Proceedings of the 4th WSEAS/IASME Int. Conf. on System Science and Simulation in Engineering, Tenerife, Spain, December 16-18, 2005 (pp83-89)

A Spline Wavelet Approach to Circuit Simulation

Yi Hu, Dian Zhou, Ruiming Li, and Hua Zhang
Department of Electrical Engineering
University of Texas at Dallas
Richardson, Texas 75083-0688, USA

Xuan Zeng
Microelectronics Department
Fudan University
Shanghai 200433, P. R. China

Abstract: A two-step spline wavelet approach to circuit simulation in the digital signal processing perspective is proposed in this paper. First a fast recursive B-spline filtering approach to solving ordinary differential equations is introduced, then the fast wavelet decomposition and reconstruction algorithms are applied using multiresolution analysis. This work also exploits the relationship between digital signal processing theory and the numerical methods for circuit simulation.

Key–Words: spline wavelet, circuit simulation, digital signal processing, numerical methods, filtering

1 Introduction

Recently the application of wavelet theory to circuit simulation has received considerable interests [10, 11, 3, 4], partly due to the fact that it is a promising approach to deal with the difficulties encountered by the conventional circuit simulation methods, such as nonuniform error distribution and singularities which often develop in high speed circuit.

The wavelet collocation method proposed in [10] was the first wavelet approach to circuit simulation, and the basic idea is that the solutions of ordinary differential equations (ODEs) which describe VLSI systems are decomposed into wavelet representations, then the wavelet coefficients are obtained by a fast collocation method. It was shown that the wavelet collocation method was very effective to handle the uniform error distribution problem and provided a mechanism to capture the signal singularities arising in high speed circuits. In [11] the wavelet collocation method was further generalized into nonlinear circuit simulation, and later, the work in [3, 4] successfully applied this method to the behavioral modelling for analog system-level simulation to address the issue of error distribution control.

However, wavelet collocation method is a theoretically complex method and it is still very difficult for the engineering society to grasp the whole idea and apply to their own applications. Additionally its connection to the signal processing area where the whole wavelet theory originates and continually enjoys a huge success is not clear. In this paper, we try to exploit the possible applications of digital signal processing theory to the numerical methods for circuit simulation, and we will focus on the spline wavelet functions [1, 10].

Spline functions are used extensively in interpolation problems, and the traditional method to find the spline coefficients is by solving linear equations. Unser and his colleagues [8, 9] were among the first to recognize the connection between spline interpolation and digital filtering, and developed a fast filtering algorithm for spline interpolation. In this paper, we further utilize this connection for solving ODEs and correspondingly derive a recursive filtering algorithm. After obtaining spline coefficients, we can just use the filter bank approach for spline wavelet decomposition and reconstruction. Since the latter is a well established result in wavelet theory [1], to save space, we concentrate on the description of the recursive spline filtering algorithm. Our experimental results show that the proposed spline wavelet approach is very promising, and more importantly, to our best knowledge this work is the first to exploit the connection between the two areas, namely, digital signal processing and numerical methods for circuit simulation.

This paper is organized as follows. In section 2, the important properties of linear digital system and B-spline function are introduced, and the multiresolution analysis is also described. The proposed approach is presented in section 3, the applications are given in 4, and the conclusions are given in section 5.

2 Preliminaries

2.1 Discrete-time Signal and z Transform

The space L_2 of square-summarizable real-valued sequences $\{x(k)\}_{k\in Z}$ is a Hilbert space whose metric is derived from the standard inner product: $< \mathbf{x}, \mathbf{h} >=$

$\sum_k x(k) \cdot h(k)$. The convolution operation between two sequences $x(n) \in L_2$ and $h(n) \in L_2$ is defined as:

$$x(n)*h(n) = \sum_k x(n-k)\cdot h(k) = \sum_k x(k)\cdot h(n-k) \quad (1)$$

A well known result about the linear-time invariant system is that the system output $y(n)$ can be obtained by the convolution of the system input $x(n)$ and the system impulse response $h(n)$, i.e., $y(n) = x(n) * h(n)$. Another way to characterize the sequence $h(n)$ is by its z transform which is defined by $h(z) = \sum_k h(k) \cdot z^{-k}$. The z transform of the convolution of two sequences $x(n)$ and $h(n)$ can be obtained by the product of the z transform of the two sequences, i.e.,

