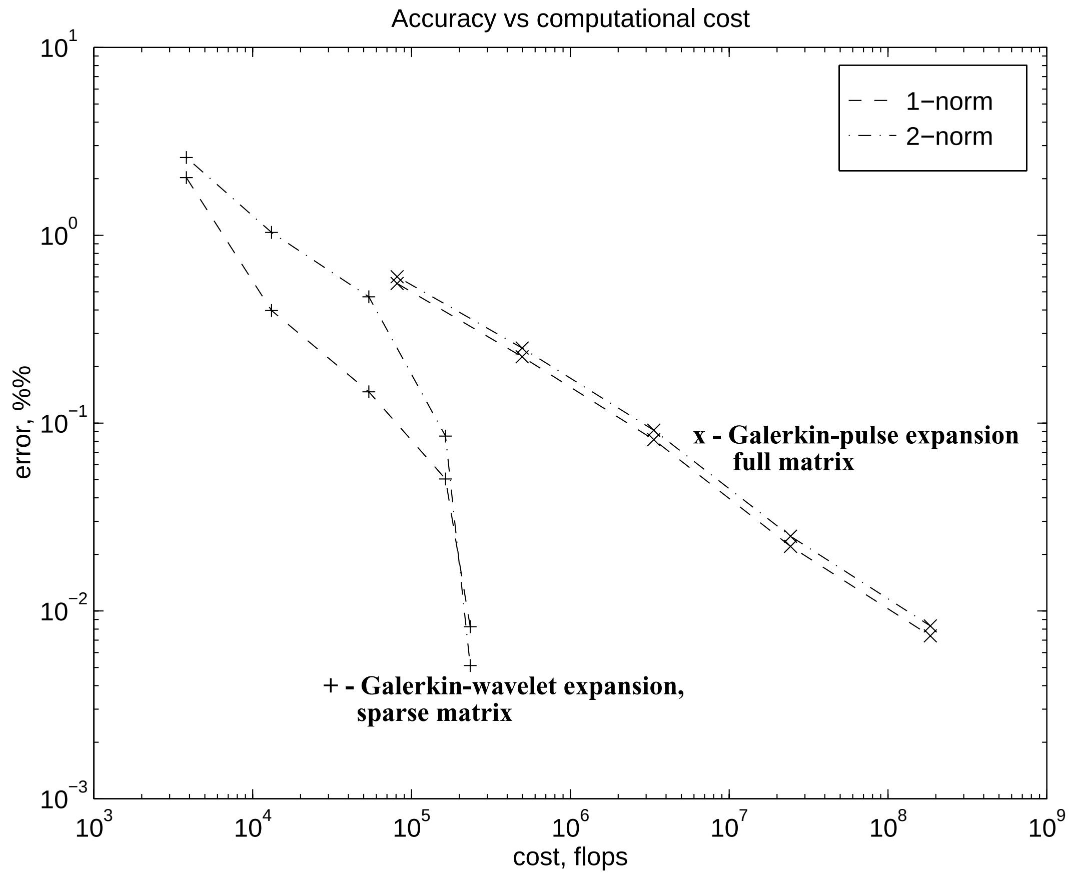


FIGURE 4.14. Capacitance matrix computation error versus computational cost: 10 microstrips, Haar wavelets, 4...64 wavelets per microstrip, 0.2-relaxed hard thresholding.

Second example (Table 4.2) involves progressively increasing number of conductors arranged in the same manner as above (Fig. 4.13) with fixed separation between the conductors. Width of the structure was adjusted accordingly to keep 2 mm separation between the side walls and conductors. Expansion order was kept at 64 basis functions per conductor. Expansion was performed with order 5 Daubechies wavelets. A Gaussian elimination with partial pivoting matrix solver was used for both dense and sparse matrices with nonsymmetric minimum degree preordering in the sparse case. Hard thresholding with $\gamma = 1$ was used as matrix compression scheme. The goal of this experiment was to compare convergence of uncompressed and compressed systems and verify that compressed system converges with the same speed as uncompressed one. The last column lists compression error estimated in Euclidian matrix norm of capacitance matrix. This data shows that compression error not only

TABLE 4.2. Convergence comparison of uncompressed and compressed matrix.

		uncompressed matrix			compressed matrix				
L[a]	N[b]	CPU time, s	Mflops count	memory, Mbytes	CPU time, s[d]	Mflops count	memory, Mbytes	nz[e]	error[c]
10	640	10.1	176.5	3.12	0.02	0.008	0.014	906	1.3%
15	960	34.1	593.5	7.03	0.04	0.017	0.022	1436	0.8%
20	1280	88.6	1,404.2	12.5	0.06	0.035	0.031	2016	0.5%
25	1600	159.9	2,739.3	19.5	0.11	0.111	0.040	2646	0.4%
30	1920	288.9	4,729.8	28.1	0.13	0.088	0.051	3326	0.35%
35	2240	449.9	7,506.5	38.3	0.16	0.269	0.062	4056	0.29%
40	2560	789.1	11,200.0	50.0	0.24	0.185	0.073	4836	0.25%
45	2880	975.3	15,942.0	63.3	0.27	0.529	0.086	5666	0.22%
50	3200	1,426.6	21,863.0	78.2	0.30	0.334	0.099	6546	0.19%
55	3520	1,844.4	29,093.0	94.5	0.45	0.919	0.114	7476	0.17%
60	3840	2,561.0	37,764.0	112.5	0.42	0.548	0.129	8456	0.15%

a. Number of conductors.

b. Matrix size.

c. Estimated as $\|C - C_c\| / \|C\|$, where C is obtained from solution of the uncompressed system and C_c obtained from solution of the compressed one.
d. Accurate to the CPU time slice.
e. Number of nonzero entries in sparse matrix representation.

remains bounded, but slightly decreases with increased number of conductors, which suggests that for large scale problems even more aggressive thresholding can be used approaching computational costs to the theoretical limit of O(N). The actual computational cost in terms of the number of nonzero elements in the stiffness matrix together with the asymptotic slopes is also plotted in Fig. 4.15 and it is quite reasonable to conclude that computational cost is much closer to O(N) than to O(N^2).

