Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2008, Article ID 323080, 16 pages
doi:10.1155/2008/323080

Research Article

Combined Preorder and Postorder Traversal Algorithm for the Analysis of Singular Systems by Haar Wavelets

Beom-Soo Kim,[1] Il-Joo Shim,[2] Myo-Taeg Lim,[3] and Young-Joong Kim[3]

[1] *School of Mechanical and Aerospace Engineering, Gyeongsang National University, 445 Inpyeong-Dong, Tongyeong, Gyeongnam 650-160, South Korea*
[2] *Department of Automatic System Engineering, Daelim College, 526-7 Bisan-Dong, Anyang, Gyeonggi 431-715, South Korea*
[3] *School of Electrical Engineering, Korea University, 1-5 Anam-dong, Sungbuk-gu, Seoul, 136-701, South Korea*

Correspondence should be addressed to Myo-Taeg Lim, mlim@korea.ac.kr

Received 31 May 2008; Accepted 25 August 2008

Recommended by Carlo Cattani

An efficient computational method is presented for state space analysis of singular systems via Haar wavelets. Singular systems are those in which dynamics are governed by a combination of algebraic and differential equations. The corresponding differential-algebraic matrix equation is converted to a generalized Sylvester matrix equation by using Haar wavelet basis. First, an explicit expression for the inverse of the Haar matrix is presented. Then, using it, we propose a combined preorder and postorder traversal algorithm to solve the generalized Sylvester matrix equation. Finally, the efficiency of the proposed method is discussed by a numerical example.

1. Introduction

Wavelets are mathematical functions that cut up data into different frequency components and then study each component with a resolution matched to its scale. Wavelets are now being applied in many areas of science and engineering [1–4]. Much attention has been focused on the use of wavelet transforms to investigate dynamic systems. This is due to the powerful ability of wavelet transforms to decompose time series in time-frequency domain and wavelet basis functions. Chen and Hsiao [3, 4] derived a Haar operational matrix for integration and solved the lumped and distributed parameter systems by constructing operational matrices of various order. The main characteristic of this technique is that it converts a differential equation into an algebraic one with the result that the solution procedures are greatly reduced and simplified. This approach gives new insight into the use of the Haar wavelet method.

Singular systems (also referred to as descriptor or semistate systems) arise more naturally than do state-variable descriptions in the analysis of many sorts of systems. Examples occur in electrical networks, neural networks, control systems, chemical systems, economic systems, and so on (see [5, 6] and references therein). These systems are governed by a mixture of differential and algebraic equations. The complex nature of singular systems causes many difficulties in the analytical and numerical treatment of such systems.

Recently, Haar wavelet technique was applied to state analysis and observer design of singular systems [7]. This approach replaces the state function and the forcing function by the truncated Haar series, respectively. Then the state trajectories are obtained by solving a generalized Sylvester matrix equation. But there exists a trade-off between the resolution of the wavelets and the computation time. The accuracy of the solution can be achieved by increasing the resolution level, but this requires more computation time and very large memory.

In this paper, an efficient computational method is presented for state space analysis of singular systems via Haar wavelets. First, an explicit expression for the inverse of the Haar matrix is presented. This inverse matrix also has a recursive structure. By using this matrix, we propose a combined preorder and postorder traversal algorithm. Then, the full-order generalized Sylvester matrix equation should be solved in terms of the solutions of simple linear matrix equations. Finally, the efficiency of the proposed method is discussed by a numerical example.