$$y(n) = x(n) * h(n) \Leftrightarrow y(z) = x(z) \cdot h(z)$$

and the shifting property of z transform can be described as:

$$y(n) = x(n - n_0) \Leftrightarrow y(z) = x(z) \cdot z^{-n_0} \quad (2)$$

2.2 Sampling and Discrete B-Splines

The time-marching methods for circuit simulation calculate the system output at discrete points. If we treat this computation process as a procedure of sampling the solution variables, we can construct a connection between the continuous-time linear system described by the ordinary differential equations and the discrete-time linear system described by the linear equations. As we will show later, this is a very important step, which enables us to directly apply those well established results in digital signal processing theory to the numerical methods for circuit simulation. For simplicity and clarity, in this paper we only discuss the method to compute the system output at the equidistance points (i.e., a uniform sampling procedure is used), and the case of a variable step length can be discussed similarly.

Now assuming that the system output sequence $g(n)$ at those sampling points has been obtained, we are concerned with the problem of how to interpolate $g(n)$ to approximate the true system output $g(t)$. For now, we focus on utilizing polynomial splines to accomplish this task [7, 2, 8]. The polynomial spline functions of order i with unit spaced knots span a space S_1^i which is a subspace of L_2, where the superscript i indicates the degree of the piecewise polynomial segment, and the subscript 1 means that the piecewise polynomial segments join at the unit spaced knots. Another convenient way to represent S_1^i is by using i-th order B-spline basis function, i.e.,$S_1^i =$

i	2	3
z-transform	$\frac{z^{-1}+z^{-2}}{2}$	$\frac{z^{-1}+4z^{-2}+z^{-3}}{6}$

Table 1: Example z transform of 2nd and 3rd B-spline.

$\{g^i(t) = \sum_k y(k)\phi^i(t-k), (t \in R, y \in L_2)\}$, where $\phi^i(t)$ is the B-spline of order i with compact support $supp(\phi^i(t)) = [0, i+1]$ and is defined as:

$$\phi^i(t) = \sum_{j=0}^{i+1} \frac{(-1)^j}{i!} \left(\begin{array}{c} i+1 \\ j \end{array} \right) (t-j)_+^i \quad (3)$$

and where the function t_+^i is defined as:

$$t_+^i = \left\{ \begin{array}{ll} t^i, & t \geq 0 \\ 0, & t < 0 \end{array} \right. \quad (4)$$

Obviously any polynomial spline function $g^i(t) \in S_1^i$ can be written as a weighted sum of the shifted B-spline, and the discrete-time sequence $g^i(n)$, $n \in Z$ can be obtained by:

$$\begin{aligned} g^i(n) &= g^i(t)|_{t=n} = \sum_k y(k)\phi^i(n-k) \\ &= \sum_k y(n-k)\phi^i(k) = y(n) * \phi^i(n) \end{aligned} \quad (5)$$

where the sequence $\phi^i(n)$ is defined as $\phi^i(n) = \phi^i(t)|_{t=n},\ n \in [1, i]$, $\phi^i(n)$ is referred to as discrete B-spline, which can be viewed as a finite impulse response (FIR) filter and is totally described by its z transform $\phi^i(z)$. Table 1 presents the z transform of the 2nd order and 3rd order (cubic) B-spline. It can be seen from (5) that the z transform of $y(n)$ is obtained by:

$$y(z) = \frac{1}{\phi^i(z)} \cdot g^i(z) \quad (6)$$

where $g^i(z)$ is the z transform of $g^i(n)$. The above equation means that the coefficients sequence $y(n)$ can be recovered by applying an infinite impulse response (IIR) filter $1/\phi^i(z)$ to $g^i(n)$. To make this clearer, next we give a simple example. Suppose we use the cubic B-spline shown in Table 1, i.e., $\phi^3(z) = z^{-1}/6 + 2z^{-2}/3 + z^{-3}/6$, plugging it into (6), we will get $y(z) = 6/(z^{-1} + 4z^{-2} + z^{-3}) \cdot g^3(z)$. Applying the inverse z transform to the above equation and using (2), we will get $y(n-1) = 6g^3(n) - 4y(n-2) + y(n-3)$, therefore $y(n)$ can be iteratively calculated.