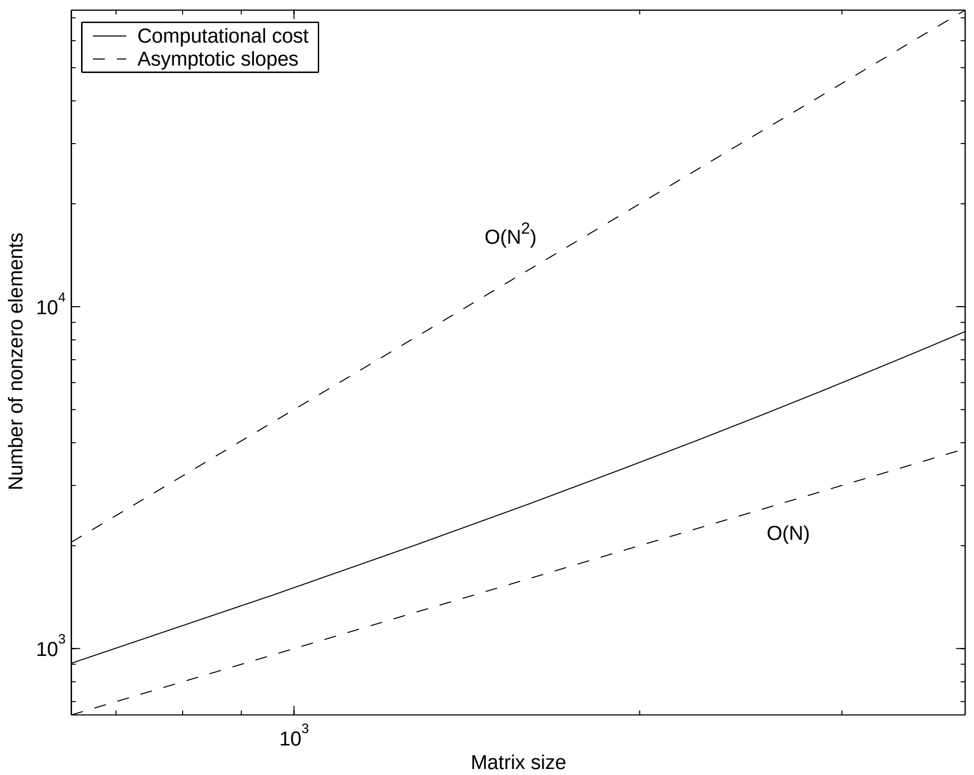


FIGURE 4.15. Computational cost for example 2.

Third example compares proposed technique with the one previously published in the literature [67]. This example represents multiconductor transmission line with thick substrate where the transmission lines are located high above the ground plane in comparison with their cross-section and separation between conductors (Fig. 4.16). Generalized Green's

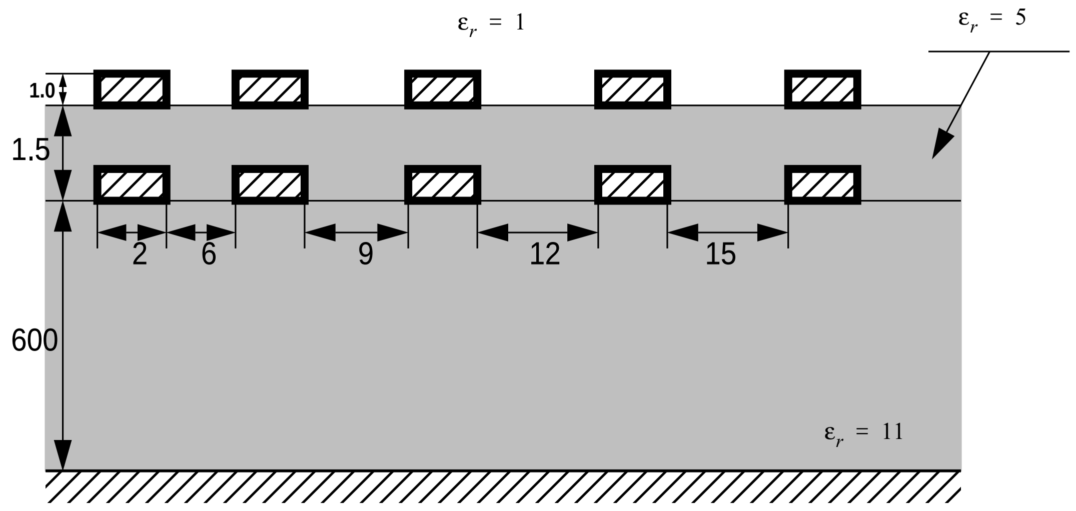


FIGURE 4.16. Configuration for Example 3. All dimensions are in micrometers.

function for this configuration were derived in [68, equation 16b] and used here with nine exponentials in approximation. The proposed technique was formulated with 64 Daubechies order 5 wavelets per each conductor and hard thresholding with $\gamma = 1$. The hybrid method of [67] was formulated as previously published, i.e. with 16 wavelets per conductor plus 256 wavelets at the dielectric interfaces and soft thresholding at $\varepsilon = 10^{-3}$. Traditional BEM solution as quoted in [67], Table IIIb, was taken as a reference to calculate Euclidian norm error of the capacitance matrix. Results of comparison are summarized in Table 4.3.