2. Kronecker product

Let $\mathbf{A} = [a_{ij}]$ and $\mathbf{B} = [b_{ij}]$ be $n \times p$ and $r \times q$ matrices, respectively. The Kronecker product of the matrices, denoted by $\mathbf{A} \otimes \mathbf{B} (\in \mathbb{R}^{nr \times pq})$, is defined as

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots & a_{1p}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & & a_{2p}\mathbf{B} \\ \vdots & & & \vdots \\ a_{n1}\mathbf{B} & a_{n2}\mathbf{B} & \cdots & a_{np}\mathbf{B} \end{bmatrix}. \tag{2.1}$$

The *vec* operator transforms a matrix $\mathbf{A}$ of size $n \times p$ to a vector of size $np \times 1$ by stacking the columns of $\mathbf{A}$. Some properties of the Kronecker product are given below [8]:

$$\begin{gathered} (\mathbf{A} + \mathbf{B}) \otimes \mathbf{C} = \mathbf{A} \otimes \mathbf{C} + \mathbf{B} \otimes \mathbf{C}, \\ (\mathbf{A} \otimes \mathbf{B})\mathbf{C} = (\mathbf{AC} \otimes \mathbf{B}), \\ (\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC} \otimes \mathbf{BD}), \\ (\mathbf{A} \otimes \mathbf{B})^T = \mathbf{A}^T \otimes \mathbf{B}^T. \end{gathered} \tag{2.2}$$

3. Haar wavelets and their properties

Wavelets constitute a family of functions constructed from dilation and translation of a single function called the mother wavelet that generates orthogonal bases of $L_2(R)$. The simplest and most basic of the wavelet systems is the Haar wavelet which is a group of square waves with magnitudes of ±1 in certain intervals and zeros elsewhere [9]. The scaling function $\varphi_0(t)$ and mother wavelet $\varphi_1(t)$ are defined by, respectively,

$$\varphi_0(t) = \begin{cases} 1, & t \in [0, 1), \\ 0, & t \notin [0, 1), \end{cases}$$
$$\varphi_1(t) = \begin{cases} 1, & t \in \left[0, \dfrac{1}{2}\right), \\ -1, & t \in \left[\dfrac{1}{2}, 1\right), \\ 0, & t \notin [0, 1). \end{cases} \tag{3.1}$$

Then, all the other basis functions $\varphi_k(t)$ are obtained by dilation and translation of the mother wavelet as follows:

$$\varphi_k(t) = \varphi_1\left(2^n t - j\right) = \begin{cases} 1, & t \in \left[t_a, t_b\right), \\ -1, & t \in \left[t_b, t_c\right), \\ 0, & t \notin \left[t_a, t_c\right), \end{cases} \tag{3.2}$$

where $k = 2^n + j$, integer $n \geq 1$ is a dilation parameter, integer $0 \leq j < 2^n$ is a shift parameter, and the intervals are given by $t_a = m/2^n$, $t_b = (0.5 + j)/2^n$, and $t_c = (1 + j)/2^n$. Since the support of the Haar wavelet is $[0, 1)$, any square integrable function $y(t) \in L_2[0, 1)$ can be written as an infinite linear combination of Haar functions

$$y(t) = \sum_{k=0}^{\infty} c_k \varphi_k(t), \quad t \in [0, 1), \tag{3.3}$$

where the Haar coefficients are determined by

$$c_k = \left\langle y(t), \varphi_k(t) \right\rangle = 2^n \int_0^1 y(t) \varphi_k(t) dt, \tag{3.4}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. In practical applications, Haar series are truncated to m terms, that is,

$$y(t) \cong \sum_{k=0}^{m-1} c_k \varphi_k(t) = \mathbf{C}_m^{\mathrm{T}} \mathbf{h}_m(t), \tag{3.5}$$

where Haar functions coefficient vector $\mathbf{C}_m$ and Haar functions vector $\mathbf{h}_m$ are defined as $\mathbf{C}_m \triangleq \left[c_0 \ \ c_1 \ \ \cdots \ \ c_{m-1}\right]^{\mathrm{T}}$ and $\mathbf{h}_m(t) \triangleq \left[\varphi_0(t) \ \ \varphi_1(t) \ \ \cdots \ \ \varphi_{m-1}(t)\right]^{\mathrm{T}}$.