Now we derive a similar expression for the sequence $g_1^i(n)$ which is defined as $g_1^i(n) =$

i	2	3
z-transform	$z^{-1}-z^{-2}$	$\frac{z^{-1}-z^{-3}}{2}$

Table 2: Example z transform of $\phi_1^i(n), i = 2, 3$.

$\mathrm{d}\, g^i(t)/\,\mathrm{d}\, t|_{t=n}$,

$$\frac{\mathrm{d}\, g^i(t)}{\mathrm{d}\, t} = \sum_k y(k) \frac{\mathrm{d}\, \phi^i(t-k)}{\mathrm{d}\, t} = \sum_k y(k)\phi_1^i(t-k) \tag{7}$$

where $\phi_1^i(t)$ is defined as $\phi_1^i(t) = \frac{\mathrm{d}\, \phi^i(t)}{\mathrm{d}\, t}$, clearly $\phi_1^i(t)$ is also a function with compact support $supp(\phi_1^i(t)) = [0, i+1]$. Now $g_1^i(n)$ can be obtained from:

$$\begin{aligned} g_1^i(n) &= \sum_k y(k)\phi_1^i(n-k) = \sum_k y(n-k)\phi_1^i(k) \\ &= y(n) * \phi_1^i(n) \end{aligned} \tag{8}$$

where $\phi_1^i(n)$ can also be viewed as a FIR filter applied to $y(n)$ to obtain $g_1^i(n)$. The example z transform of $\phi_1^i(n), i = 2, 3$ is given in Table 2.

Equations (5) and (8) mean that the sequences $g^i(n)$ and $g_1^i(n)$ can be generated by applying the FIR filters $\phi^i(n)$ and $\phi_1^i(n)$ to $y(n)$, respectively. In the following sections, we will show that these results lead to a fast recursive filtering technique to solving the ordinary differential equations.

2.3 Multiresolution Spaces and Multiresolution Analysis

The theory of multiresolution analysis (MRA) was proposed by Meyer and Mallat as the most important building block for the construction of scaling functions and wavelet functions [5, 1], it also reveals the fundamental connections between the wavelet theory and the theory of multirate signal processing and filter banks. Simply put, MRA addresses the issue of how to approximate a complicated function in L_2 space at different resolutions or scales (obviously a smaller scale means a larger resolution). In current scenario, resolution level is directly related to the number of uniform collocation or sampling points in a unit length, a large number of sampling points indicates a high resolution and vice versa. In this paper, the sampling rate is set to 2^J at resolution level J (or scale level $-J$).

The function is projected onto a subspace V_J at resolution level J, and these subspaces are called multiresolution spaces. In MRA the subspace at resolution level $J+1$ always contains the subspace at resolution level J, more specifically, $\{0\} \subset \cdots V_0 \subset \cdots V_J \subset V_{J+1} \subset \cdots L_2$, and all the multiresolution spaces are spanned by an appropriately scaled version of the same prototype scaling function $\phi(x)$. In other words, at resolution level J, the integer translated functions $\sqrt{2}^J\phi(2^Jx-k), k \in Z$ form a Riesz basis and span the multiresolution subspace V_J [5]. Additionally, if a function $f_J(x) \in V_J$, then its scaled version $f_J(2x)$ will reside in V_{J+1}.

In the multiresolution space V_{J+1} there exists a subspace W_J which is orthogonal to V_J, clearly the direct sum of W_J and V_J form V_{J+1},

$$V_{J+1} = V_J \oplus W_J \tag{9}$$

W_J is called wavelet subspace and is also generated by a scaled version $\psi_J(x)$ of the prototype wavelet function $\psi(x)$, i.e., $W_J = \mathrm{span}\{\sqrt{2}^J\psi(2^Jx-k), k \in Z\}$. There are two fundamental relations between the basis functions $\phi_J(x)$ and $\psi_J(x)$ that generate V_J and W_J, respectively, and the basis function $\phi_{J+1}(x)$ that generates V_{J+1}. The first one is referred to as two-scale relation and is expressed as

$$\begin{aligned} \phi(2^Jx) &= \sum_k p_0(k)\phi(2^{J+1}x-k) \\ \psi(2^Jx) &= \sum_k p_1(k)\phi(2^{J+1}x-k) \end{aligned} \tag{10}$$

and the second one is referred to as decomposition relation and is expressed as

$$\begin{aligned} \phi(2^{J+1}x-l) &= \sum_k \{h_0(2k-l)\phi(2^Jx-k) \\ &+ h_1(2k-l)\psi(2^Jx-k)\} \end{aligned} \tag{11}$$