TABLE 4.3. Comparison of the proposed technique with previously published in the literature.

	Proposed technique	**Method from [67]**
matrix size	640 x 640	416 x 416
nonzero entries	2200	9604
CPU time	0.05 s	0.29 s
Mflops	0.13	0.53
memory requirements	0.0336 Mb	0.146 Mb
error	0.28%	1.69%

This comparison illustrates the idea that with the proposed method we should not be afraid of choosing high order of expansion from the beginning. Higher order of expansion will be compensated by more aggressive thresholding and incorporation of multilayered dielectric interfaces into the kernel. Error estimation does not necessarily mean that the proposed technique produced a more accurate solution, as the reference solution itself could be somewhat inaccurate too. Still, this gives the confidence that the proposed technique provides accuracy equivalent to the other two methods, being significantly more efficient.

4.3.8 SUMMARY OF THE CAPACITANCE EXTRACTION TECHNIQUE.

This section presented a new approach based on using periodic orthogonal wavelets as the basis functions for expansion of integral equation into linear matrix equation for superficial charge density and compression of stiffness matrix into sparse format using hard thresholding. The principal benefits of the new technique are:

- taking full advantage of multiresolution analysis that allows projection of superficial charge density onto subspaces spanned by scaling functions and directly gives total charge on a conductor without obtaining an accurate solution for the charge density;
- as a consequence of the above, we have applied an extremely aggressive thresholding algorithm that compresses stiffness matrix to almost diagonal sparse form;
- proposed capacitance extraction algorithm is kernel independent and has computational cost approaching O(N), versus O($N\log N$) for best available techniques and O(N^3) for direct methods;
- the algorithm also provides a flexible and efficient way of trading numerical accuracy for speed of computations.

This technique can produce a major impact on the computational cost of large scale physical interconnect parameter extraction and other problems involving computation of integral of the solution of integral equations with singular kernels.

4.4 CONCLUSIONS

In this chapter, we have considered applications of wavelets to the transient analysis of nonlinear circuits and analysis of distributed circuits.

Transient analysis in wavelet domain can provide accuracy comparable with the time marching methods, however so far it has to be shown that it can compete with time marching methods in terms of computational efficiency.

Macromodelling ot multiconductor transmission lines has shown definitive potential, particularly for highly nonuniform lines. However, proof of passivity has to be provided before wavelet macromodels can be used in EDA tools.

Wavelet methods for capacitance extraction are the most promising of the applications considered in this chapter. Initially, wavelets generated a lot of excitement in the EM community where integral equations always played an important role. By combining local support with vanishing moments, wavelets allowed stiffness matrix for such equations to be thresholded, thus providing sparse representation for otherwise structurally dense matrix. However, cost of construction of the stiffness matrix remains the bottleneck. Numerical evaluation of the inner products for stiffness matrix entries is often quite complicated and exceeds the cost of actual solution of the matrix. More research in this direction has to be done before wavelet methods become truly competitive. The issue of predictive thresholding also needs to be explored. Predictive thresholding [56] allows to determine locations of nonzero bands in compressed matrix *a priori* and compute only those matrix elements that will not be thresholded, instead of computing a full matrix and applying threshold afterwards.

5. SUMMARY AND FUTURE RESEARCH

In this thesis we have considered wavelet treatment of the several computationally challenging problems arising from the field of Electronics Design Automation. Between them, these problems cover three most important mathematical classes of problems in circuit analysis:

- boundary value problems described by nonlinear ODEs (steady state analysis)
- initial value problems described by nonlinear ODEs (transient analysis)
- boundary value problems described by linear and nonlinear PDEs (interconnects macromodelling)
- boundary value problems described by the linear integral equations (interconnects physical parameter extraction)

Steady-state analysis of nonlinear circuits is representative of the first class of computationally extensive problems: boundary value problems described by nonlinear ODEs. In Chapter 3, we present a new approach to the solution of this problem that takes advantage of wavelets. Following the traditional Harmonic Balance approach, we convert the set of coupled nonlinear ODEs and nonlinear algebraic equations to a set of purely nonlinear algebraic equations. Due to the essential local support of wavelets, the proposed approach results in a sparse representation for both the nonlinear and linear (discretization of differential operator) components of the Jacobian. This dramatically reduces computational cost of the analysis, particularly for multitone, large scale, highly nonlinear and broadband circuits. Some preliminary results presented in that chapter have been peer reviewed and accepted for publication [33], [34]. As this work appears to be the first attempt to apply wavelets to steady state analysis, the whole range of possibilities for future research is now wide open:

- Analysis of autonomous circuits. One can expect significant advantage in this area as wavelet method with trivial truncation provide full coverage of the frequency interval, thus effectively being ready to capture oscillations at any frequency inside the interval.
- Time domain adaptive methods. Box and diamond truncation schemes used in Harmonic Balance allow reduction of the analysis grid in frequency domain. Proposed method uses wavelet bases that are poorly localized in frequency domain, but it's the poor frequency domain localization that allows to obtain Jacobians without dense blocks. To further accelerate wavelet methods, time domain adaptive schemes can be devised, similar to those that are proposed for transient analysis in [39]. Wavelet analysis provides a systematic framework for such adaptive methods that solely rely on orthogonal transforms, thus preserving convergence and stability of the non-adaptive methods, unlike Fourier

methods that lose orthogonality when time domain adaptive schemes are introduced.