Integrals of the Haar functions with respect to variable t form ramp and triangular waveforms standing with uniform slope, respectively, at the positions of the corresponding rectangular functions. The group of these integrals can be expressed as follows:

$$\int_0^1 \varphi_0(t)dt = t, \quad t \in [t_a, t_b),$$
$$\int_0^1 \varphi_k(t)dt = \begin{cases} t - t_a, & t \in [t_a, t_b), \\ -t + t_c, & t \in [t_b, t_c), \\ 0, & t \notin [t_a, t_c). \end{cases} \tag{3.6}$$

Then, the Haar matrix $\mathbf{H}_m$ is defined as

$$\mathbf{H}_m(t) \triangleq \left[\mathbf{h}_m(t_0) \;\; \mathbf{h}_m(t_1) \;\; \cdots \;\; \mathbf{h}_m(t_{m-1})\right], \tag{3.7}$$

where $i/m \le t_i \le (i+1)/m$.

Integration of the Haar function vector can be written as

$$\int_0^t \mathbf{h}_m(\tau)d\tau \cong \mathbf{P}_m\mathbf{h}_m(t), \tag{3.8}$$

where $\mathbf{P}_m$ is the m-square operational matrix of integration which satisfies the following recursive formula [3]:

$$\mathbf{P}_m = \begin{bmatrix} \mathbf{P}_{m/2} & -\dfrac{1}{2m}\mathbf{H}_{m/2} \\ \dfrac{1}{2m}\mathbf{H}_{m/2}^{-1} & \mathbf{0}_{m/2} \end{bmatrix}, \qquad \mathbf{P}_1 = \left[\frac{1}{2}\right], \tag{3.9}$$

where $\mathbf{0}_{m/2}$ is an $m/2$-square zero matrix. The Haar matrix $\mathbf{H}_m$ also has the following recursive formula [3]:

$$\mathbf{H}_m = \begin{bmatrix} \mathbf{H}_{m/2} \otimes [1 \;\; 1] \\ \mathbf{I}_{m/2} \otimes [1 \;\; -1] \end{bmatrix}, \qquad \mathbf{H}_1 = [1]. \tag{3.10}$$

Particularly, it was proven that the following relationship holds [3]:

$$\mathbf{H}_m^{-1} = \frac{1}{m}\mathbf{H}_m^T\mathbf{D}_m, \tag{3.11}$$

where $\mathbf{D}_m = \operatorname{diag}(1\ 1\ 2\ 2 \cdots \underbrace{2^{p-1} \cdots 2^{p-1}}_{m/2})$ and $p = \log_2 m$. This diagonal matrix $\mathbf{D}_m$ also can be represented in the recursive form

$$\mathbf{D}_m = \begin{bmatrix} \mathbf{D}_{m/2} & \mathbf{0}_{m/2} \\ \mathbf{0}_{m/2} & \dfrac{m}{2}\mathbf{I}_{m/2} \end{bmatrix}, \qquad \mathbf{D}_1 = [1], \tag{3.12}$$

where $m = 2^k,\ k = 1, 2, \ldots, J$, and J is called a resolution scale or level.

We present the following lemma which will be used to decompose the generalized Sylvester matrix equation.

Lemma 3.1. *Let $\mathbf{H}_m$ be a Haar matrix defined in* (3.10). *Then, its inverse matrix has the following recursive form:*

$$\mathbf{H}_m^{-1} = \left[\mathbf{H}_{m/2}^{-1} \otimes \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \quad \mathbf{I}_{m/2} \otimes \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix} \right]. \tag{3.13}$$