With (9), a function $f_{J+1}(x) = \sum_k y_{J+1}(k)\phi(2^{J+1}x-k) \in V_{J+1}$ can be decomposed into two orthogonal parts, namely, $f_J(x) = \sum_k y_J(k)\phi(2^Jx-k) \in V_J$, and $q_J(x) = \sum_k w_J(k)\psi(2^Jx-k) \in W_J$. It can be shown that by applying the two-scale relation (10) and the decomposition relation (11), one can obtain

$$\begin{aligned} y_J(k) &= \sum_l h_0(2k-l)y_{J+1}(l) \\ w_J(k) &= \sum_l h_1(2k-l)y_{J+1}(l) \end{aligned} \tag{12}$$

and

$$y_{J+1}(k) = \sum_l \{p_0(k-2l)y_J(l) + p_1(k-2l)w_J(l)\} \tag{13}$$

Equations (12) and (13) provide the decomposition process and the reconstruction process, respectively. In signal processing perspective, the four sequences h_0, h_1, p_0 and p_1 can be viewed as filters (FIR

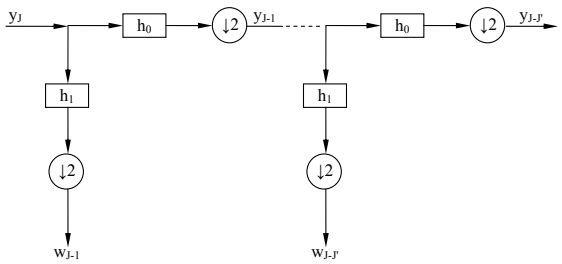

Figure 1: Fast decomposition tree

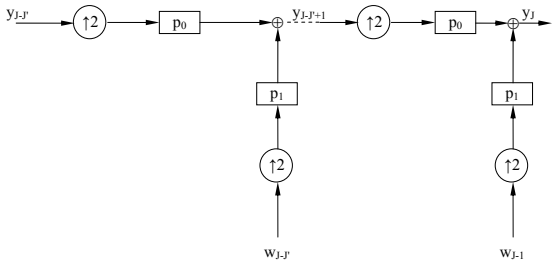

Figure 2: Fast reconstruction tree

or IIR), so the decomposition and reconstruction procedures are merely filtering operations. With this in mind, (12) and (13) can be rewritten as

$$\begin{aligned} y_J(k) &= (y_{J+1}(k) * h_0(k))_{\downarrow 2} \\ w_J(k) &= (y_{J+1}(k) * h_1(k))_{\downarrow 2} \end{aligned} \tag{14}$$

and

$$y_{J+1}(k) = (y_J(k))_{\uparrow 2} * p_0(k) + (w_J(k))_{\uparrow 2} * p_1(k) \tag{15}$$

where $_{\downarrow 2}$ is a downsampling operator which keeps the even indexed samples and drops the odd indexed samples, and $_{\uparrow 2}$ is an upsampling operator which inserts a 0 value between adjacent samples [5, 1]. By iteratively applying (14) and (15), we can obtain a decomposition tree and a reconstruction tree shown in figures 1 and 2, respectively.

Interesting enough, an arbitrary order B-spline function satisfies all the conditions needed to be a scaling function and can be used to generate the multiresolution spaces. This also means that any function in L_2 space can always be approximated with arbitrarily small approximation error at high enough a resolution level, in other words, by increasing the sampling rate we can approximate any L_2 function with the scaled B-spline function. Correspondingly, the spline wavelet function $\psi_J(x)$ generates the wavelet spaces W_J, $J \in Z$, according to (10), the prototype wavelet function $\psi(k), k \in Z$ is obtained by $\psi(k) = (p_1(k) * \phi(k))_{\downarrow 2}$, for an i-th order B-spline function $\phi^i(x)$, $p_1(k)$ and $p_0(k)$ are two FIR filters defined in [1].

3 The Proposed Approach

In this section, the proposed spline wavelet approach is presented. According to the MRA theory, after we obtain the spline coefficients at the highest resolution level, we can always utilize a fast filter bank approach to get the wavelet coefficients at various resolution levels, so here we focus on developing the fast recursive spline filtering method to solve ODEs at a designated resolution level.