- Frequency domain adaptive methods. It may be advantageous to explore expansion bases arising from either complete wavelet decomposition tree (Fig. 2.15) or wavelet packet trees ([6], [7]). Wavelet tree gives dyadic resolution in frequency domain, while wavelet packet trees give arbitrary resolution in frequency domain, so they provide a systematic framework for approximating systems in frequency domain with arbitrary resolution. On the other hand, improvements in frequency domain resolution lead to denser transform matrices (Fourier basis is the ultimate in frequency resolution, but it's transform matrices are dense). Denser transform matrices lead to denser Jacobian. An optimum may exist somewhere down the line, but this direction needs a lot of exploration.
- Matrix solver. As it was shown, Jacobian matrices arising from wavelet expansion are essentially bandlimited. Matrix solver customized for this sparsity pattern can accelerate the method 2-4 times (see estimations on page 85). Also, iterative solvers have to be explored.
- Effects of using different wavelet families. Formulation presented in this thesis is general enough to be directly applicable to other orthogonal and biorthogonal wavelet families. Experimental work have been only done with Daubechies family because it was deemed to provide maximum sparsity in the Jacobian (for a given number of vanishing

moments Daubechies wavelets have minimum support). However, this matter needs to be explored further.

Transient analysis is representative of the class of initial value problems described by nonlinear ODEs. In this area time marching numerical integration methods ave extremely well established and although wavelet expansion show some promise (particularly in the view of O(h^4) convergence), a lot more research is necessary into perfection of adaptive schemes before wavelet methods can compete with time marching schemes.

Interconnect macromodelling is representative of the class of boundary value problems described by linear and nonlinear PDEs. The main obstacle here is the lack of proof of passivity for wavelet expansion. Any researcher in this area should first attempt to prove passivity, most likely, based on the integrated congruence transform [42].

Capacitance extraction is representative of the class of deterministic boundary value problems described by a linear integral equation. In the most practically important case of multilayered dielectric media, this equation has hypersingular kernel which is extremely computationally intensive to solve. Extension of the previously described wavelet approach potentially offers significant advantages over traditional methods in terms of both computational cost and ability to trade accuracy for speed. These results have been peer reviewed and published in an extensive paper [71].

Directions for future research must include development of fast techniques for computing the elements of the stiffness matrix (from arbitrary kernels) and further research into predictive thresholding [56]. Other directions may include development of a custom matrix solver briefly discussed in Section 4.3.6 and study of applicability of other known acceleration techniques to the wavelet expansion matrix.

APPENDIX A.

CALIBRATION OF CPU PERFORMANCE

A.1 THE BENCHMARK

For a lack of better choice, LU factorization of large dense matrices was chosen as a benchmark for estimating the floating point performance of target hardware/software platform. Factorization of dense matrices provides exact count of flops, which is difficult to estimate for sparse matrices. However, it should be noted that sparse matrix performance will be slower because of the overhead associated with sparse storage and manipulations with sparse data structures.

The equivalent CPU performance is estimated by benchmarking of a series of LU/FBS solutions of random square matrix equations of increasing dimension *n*. For small matrix dimensions, execution time was averaged over several runs in order to smooth out the influence of the discrete system clock. It is imperative to perform the measurements on large matrices which storage requirements significantly exceed the size of L2 cache.

Matlab code used for benchmarking is given below.

```
function n = lubench(nrange, tmin)
% CPU benchmark on LU factors of large dense systems
%
% use:
% lubench(nrange, tmin)
% tmin = 10 by default
% example:
% eval('lubench(100:100:3000)','disp([datestr(now) '' '' lasterr]); diary off;')

if (nargin < 1)
      help lubench
      error('no arguments supplied')
elseif (nargin < 2)
      tmin = 10
end;

diary off
format compact
warning backtrace

program = 'lubench';
start_t = 0; t = 0; raw_t = 0; % allocate variables
machinename = getenv('HOST')
diaryfilename = [program '.' machinename '.' num2str(now) '.diary']
diary(diaryfilename)

[s,w]=unix('fpversion');
disp(w);
disp(['matlab version: ' version]);
disp([datestr(now) ' starting ' program]);
diary off; diary on;% flush diary output

disp('======================================================================')
disp('time        nruns        n      memory  megaflops     avg CPU    raw time')
disp('======================================================================')
for n = nrange;
   lasterr('');
   A = randn(n,n); b = randn(n,1); x = randn(n,1);
   [x, t, raw_t, elflops, memory, nruns] = lusolve(A, b, x, tmin);
   mflops = elflops / t / 1.e6;
   s = sprintf('%s %8d %8d %10.1fk %10.1f %10.4fs %10.2fs', ...
       datestr(now,13), ...
       nruns, ...
       n, ...
       memory/1024, ...
       mflops, ...
       t, ...
       raw_t);
   disp(s);
   diary off; diary on;
   clear A b x
end
disp('======================================================================')
disp([datestr(now) ' ' program ' completed'])
diary off
return

% ======================================================================
function [x, t, raw_t, elflops, memory, nruns] = lusolve(A, b, x, tmin)

[n, m] = size(A);

% oldflops = flops;
t = 0; raw_t = 0; nruns = 0;
while t < tmin
   nruns = nruns + 1;
   tic; start_t = cputime;
   x = A\b;
   t = t + cputime - start_t; raw_t = raw_t+toc;
end;   % while
elflops = (2/3*n^3 + 2*n^2);
t = t / nruns;
% elflops = flops - oldflops;
% memory = 8 * ( prod(size(A)) + prod(size(b)) + prod(size(x)) );
dummy = whos; memory = sum([dummy.bytes]);

return;
```

A.2 Calibration results for Section 4.3.