Proof. We assume that $\mathbf{H}_m^{-1}$ has the following recursive structure:

$$\mathbf{H}_m^{-1} = \left[\mathbf{H}_{m/2}^{-1} \otimes \begin{bmatrix} a \\ b \end{bmatrix} \quad \mathbf{I}_{m/2} \otimes \begin{bmatrix} c \\ d \end{bmatrix} \right], \tag{3.14}$$

where $a,\ b,\ c,\ d$ are constants to be determined. Now, we multiply $\mathbf{H}_m$ and $\mathbf{H}_m^{-1}$:

$$\mathbf{H}_m \mathbf{H}_m^{-1} = \begin{bmatrix} \left(\mathbf{H}_{m/2} \otimes [1\ \ 1]\right)\left(\mathbf{H}_{m/2}^{-1} \otimes \begin{bmatrix} a \\ b \end{bmatrix}\right) & \left(\mathbf{H}_{m/2} \otimes [1\ \ 1]\right)\left(\mathbf{I}_{m/2} \otimes \begin{bmatrix} c \\ d \end{bmatrix}\right) \\ \left(\mathbf{I}_{m/2} \otimes [1\ \ -1]\right)\left(\mathbf{H}_{m/2}^{-1} \otimes \begin{bmatrix} a \\ b \end{bmatrix}\right) & \left(\mathbf{I}_{m/2} \otimes [1\ \ -1]\right)\left(\mathbf{I}_{m/2} \otimes \begin{bmatrix} c \\ d \end{bmatrix}\right) \end{bmatrix}. \tag{3.15}$$

Then, using the property of $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}$, we obtain

$$\mathbf{H}_m \mathbf{H}_m^{-1} = \begin{bmatrix} \mathbf{I}_{m/2} \otimes (a+b) & \mathbf{H}_{m/2} \otimes (c+d) \\ \mathbf{H}_{m/2}^{-1} \otimes (a-b) & \mathbf{I}_{m/2} \otimes (c-d) \end{bmatrix}. \tag{3.16}$$

Thus, $a = b = 0.5,\ c = 0.5,\ d = -0.5$ satisfy $\mathbf{H}_m \mathbf{H}_m^{-1} = \mathbf{H}_m^{-1} \mathbf{H}_m = \mathbf{I}$. □

4. Singular linear system

Consider a linear continuous-time singular system described by

$$\mathbf{E}\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \quad \mathbf{x}(0) = \mathbf{x}_0, \tag{4.1}$$

where $\mathbf{x}(t) \in \mathbb{R}^p$ denotes the vector of state variables, $\mathbf{u}(t) \in \mathbb{R}^q$ denotes the vector of manipulated inputs, $\mathbf{E}$, $\mathbf{A}$ are $p \times p$ matrices, $\mathbf{E}$ is generally singular, and $\mathbf{B}$ is a $p \times q$

matrix. Without loss of generality, we assume that $\operatorname{rank}(\mathbf{A}) = p$ and (4.1) is regular, that is, $\det(\lambda\mathbf{E} - \mathbf{A}) \neq 0$. Regularity means that the solution $\mathbf{x}(t)$ is uniquely determined by the given initial value $\mathbf{x}_0$ and input $\mathbf{u}(t)$.

If the input function vector $\mathbf{u}(t)$ is square integrable in the interval $[0,1)$, then it can be represented in a Haar function basis $\mathbf{h}_m(t)$ as

$$\mathbf{u}(t) = \mathbf{G}\mathbf{h}_m(t), \tag{4.2}$$

where $\mathbf{G} \in \mathbb{R}^{q\times m}$ is a Haar coefficient matrix and can be obtained by the method described in Section 3. Likewise, $\dot{\mathbf{x}}(t)$ is expanded in Haar function basis

$$\dot{\mathbf{x}}(t) = \mathbf{V}\mathbf{h}_m(t), \tag{4.3}$$

where $\mathbf{V} \in \mathbb{R}^{p\times m}$ is the unknown matrix to be determined. From the definition of the Haar function, the initial state can be represented as follows:

$$\mathbf{x}_0 = \begin{bmatrix}\mathbf{x}_0 & 0 & \cdots & 0\end{bmatrix}\mathbf{h}_m(t). \tag{4.4}$$

Integrating (4.3) from 0 to t, we have

$$\mathbf{x}(t) = \mathbf{V}\mathbf{P}_m\mathbf{h}_m(t) + \mathbf{x}_0. \tag{4.5}$$