3.1 Principles of the Recursive Spline Filtering Approach

Suppose we want to numerically solve the following first-order ordinary differential equations:

$$\begin{cases} \mathbf{A}\frac{\mathrm{d}\,\mathbf{f}(t)}{\mathrm{d}\,t} + \mathbf{C}\mathbf{f}(t) = \mathbf{u}(t) \\ \mathbf{f}(0) = \mathbf{f}_0 \end{cases} \tag{16}$$

where $\mathbf{f}(t)$ is the unknown N-by-1 vector function, $\mathbf{u}(t)$ is a known N-by-1 vector function, $\mathbf{A} = \{a_{ij}\}$ and $\mathbf{C} = \{c_{ij}\}$ are two N-by-N constant matrices, and $\mathbf{f}_0$ is the initial condition of $\mathbf{f}(t)$.

We obtain the collocation points $\mathbf{f}(t_n)$ by uniformly sampling the time t with a sampling rate M, i.e., $\mathbf{f}(t_n) = \mathbf{f}(n/M),\ n \in Z$. To utilize the recursive filtering techniques, we do the following transform $\mathbf{g}(t) = \mathbf{f}(t/M)$, now the collocation points $\mathbf{f}(t_n)$ will become $\mathbf{g}(t)|_{t=n}$, and equation (16) can be rewritten as:

$$\begin{cases} \mathbf{A}\frac{\mathrm{d}\,\mathbf{g}(t)}{\mathrm{d}\,t} + \mathbf{B}\mathbf{g}(t) = \mathbf{x}(t) \\ \mathbf{g}(0) = \mathbf{f}_0 \end{cases} \tag{17}$$

where $\mathbf{B} = \{b_{ij}\} = \{c_{ij}/M\}$ and $\mathbf{x}(t) = \frac{1}{M}\mathbf{u}(t/M)$.

With a high enough sampling rate M, the function $\mathbf{f}(t)$ can be approximated satisfactorily by a function $\mathbf{g}^i(t)$ in S_1^i,

$$\mathbf{g}^i(t) = \sum_{k=1}^{i} \mathbf{y}(t-k) \cdot \phi^i(k) \tag{18}$$

where $\mathbf{y}(t) = (y_1(t) \dots y_N(t))^T$ is the coefficients vector, correspondingly, the first order differentiation of $\mathbf{g}(t)$ can be written as $\frac{\mathrm{d}\,\mathbf{g}(t)}{\mathrm{d}\,t} = \sum_{k=1}^{i} \mathbf{y}(t-k) \cdot \phi_1^i(k)$. By denoting the row vector of $\mathbf{A}$ as $\mathbf{A}_i^T$, and the row vector of $\mathbf{B}$ as $\mathbf{B}_i^T$, now the first equation of

(17) becomes

$$\left(\begin{array}{c}\mathbf{A}_1^T\\ \vdots\\ \mathbf{A}_N^T\end{array}\right)\cdot\sum_{k=1}^{i}\left(\begin{array}{c}y_1(t-k)\phi_1^i(k)\\ \vdots\\ y_N(t-k)\phi_1^i(k)\end{array}\right)+$$
$$\left(\begin{array}{c}\mathbf{B}_1^T\\ \vdots\\ \mathbf{B}_N^T\end{array}\right)\cdot\sum_{k=1}^{i}\left(\begin{array}{c}y_1(t-k)\phi^i(k)\\ \vdots\\ y_N(t-k)\phi^i(k)\end{array}\right)=\mathbf{x}(t) \quad (19)$$

which means that for the l-th row, the following always holds:

$$\sum_{k=1}^{i}\left(\mathbf{A}_l^T\left(\begin{array}{c}y_1(t-k)\\ \vdots\\ y_N(t-k)\end{array}\right)\phi_1^i(k)\right)+$$
$$\sum_{k=1}^{i}\left(\mathbf{B}_l^T\left(\begin{array}{c}y_1(t-k)\\ \vdots\\ y_N(t-k)\end{array}\right)\phi^i(k)\right)=x_l(t) \quad (20)$$