All numerical experiments presented in Chapter 4.3 were performed on a Sun SPARCstation with 143 MHz UltraSPARC CPU, 0.5 MB of L2 cache and 128 MB of 71.5 MHz physical RAM. The equivalent CPU speed of this workstation was found to be approximately 19 Mflops/s.

```
matlab version: 5.3.1.29215a (R11.1)
=========================================================================
time        nruns        n      memory  megaflops     avg CPU    raw time
=========================================================================
16:39:02      140      100       79.7k       19.2     0.0359s       5.33s
16:39:08       46      150      178.1k       21.1     0.1089s       5.11s
16:39:13       21      200      315.6k       22.2     0.2433s       5.11s
16:39:18       11      250      492.2k       22.7     0.4645s       5.10s
16:39:23        6      300      707.8k       21.7     0.8383s       5.04s
16:39:29        4      350      962.5k       20.4     1.4100s       5.65s
16:39:36        3      400     1256.2k       19.8     2.1733s       6.56s
16:39:42        2      450     1589.1k       19.1     3.1950s       6.40s
16:39:51        2      500     1960.9k       20.7     4.0500s       8.90s
16:39:59        1      600     2821.9k       19.9     7.2600s       7.68s
16:40:12        1      700     3839.1k       18.8    12.2000s      12.25s
16:40:31        1      800     5012.5k       18.6    18.4100s      18.69s
16:40:57        1      900     6342.2k       18.6    26.2400s      26.37s
16:41:34        1     1000     7828.1k       19.1    35.0200s      36.48s
=========================================================================
```

A.3 Calibration results for Chapter 3.

All numerical experiments presented in Chapter 3 were performed in Matlab 6.5.0 (R13), running on a SUN Blade-1000 workstation with 900 MHz UltraSPARC-III CPU, 8 MB L2 cache and 5 GB of physical RAM. The equivalent CPU speed of this workstation was found to be between 600 and 800 Mflops/s.

```
matlab version: 6.5.0.180913a (R13)
=========================================================================
time         nruns         n      memory  megaflops     avg CPU    raw time
=========================================================================
16:33:25        92      1000     7828.2k      613.0     1.0909s     104.12s
16:35:10        12      2000    31281.3k      623.7     8.5642s     105.31s
16:37:02         4      3000    70359.4k      673.6    26.7500s     110.44s
16:39:08         2      4000   125062.6k      706.8    60.4150s     124.28s
16:41:09         1      5000   195390.7k      725.7   114.9000s     118.10s
16:44:33         1      6000   281343.8k      745.5   193.2600s     199.21s
16:49:50         1      7000   382921.9k      755.6   302.7400s     311.50s
16:57:44         1      8000   500125.1k      755.0   452.2600s     466.24s
17:08:47         1      9000   632953.2k      768.8   632.3800s     653.27s
17:23:37         1     10000   781406.3k      782.6   852.1600s     878.18s
=========================================================================
```

APPENDIX B. COMPUTING THE CONNECTION COEFFICIENTS

Matlab code for computing connection coefficients for (bi)orthogonal local support wavelets with vanishing moments (see section Section 3.2.3 on page 54).

```
function [r, a, C] = cc(M, L)
% [r, a, C] = cc(M, L)
% connection coeffs for local support wavelets with vanishing moments
%
% M (positive integer >= 2) number of vanishing moments (including zero
%       average, so that for Haar: M=1, Daub2: M=2, etc).
%
% L (positive integer >= 4) length of decomposition LPF
%
% r (column symbolic vector of length L-2) connection coeffs:
%             +inf         d
%       r(l) = INT phi(x-l) -- phi(x) dx
%             -inf         dx
%       where phi(x) is decomposition scaling function generated by LPF with
%       filter coefficients h(1:L);
%       r(:) are accurate for h(:) normalized such that sum(h.^2) = 1;
%       generally, r(l) ~= 0 for (-L+2) <= l <= (L-2), -r(-l) = -r(l),
%       r(0) = 0;
%
% a (symbolic row vector of length L)  contains autocorrelation coefficients
%       of h(:):
%                L-n
%       a(n) = 2*SUM(h(i)*h(i+n)),      a(n) = 0 for even n;
%                i=1
%
% C is an intermediate result of computing a(:) symbolically:
%                                           2
%                              ((2 M - 1)!)
%                      C = -------------------------
%                                    2   (M - 1) 2
%                          ((M - 1)!)  (4       )
%
% Reference: G. Beylkin, On the Representation of Operators in Bases of
% Compactly Supported Wavelets, SIAM J. on Numerical Analysis, Vol.6, No.6,
% pp. 1716-1740.
%
% Requires: symbolic toolbox.
```

```
%
% v.1.0, nsoveiko@doe.carleton.ca
% Fri Jun  2 15:40:07 EDT 2000

syms C m positive
syms a h real

M = sym(M);     % number of vanishing moments
L = sym(L);     % filter support
m = [1:M];
a = sym(zeros(1,double(L)));

C = subs( simple( sym('((2*M-1)! / ((M-1)! * 4^(M-1)))^2') ) );

a(double(2*m-1)) = subs( simplify( ...
         sym('(-1)^(m-1) * C / ((M-m)! * (M+m-1)! * (2*m-1))')));

Lr = double(L-2);
B = sym(zeros(Lr+1,Lr));
for l = 1:(Lr)
        if (2*l <= (Lr)), B(l,2*l) = B(l,2*l) + 1; end;
        for k = 1:(double(L/2))
                j = 2*l-2*k+1;
               if (j <= Lr), B(l,abs(j)) = B(l,abs(j)) + sign(j)*a(2*k-1)/2; end
                j = 2*l+2*k-1;
               if (j <= Lr), B(l,abs(j)) = B(l,abs(j)) + sign(j)*a(2*k-1)/2; end
        end; %  for k = 1:(L/2)
end; %  for l = 1:(L-2)
B = 2*B;
B = eye(size(B)) - B;
l = Lr+1;
B(l,:) = 2*sym(1:Lr);
r = B \ sym([zeros(Lr,1); -1]);

return
```