Integrating (4.1) and using (3.8) and (4.4), after canceling $\mathbf{h}_m(t)$, we obtain

$$\mathbf{E}\mathbf{V} - \mathbf{A}\mathbf{V}\mathbf{P}_m = \mathbf{Q}, \tag{4.6}$$

where we define $\mathbf{Q} \triangleq \begin{bmatrix}\mathbf{A}\mathbf{x}_0 & 0 & \cdots & 0\end{bmatrix} + \mathbf{B}\mathbf{G}$. Thus, the differential matrix equation (4.1) has been transformed to a generalized Sylvester matrix equation that must be solved for $\mathbf{V}$. Equation (4.6) can be solved by using Kronecker product as in [6]

$$\left(\mathbf{I}_m \otimes \mathbf{E} + \mathbf{P}_m^{\mathrm{T}} \otimes \mathbf{A}\right)\operatorname{vec}(\mathbf{V}) = \operatorname{vec}(\mathbf{Q}), \tag{4.7}$$

where $\mathbf{I}_m$ is a unit matrix. Equation (4.7) can be solved by LU factorization. However, the coefficient matrix $\mathbf{I}_m \otimes \mathbf{E} + \mathbf{P}_m^{\mathrm{T}} \otimes \mathbf{A}$ has dimension $pm \times pm$, making this approach impractical except for small systems. There are other methods for solving the Sylvester matrix equation (4.6), for example, the Bartels-Stewart algorithm, Krylov subspace method, and matrix sign function method (see [10] and references therein). In [3, 11], recursive algorithms were derived to solve the equations of type $\mathbf{V} - \mathbf{A}\mathbf{V}\mathbf{P}_m = \mathbf{Q}$ for linear systems. It should be noted that the algorithm in [3] is not applicable to a generalized Sylvester matrix equation (4.6), since $\mathbf{E}$ is a singular matrix.

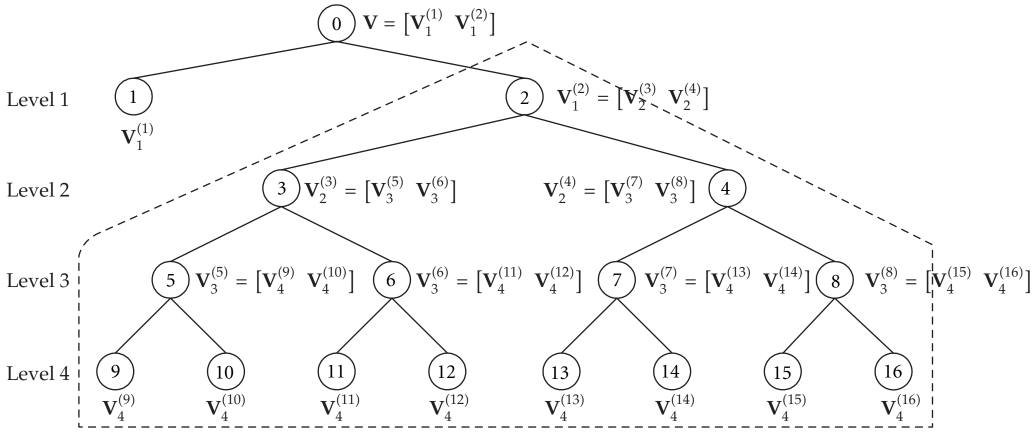

Figure 1: Binary tree for resolution scale $J = 4$.