It can be shown that the above equation is equal to:

$$\sum_{j=1}^{N}\sum_{k=1}^{i}\left(a_{lj}\phi_1^i(k)+b_{lj}\phi^i(k)\right)y_j(t-k)=x_l(t) \quad (21)$$

since t indicates integer index, we have

$$\sum_{k=1}^{i}\left(a_{lj}\phi_1^i(k)+b_{lj}\phi^i(k)\right)y_j(t-k)=\xi_{lj}(t)*y_j(t) \quad (22)$$

where $\xi_{lj}(t)=a_{lj}\phi_1^i(t)+b_{lj}\phi^i(t)$. Now equation (21) can be rewritten as

$$\sum_{j=1}^{N}\xi_{lj}(t)*y_j(t)=x_l(t) \quad (23)$$

After applying the z transform, we obtain the following equation:

$$\boldsymbol{\xi}(z)\cdot\mathbf{y}(z)=\mathbf{x}(z) \quad (24)$$

where $\boldsymbol{\xi}(z)=\{\xi_{lj}(z)\}_{N\times N}$, and

$$\mathbf{y}(z)=(y_1(z)\dots y_N(z))^T,\ \mathbf{x}(z)=(x_1(z)\dots x_N(z))^T$$

Clearly $\xi_{lj}(z)$ is defined as $\xi_{lj}(z)=a_{lj}\phi_1^i(z)+b_{lj}\phi^i(z)$, and $\boldsymbol{\xi}(z)=\sum_{k=1}^{i}(\mathbf{A}\phi_1^i(k)+\mathbf{B}\phi^i(k))z^{-k}$. Plugging the above equation into (24) and applying the inverse z transform, we will have: $\sum_{k=1}^{i}(\mathbf{A}\phi_1^i(k)+\mathbf{B}\phi^i(k))\mathbf{y}(n-k)=\mathbf{x}(n)$, which can be rewritten as:

$$(\mathbf{A}\phi_1^i(1)+\mathbf{B}\phi^i(1))\mathbf{y}(n-1)=\mathbf{x}(n)-\sum_{k=2}^{i}(\mathbf{A}\phi_1^i(k)+\mathbf{B}\phi^i(k))\mathbf{y}(n-k) \quad (25)$$

So it is clear that the coefficient vectors for the collocation points can be obtained iteratively, given the initial values for $\mathbf{y}(n), n\in[-i,-1]$, as shown next, we can exactly establish i equations to solve the i initials values. Now we have $\mathbf{y}(n)$, the sequence $\mathbf{g}^i(t)|_{t=n}$ can be obtained from:

$$\mathbf{g}^i(n)=\mathbf{y}(n)*\phi^i(n) \quad (26)$$

and it is well known that the above calculation can be implemented with Fast Fourier Transform (FFT) for a large sampling rate M.

3.2 A Cubic B-Spline Example

For a cubic B-spline example, $\xi_{lj}(z)$ and $\boldsymbol{\xi}(z)$ are defined as

$$\xi_{lj}(z)=(\frac{1}{2}a_{lj}+\frac{1}{6}b_{lj})z+\frac{4}{6}b_{lj}+(\frac{1}{6}b_{lj}-\frac{1}{2}a_{lj})z^{-1}$$

$$\boldsymbol{\xi}(z)=(\frac{1}{2}\mathbf{A}+\frac{1}{6}\mathbf{B})z^{-1}+\frac{4}{6}\mathbf{B}z^{-2}+(\frac{1}{6}\mathbf{B}-\frac{1}{2}\mathbf{A})z^{-3}$$

with (25), we have:

$$(\frac{1}{2}\mathbf{A}+\frac{1}{6}\mathbf{B})\cdot\mathbf{y}(n-1)=\mathbf{x}(n)-\frac{4}{6}\mathbf{B}\cdot\mathbf{y}(n-2)-(\frac{1}{6}\mathbf{B}-\frac{1}{2}\mathbf{A})\cdot\mathbf{y}(n-3) \quad (27)$$