REFERENCES

[1] Donald G. Dudley, *Mathematical foundations for Electromagnetic Theory*, New York: IEEE Press, 1994.

[2] R. F. Harrington, *Field Computation by Moment Methods*, New York: MacMillan, 1968.

[3] T.J. Thomas, *Numerical Partial Differential Equations: Finite Difference Methods*, New York: Springer, 1995.

[4] O.C. Zienkewicz, *The Finite Element Method*, 5th ed., London: McGraw-Hill, 2000.

[5] Ingrid Daubechies, *Ten Lectures on Wavelets*, Philadelphia: SIAM, 1992.

[6] Gilbert Strang and Truong Nguyen, *Wavelets and Filter Banks*, Wellesley: Wellesley-Cambridge Press, 1997.

[7] Stephane Mallat, *A Wavelet Tour of Signal Processing*, San Diego: Academic Press, 1998.

[8] Stefan Goedecker, *Wavelets and Their Applications for the Solution of Partial Differential Equations in Physics*, Lausanne: Presses polytechniques et universitaires romandes, 1998.

[9] Yves Nievergelt, *Wavelets Made Easy*, Boston: Birkhauser, 1999.

[10] Jadeiva C. Goswami, Andrew K. Chan, *Fundamentals of Wavelets: Theory, Algorithms and Applications*, New York: Wiley, 1999.

[11] G. Beylkin, R. Coifman and V. Rokhlin, “Fast wavelet transforms and numerical algorithms I”, *Commun. Pure Appl. Math*, vol. 44, pp. 141-183, 1991.

[12] G. Beylkin, “Wavelets and Fast Numerical Algorithms”, preprint, 1993.

[13] G. Beylkin, “On Wavelet-Based Algorithms for Solving Differential Equations”, preprint, 1992.

[14] B.K. Alpert, “Wavelets and other bases for fast numerical linear algebra,” in *Wavelets: A Tutorial in Theory and Applications,* C.K. Chui, Ed., New York: Academic Press, 1992.

[15] Wim Sweldens, “The Construction and Application of Wavelets in Numerical Analysis,” Ph.D Thesis, University of South Carolina/ Katholieke Universiteit Leuven, May 1995.

[16] Paulo J.C. Rodrigues, *Computer-aided analysis of nonlinear micro-wave circuits*, Norwood: Artech House, 1998.

[17] Kenneth S. Kundert, Jacob K. White and Alberto Sangiovanni-Vincentelli, Steady-state methods for simulating analog and microwave circuits, Boston: Kluwer Academic Publishers, 1990.

[18] Jiri Vlach and Kishore Singhal, *Computer methods for circuit analysis and design*, New York: Van Nostrand Reinhold, 1983.

[19] Kenneth S. Kundert, Gregory B. Sorkin and Alberto Sangiovanni-Vincentelli, “Applying Harmonic Balance to Almost-Periodic Circuits”, *IEEE Trans. on Microwave Theory and Techniques*, Vol. MTT-36, pp. 366-378, February 1988.

[20] G. Beylkin, “On the representation of operators in bases of compactly supported wavelets”, *SIAM J. on Numerical Analysis*, Vol. 6, No. 6, pp. 1716-1740, June 1992.

[21] Juan Mario Restrepo, Gary K. Leafy, “Inner Product Computations Using Periodized Daubechies Wavelets”, *International Journal of Numerical Methods in Engineering*, 40, pp. 3557-3578, 1997.

[22] V. Rizzoli, F. Mastri, F. Sgallari, and G. Spaletta, “Harmonic-balance simulation of strongly nonlinear very large-size microwave circuits by inexact newton methods”, *IEEE MTT-S Int. Microwave Symp. Dig.*, pp. 1357-1360, 1996.

[23] P. Feldmann, B. Melville, and D. Long, “Efficient frequency domain analysis of large nonlinear analog circuits”, *Proc. IEEE Custom Integrated Circuits Conf.*, pp. 461-464, 1996.

[24] D. Long, R. Melville, K. Ashby, and B. Horton, “Full-chip harmonic balance”, *Proc. IEEE Custom Integrated Circuits Conf.*, pp. 379 382, 1997.