4.1. Decomposition and recursive binary tree

Under the assumption that $\mathbf{A}$ is a nonsingular matrix, (4.6) can be written as the following Sylvester equation:

$$\mathbf{A}^{-1}\mathbf{E}\mathbf{V} - \mathbf{V}\mathbf{P}_s = \mathbf{A}^{-1}\mathbf{Q}. \tag{4.8}$$

To decompose (4.8), we split $\mathbf{V}$ and $\mathbf{A}^{-1}\mathbf{Q}$ by columns:

$$\mathbf{A}_E\left[\mathbf{V}_1^{(1)}\ \ \mathbf{V}_1^{(2)}\right] - \left[\mathbf{V}_1^{(1)}\ \ \mathbf{V}_1^{(2)}\right]\begin{bmatrix} \mathbf{P}_{m/2} & -\dfrac{1}{2m}\mathbf{H}_{m/2} \\ \dfrac{1}{2m}\mathbf{H}_{m/2}^{-1} & \mathbf{0}_{m/2} \end{bmatrix} = \left[\mathbf{Q}_a\ \ \mathbf{Q}_b\right], \tag{4.9}$$

where $\mathbf{A}_E \triangleq \mathbf{A}^{-1}\mathbf{E}$, $\mathbf{A}^{-1}\mathbf{Q} = \left[\mathbf{Q}_a\ \ \mathbf{Q}_b\right]$, $\mathbf{V} = \left[\mathbf{V}_1^{(1)}\ \ \mathbf{V}_1^{(2)}\right]$ with $\mathbf{Q}_a, \mathbf{Q}_b, \mathbf{V}_1^{(1)}, \mathbf{V}_1^{(2)} \in R^{p\times m/2}$. Here $\mathbf{V}_k^{(r)}$ denotes the matrix that is decomposed at level k with $r = \{2^k, 2^k - 1\}$. Then, we obtain the following reduced-order matrix equations:

$$\mathbf{A}_E\mathbf{V}_1^{(1)} - \mathbf{V}_1^{(1)}\mathbf{P}_{m/2} - \frac{1}{2m}\mathbf{V}_1^{(2)}\mathbf{H}_{m/2}^{-1} = \mathbf{Q}_a. \tag{4.10}$$

$$\mathbf{A}_E\mathbf{V}_1^{(2)} + \frac{1}{2m}\mathbf{V}_1^{(1)}\mathbf{H}_{m/2} = \mathbf{Q}_b. \tag{4.11}$$

Since $\mathbf{E}$ is a singular matrix, $\mathbf{A}_E$ is also singular. Thus, we postmultiply by $\mathbf{H}_{m/2}^{-1}$ both sides of (4.11) to express $\mathbf{V}_1^{(1)}$ in terms of $\mathbf{V}_1^{(2)}$

$$\mathbf{V}_1^{(1)} = -2m\mathbf{A}_E\mathbf{V}_1^{(2)}\mathbf{H}_{m/2}^{-1} + 2m\mathbf{Q}_b\mathbf{H}_{m/2}^{-1}. \tag{4.12}$$

Substituting (4.12) into (4.10) yields

$$\begin{aligned}&-2m\mathbf{A}_E^2\mathbf{V}_1^{(2)}\mathbf{H}_{m/2}^{-1}+2m\mathbf{A}_E\mathbf{Q}_b\mathbf{H}_{m/2}^{-1}+2m\mathbf{A}_E\mathbf{V}_1^{(2)}\mathbf{H}_{m/2}^{-1}\mathbf{P}_{m/2}\\&\quad-2m\mathbf{Q}_b\mathbf{H}_{m/2}^{-1}\mathbf{P}_{m/2}-\frac{1}{2m}\mathbf{V}_1^{(2)}\mathbf{H}_{m/2}^{-1}=\mathbf{Q}_a.\end{aligned}\tag{4.13}$$

Therefore, the original problem is decomposed into a reduced-order generalized Sylvester matrix equation (4.13) and a matrix algebraic equation (4.12). Again postmultiplying by $\mathbf{H}_{m/2}$ both sides of (4.13), we have

$$\begin{aligned}&\left(-2m\mathbf{A}_E^2-\frac{1}{2m}\mathbf{I}\right)\mathbf{V}_1^{(2)}+2m\mathbf{A}_E\mathbf{V}_1^{(2)}\mathbf{H}_{m/2}^{-1}\mathbf{P}_{m/2}\mathbf{H}_{m/2}\\&\quad=\mathbf{Q}_a\mathbf{H}_{m/2}-2m\mathbf{A}_E\mathbf{Q}_b+2m\mathbf{Q}_b\mathbf{H}_{m/2}^{-1}\mathbf{P}_{m/2}\mathbf{H}_{m/2}.\end{aligned}\tag{4.14}$$