So it is clear that the coefficient vectors for the collocation points can be obtained iteratively, given the initial values for $\mathbf{y}(-1)$, $\mathbf{y}(-2)$ and $\mathbf{y}(-3)$ which can be obtained by let $n=0$ in (25), let $t=0$ in (18), and let $t=0$ after differentiating equation (17), i.e.,

$$\left(\begin{array}{ccc}\frac{1}{2}\mathbf{A}+\frac{1}{6}\mathbf{B} & \frac{4}{6}\mathbf{B} & \frac{1}{6}\mathbf{B}-\frac{1}{2}\mathbf{A}\\ \frac{1}{6}\mathbf{I}_{N\times N} & \frac{4}{6}\mathbf{I}_{N\times N} & \frac{1}{6}\mathbf{I}_{N\times N}\\ \mathbf{A}+\frac{1}{2}\mathbf{B} & -2\mathbf{A} & \mathbf{A}-\frac{1}{2}\mathbf{B}\end{array}\right)\cdot\left(\begin{array}{c}\mathbf{y}(-3)\\ \mathbf{y}(-2)\\ \mathbf{y}(-1)\end{array}\right)=\left(\begin{array}{c}\mathbf{x}(0)\\ \mathbf{f}_0\\ \mathbf{x}'(0)\end{array}\right) \quad (28)$$

4 Discussions and Examples

In this section, we give several examples to demonstrate the applications of the proposed approach. Since the wavelet decomposition (12) and reconstruction (13) are very easy to understand, here we only give the applications of the proposed fast recursive spline filtering approach.

An obvious application of the proposed approach is that at a preset resolution level, if we want to extrapolate the function values at a higher resolution level,

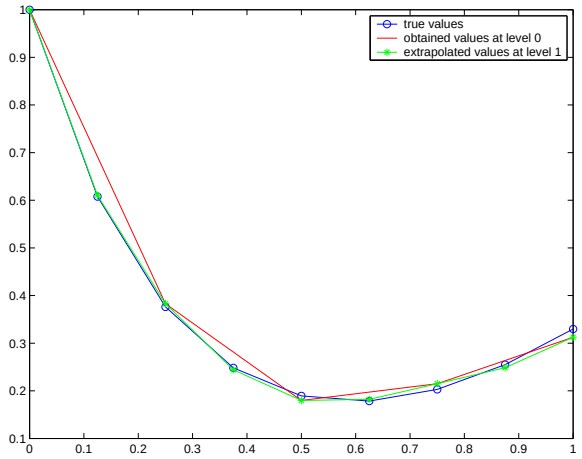

Figure 3: An example of extrapolation

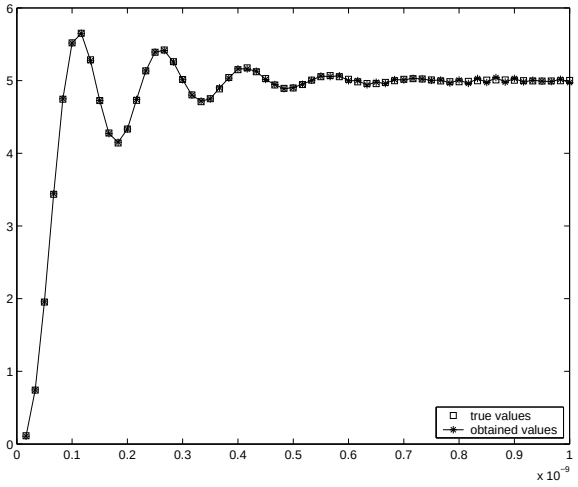

Figure 4: An example figure from [10]

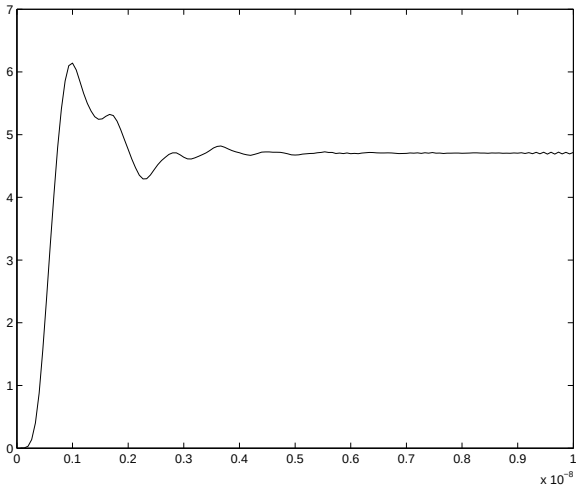

Figure 5: An example from [6]

it is just an upsampling operation followed by the FIR filtering, i.e.,

$$f'_J(k) = (y_J(k))_{\uparrow 2} * p_0(k) * \phi^i(k) \qquad (29)$$

To illustrate this point let's consider the following example,

$$\begin{cases} \frac{\mathrm{d}\, f}{\mathrm{d}\, t} = -4f(t) + 2t^2 \\ f(0) = 1 \end{cases} \qquad (30)$$

and we want to find the function values in the closed interval $[0, 1]$. Suppose the resolution level is set to 2 (the sampling rate M is $2^2 = 4$, and the maximum approximation error is about 5%), by applying the proposed approach and using (29), we will obtain Figure 3.