[25] E. Gad, R. Khazaka, M. Nakhla and R. Griffith, “A circuit reduction technique for finding the steady-state solution of nonlinear circuits”, *IEEE Trans. Microwave Theory Tech.*, vol. 48, pp. 2389-2396, Dec. 2000.

[26] M. Steer and C. Christoffersen, “Generalized circuit formulation for the transient simulation of circuits using wavelet, convolution and time-marching techniques”, *Proc. of the 15th European Conference on Circuit Theory and Design*, Aug. 2001, pp. 205-208.

[27] Roni Khazaka, *Projection Based Techniques For The Simulation Of RF Circuits And High Speed Interconnects*, Ph.D. Thesis, Ottawa: Carleton University, 2002.

[28] *Matlab User Manual*, MathWorks Corporation, *http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml*

[29] John R. Gilbert, Cleve Moler and Robert Schrieber, “Sparse Matrices in Matlab: Design and Implementation”, *http://www.mathworks.com/access/helpdesk/help/pdf_doc/otherdocs/simax.pdf*

[30] T.A. Davis, *UMFPACK Version 4.0 User Guide* (http://www.cise.ufl.edu/research/sparse/umfpack/v4.0/UserGuide.pdf), Dept. of Computer and Information Science and Engineering, Univ. of Florida, Gainesville, FL, 2002.

[31] Alan George and Joseph Liu, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, 1981.

[32] Gene H. Golub, Charles F. Van Loan, *Matrix Computations*, Baltimore: The Johns Hopkins University Press, 1989.

[33] Nick Soveiko and Michel Nakhla, “Wavelet Harmonic Balance”, preprint, to appear in *IEEE Microwave and Wireless Components Lett.*, July 2003.

[34] Nick Soveiko and Michel Nakhla, “Steady State Analysis Of Multitone Nonlinear Periodic Circuits In Wavelet Domain”, preprint, to appear in *2003 Int. Microwave Symp. Digest*, Philadelphia: June 8-13, 2003.

[35] A. Cohen, I. Daubechies, P. Vial, “Wavelets on the interval and fast wavelet transforms”, *Appl. Comput. Harm. Anal.*, 1:54--81, 1993.

[36] D. Zhou, N. Chen and W. Cai, “A fast wavelet collocation method for high-speed VLSI circuit simulation”, *1995 IEEE/ACM ICCAD Symp. Digest*, pp. 115-122, 1995.

[37] D. Zhou, X. Li, W. Zhang and W. Cai, “Nonlinear circuit simulation based on adaptive wavelet method”, *1997 ISCAS Symp. Digest*, Vol. 3, pp. 1720-1723, 1997.

[38] D. Zhou and W. Cai, “A fast wavelet collocation method for high-speed VLSI circuit simulation”, *IEEE Trans. on Circuits and Systems -- I: Fundamental Theory and Appl.*, Vol. 46, pp. 920-930, Aug. 1999.

[39] D. Zhou, W. Cai and W. Zhang, “An adaptive wavelet method for nonlinear circuit simulation”, *IEEE Trans. on Circuits and Systems -- I: Fundamental Theory and Appl.*, Vol. 46, pp. 931-938, Aug. 1999.

[40] C. E. Christoffersen and M. B. Steer, “State-Variable-Based Transient Circuit Simulation Using Wavelets”, *IEEE Microwave and Guided Waves Letters*, April 2001, pp. 161-163.

[41] C. E. Christoffersen and M. B. Steer, “Comparison of Wavelet- and Time-Marching-Based Microwave Circuit Transient Analyses” *2001 IEEE Int. Microwave Symp. Digest*, May 2001, pp. 447-450.

[42] R. Achar and M. Nakhla, “Simulation of high-speed interconnects”, *Proceedings of the IEEE*, Vol. 89, No.5, May 2001.

[43] A. Dounavis, R. Achar and M. Nakhla, “A General Class of Passive Macromodels for Lossy Multiconductor Transmission Lines”, *IEEE Trans. Microwave Theory Tech.*, Vol. 49, No. 10, pp. 1686-1696, Oct. 2001.

[44] M.Celik, A. C. Cangellaris and A. Yaghmour, “An all purpose transmission line model for interconnect simulation in SPICE”, *IEEE Trans. Microwave Theory Tech.*, Vol.45, No. 10, pp.1857-1867, Oct. 1997.

[45] S. Grivet-Talocia, F. Canavero, “Wavelet-Based Adaptive Solution for the Nonuniform Multiconductor Transmission Lines”, *IEEE Microwave and Guided Wave Letters*, pp. 287-289, vol. 8, No. 8, August 1998.

[46] S. Grivet-Talocia, “Adaptive transient solution of nonuniform multiconductor transmission lines using wavelets”, *IEEE Transactions on Antennas and Propagation*, pp. 1563-1573, vol. 48, October 2000.

[47] S. Grivet-Talocia, F. G. Canavero, “Wavelet-Based High-Order Adaptive Modeling of Lossy Interconnects”, *IEEE Transactions on Electromagnetic Compatibility, Special Issue on ‘Recent Advances in EMC of Printed Circuit Boards’*, pp. 471-484, vol. 43, No. 4, November 2001.

[48] M. Raugi, “Wavelet Transform Solution of Multiconductor Transmission Line Transients”, IEEE Trans. Magn., pp. 1554-1557, Vol. 35, No. 3, May 1999.

[49] S. Barmada and M. Raugi, “Transient Numerical Solutions of Nonuniform MTL Equations with Nonlinear Loads by Wavelet Expansion in Time or Space Domain”, IEEE Trans. Circuits Syst. I -- Fund. Theory Appl., pp. 1178-1190, Vol. 47, No. 8, Aug. 2000.