In (4.14), we define

$$\mathbf{C}_{m/2}\triangleq\mathbf{H}_{m/2}^{-1}\mathbf{P}_{m/2}\mathbf{H}_{m/2}.\tag{4.15}$$

Then, the matrix $\mathbf{C}_{m/2}$ is an upper triangular matrix and has the following recursive form:

$$\mathbf{C}_{m/2}=\begin{bmatrix}\frac{1}{2}\mathbf{C}_{m/4}&\frac{2}{m}\mathbf{1}_{m/4}\\\mathbf{0}_{m/4}&\frac{1}{2}\mathbf{C}_{m/4}\end{bmatrix},\qquad\mathbf{C}_1=\begin{bmatrix}\frac{1}{2}\end{bmatrix},\tag{4.16}$$

where $\mathbf{1}_{m/4}$ denotes $m/4$-square matrix with all elements being 1 (see Appendix A).

Substituting (4.16) into (4.14) and splitting $\mathbf{V}_1^{(2)}$ and the right-hand side of (4.14) by columns yields

$$\mathbf{A}_h\begin{bmatrix}\mathbf{V}_2^{(3)}&\mathbf{V}_2^{(4)}\end{bmatrix}+2m\mathbf{A}_E\begin{bmatrix}\mathbf{V}_2^{(3)}&\mathbf{V}_2^{(4)}\end{bmatrix}\begin{bmatrix}\frac{1}{2}\mathbf{C}_{m/4}&\frac{2}{m}\mathbf{1}_{m/4}\\\mathbf{0}_{m/4}&\frac{1}{2}\mathbf{C}_{m/4}\end{bmatrix}=\begin{bmatrix}\mathbf{T}_2^{(3)}&\mathbf{T}_2^{(4)}\end{bmatrix},\tag{4.17}$$

where

$$\begin{aligned}\mathbf{A}_h&\triangleq\left(-2m\mathbf{A}_E^2-\frac{1}{2m}\mathbf{I}\right),\\\mathbf{Q}_a\mathbf{H}_{m/2}-2m\mathbf{A}_E\mathbf{Q}_b+2m\mathbf{Q}_b\mathbf{C}_{m/2}&\triangleq\mathbf{T}_1^{(2)}=\begin{bmatrix}\mathbf{T}_2^{(3)}&\mathbf{T}_2^{(4)}\end{bmatrix},\\\mathbf{V}_1^{(2)}&=\begin{bmatrix}\mathbf{V}_2^{(3)}&\mathbf{V}_2^{(4)}\end{bmatrix}.\end{aligned}\tag{4.18}$$

Thus, (4.17) is decomposed into two matrix equations with dependent and independent subsystems.

$$\mathbf{A}_h\mathbf{V}_2^{(3)} + m\mathbf{A}_E\mathbf{V}_2^{(3)}\mathbf{C}_{m/4} = \mathbf{T}_2^{(3)}. \tag{4.19}$$

$$\mathbf{A}_h\mathbf{V}_2^{(4)} + m\mathbf{A}_E\mathbf{V}_2^{(4)}\mathbf{C}_{m/4} = \mathbf{T}_2^{(4)} - 4\mathbf{A}_E\mathbf{V}_2^{(3)}\mathbf{1}_{m/4}. \tag{4.20}$$

In (4.19) and (4.20), we first solve for $\mathbf{V}_2^{(3)}$ and then after updating the right-hand side of (4.20) with respect to $\mathbf{V}_2^{(3)}$, solve for $\mathbf{V}_2^{(4)}$. Since (4.19) and (4.20) have the same form as (4.17) and $\mathbf{C}_{m/4}$ is still an upper triangular matrix, they can be decomposed into two subsystems in which the dimension has been reduced by half, respectively. Therefore, we recursively decompose each equation into two equations until no further decomposition is possible in which all $\mathbf{V}_J^{(r)}$, $\mathbf{T}_J^{(r)}$ $(r = 2^{J-1}+1,\ldots,2^J)$ are column vectors. This procedure constructs the binary tree as shown in Figure 1.