The second example is a 2nd-order RLC network from [10] (pp. 926):

$$\begin{bmatrix} \frac{\mathrm{d}\, v_c}{\mathrm{d}\, t} \\ \frac{\mathrm{d}\, i_l}{\mathrm{d}\, t} \end{bmatrix} = \begin{bmatrix} 0 & 1/C \\ -1/L & -R/L \end{bmatrix} \begin{bmatrix} v_c \\ i_l \end{bmatrix} + \begin{bmatrix} 0 \\ 1/L \end{bmatrix} E$$

where $C = 58pF, R = 0.124\Omega, L = 10pH$. The voltage output of the above RLC network is shown in Figure 4.

The third example is from the classical work by Pillage and Rohrer [6] (Fig. 26, pp. 365). Figure 5 is the results obtained with the proposed approach, which is within 1% error of the Spice output.

5 Conclusions

An alternative spline wavelet approach to circuit simulation was proposed in this paper. The main difference between the proposed method and the wavelet collocation method is that, in the collocation method the underlying function is directly decomposed into the wavelet basis functions at all levels and a matrix form is formulated, while in the proposed approach the MRA is applied, and the decomposition is carried out in two steps, first the expansion coefficients with respect to the scaling spline basis functions at the highest resolution level are obtained using the fast recursive filtering approach, then the wavelet coefficients at all levels are obtained by applying the fast decomposition algorithm (14). An obvious advantage of the proposed approach is that it is highly structured, therefore hardware implementation will be much more easier, the proposed approach is also better in terms of the computation complexity. Additionally, the connection between digital signal processing theory and numerical methods for circuit simulation was also exploited.

Acknowledgements: The research is supported in part by the University of Texas at Dallas and NSF

grant CCR-0306298, and is also supported partly by NSFC research project 90307017, 60176017, partly by Cross-Century Outstanding Scholars fund of Ministry of Education of China, partly by National 863 plan projects 2004AA1Z1050, Shanghai Science and Technology committee Key project 04JC14015 and Shanghai AM R&D fund 0418.

References:

[1] C. K. Chui. *An introduction to wavelets*. Academic Press, Inc, 1992.

[2] C. de Boor. *A practical guide to splines*. New York: Springer-Verlag, 1978.

[3] X. Li, X. Zeng, D. Zhou, and X. Ling. Behavioral modeling of analog circuits by wavelet collocation method. In *ICCAD*, pages 65–69, Nov 2001.

[4] X. Li, X. Zeng, D. Zhou, X. Ling, and W. Cai. Behavioral modeling for analog system-level simulation by wavelet collocation method. *IEEE Trans. Circuits Syst. II*, 50(6):299–314, June 2003.

[5] S. G. Mallat. A theory for multiresolution signal decomposition: The wavelet presentation. *IEEE Trans. Pattern Anal. Machine Intell.*, 11:674–693, July 1989.

[6] L. Pillage and R. Rohrer. Asymptotic waveform evaluation for timing analysis. *IEEE Trans. Computer-Aided Design*, 9:352–356, April 1990.

[7] L. L. Schumaker. *Spline Functions: Basic Theory*. Wiley, 1981.

[8] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing: Part i-theory. *IEEE Trans. Signal Processing*, 41:821–833, Feb 1993.

[9] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing: Part ii-efficient design and applications. *IEEE Trans. Signal Processing*, 41:834–848, Feb 1993.

[10] D. Zhou and W. Cai. A fast wavelet collocation method for high-speed circuit simulation. *IEEE Trans. Circuits Syst. I*, 46:920–930, Aug 1999.

[11] D. Zhou, W. Cai, and W. Zhang. An adaptive wavelet method for nonlinear circuit simulation. *IEEE Trans. Circuits Syst. I*, 46:931–938, Aug 1999.