[50] S. Barmada, A. Musolino, and M. Raugi, “Hybrid F.E. Wavelet Method for Nonlinear Analysis of Nonuniform MTL Transients”, *IEEE Trans. Magn.*, pp. 977-981, Vol. 36, No. 4, July 2000.

[51] B.Z. Stejnberg and Y. Leviatan, “On the Use of Wavelet Expansion in the Method of Moments”, *IEEE Transactions on Antennas and Propagation*, Vol. 41, pp. 610-619, May 1993.

[52] G. Pan, “Orthogonal Wavelets with Applications in Electromagnetics”, *IEEE Transactions on Magnetics*, Vol. 32, No. 3, May 1996, pp. 975-983.

[53] R.L. Wagner, Weng Cho Chew, “A study of wavelets for the solution of electromagnetic integral equations”, *IEEE Transactions on Antennas and Propagation*, Vol. 43 No. 8, pp. 802 -810, Aug 1995.

[54] W. Dahmen and R. Schneider, “Composite Wavelet Bases for Operator Equations”, *Math. Comp.*, 68 (1999), pp. 1533-1567.

[55] W. Dahmen, A. Kunoth and K. Urban, “Wavelets in Numerical Analysis and their Quantitative Properties,” in *Surface Fitting and Multiresolution Methods*, A. Le Mehaute, C. Rabut, and L. L. Schumaker (eds.), Vanderbilt Univ. Press, Nashville, TN, 1997, pp. 93-130.

[56] T. von Petersdorff and C. Schwab, “Fully Discrete Multiscale Galerkin BEM”, in *Multiscale Wavelet Methods for Partial Differential Equations*, Wolfgang Dahmen et al., Eds., New York: Academic Press, 1997.

[57] Clayton R. Paul, *Analysis of Multiconductor Transmission Lines*, New York: Wiley, 1994.

[58] W. Hong et al., “A Novel Dimension Reduction Technique for the Capacitance Extraction of 3D VLSI Interconnects”, *1996 IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers*, IEEE Computer Society Press, 1996, pp. 381-386.

[59] U. Choudhury and A. Sangiovanni-Vintecelli, “Automatic Generation of Analytical Models for Interconnect Capacitances”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 14, No. 4, April 1995, pp. 470-480.

[60] “Interconnect Parasitic Extraction For Deep Submicron IC Design. xCalibre^{TM} White Paper”, Mentor Graphics Corporation, *http://www.mentorg.com/dsm/*.

[61] K.E. Atkinson, “A survey of boundary integral equation methods for the numerical solution of Laplace’s equation in three dimensions,” in *Numerical Solution of Integral Equations*, M. Goldberg, ed., Plenum Press, 1990.

[62] K. Nabors and J. White, “FastCap: A Multipole Accelerated 3-D Capacitance Extraction Program”, *IEEE Transactions on Computer-Aided Design*, Vol. 10, No. 11, November 1991, pp. 1447-1459.

[63] S. Kapur and D.E. Long, “IES^3: A Fast Integral Equation Solver for Efficient 3-dimensional Extraction”, *1997 IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers*, IEEE Computer Society Press, 1997, pp. 448-455.

[64] J.R. Phillips and J. White, “A Precorrected-FFT Method for Capacitance Extraction of Complicated 3-D Structures”, *1994 IEEE/ACM International Conference on Computer-Aided Design, Digest of Technical Papers*, IEEE Computer Society Press, 1994, pp. 268-271.

[65] W. Shi, J. Liu, N. Kakani and T. Yu, “A Fast Hierarchical Algorithm for 3-D Capacitance Extraction”, preprint, 1998.

[66] N. Soveiko, “Application of Norm Concepts to Evaluation of Multiconductor Transmission Line Parameters”, preprint, presented at the AMEREM’96 - The World of Electromagnetics, Albuquerque, NM, May 27-31, 1996, *http://www.doe.carleton.ca/~nsoveiko/research/papers.html,* 1996.

[67] G. Wang, G. Pan and B.K. Gilbert, “A Hybrid Wavelet Expansion and Boundary Element Analysis for Multiconductor Transmission

Lines in Dielectric Media," *IEEE Transactions on Microwave Theory and Techniques*, Vol. MTT-43, pp. 664-674, March 1995.

[68] K.S. Osh, D. Kuznetsov and J.E. Schutt-Aine, "Capacitance Computations in a Multilayered Dielectric Medium Using Closed-Form Spatial Green's Functions," *IEEE Transactions on Microwave Theory and Techniques*, Vol. MTT-42, pp. 1443-1453, August 1994.

[69] K. Li, K. Atsuki and T. Hasegawa, "General Analytical Solutions of Static Green's Functions for Shielded and Open Arbitrary Multilayered Media," *IEEE Transactions on Microwave Theory and Techniques*, Vol. MTT-45, pp. 2-8, January 1997.

[70] C. Wei et al., "Multiconductor transmission lines in multilayered dielectric media," *IEEE Trans. Microwave Theory Tech.*, vol. MTT-32, pp. 439-450, Apr. 1984.

[71] N. Soveiko and M.Nakhla, "Efficient Capacitance Extraction Computations in Wavelet Domain", *IEEE Trans. on Circuits and Systems -- I: Fundamental Theory and Applications*, vol. 47, pp. 684-701, May 2000.

THE END