A binary tree is a rooted tree in which each node has at most two children, designated as a left child and a right child. A full binary tree is a binary tree in which each node has exactly two children or none. A perfect (or complete) binary tree is a full binary tree in which all leaves have the same depth [12]. In Figure 1, the binary tree in the dotted box is a perfect binary tree of depth $J-1$. An external node (or leaf node) is a node with no children. For instance, the nodes labeled 1, 9, 10, 11, 12, 13, 14, 15, and 16 in Figure 1 are external nodes.

Matrix equations corresponding to all external nodes of the perfect binary tree are classified into two types of equations described as follows:

$$\begin{aligned}\mathbf{A}_h\mathbf{V}_J^{(r)} + 4\mathbf{A}_E\mathbf{V}_J^{(r)}\mathbf{C}_1 &= \mathbf{T}_J^{(r)}, \quad r = 2^{J-1}+1,\ 2^{J-1}+3,\ldots,2^J-1 \ (r \text{ is odd}),\\ \mathbf{A}_h\mathbf{V}_J^{(r)} + 4\mathbf{A}_E\mathbf{V}_J^{(r)}\mathbf{C}_1 &= \mathbf{T}_J^{(r)} - 4\mathbf{A}_E\mathbf{V}_J^{(r-1)}\mathbf{1}_1, \quad r = 2^{J-1}+2,\ 2^{J-1}+4,\ldots,2^J \ (r \text{ is even}).\end{aligned} \tag{4.21}$$

Note that in equation (4.21), $\mathbf{C}_1 = 1/2$, $\mathbf{1}_1 = 1$. Thus, they become simple linear matrix equations as follows:

$$\begin{aligned}\left(\mathbf{A}_h + 2\mathbf{A}_E\right)\mathbf{V}_J^{(r)} &= \mathbf{T}_J^{(r)}, \quad \text{if } r \text{ is odd},\\ \left(\mathbf{A}_h + 2\mathbf{A}_E\right)\mathbf{V}_J^{(r)} &= \mathbf{T}_J^{(r)} - 4\mathbf{A}_E\mathbf{V}_J^{(r-1)}, \quad \text{if } r \text{ is even}.\end{aligned} \tag{4.22}$$

4.2. Combined preorder and postorder traversal algorithm

Visiting all the nodes in a tree in some particular order is known as a tree traversal. A preorder traversal visits the root of a subtree, then the left and right subtrees recursively. A postorder traversal visits the left and right subtrees recursively, then the root node of the subtree [12]. For example, the preorder and postorder traversals of the binary tree shown in Figure 1 are as follows:

Preorder traversal: 0 → 1 → 2 → 3 → 5 → 9 → 10 → 6 → 7 → 12 → 4 → 7 → 13 → 14 → 8 → 15 → 16

Postorder traversal: 1 → 9 → 10 → 5 → 11 → 12 → 6 → 3 → 13 → 14 → 7 → 15 → 16 → 8 → 4 → 2 → 0

```
Step 1. Initialize A_h, A_E, T.
Step 2. Obtain V_1^(2)
Input: Resolution scale J
WaveSolver(J)
{
  for (r = 2^(J-1) + 1; r < 2^J; r = r + 2)
  {
    Solve for V_J^(r) the system (A_h + 2A_E)V_J^(r) = T_J^(r)
    Update T_J^(r+1) accroding to T_J^(r+1) = T_J^(r+1) - 4A_E V_J^(r)
    Solve for V_J^(r+1) the system (A_h + 2A_E)V_J^(r+1) = T_J^(r+1)
    WaveTree (1, J, r + 1)
  }
}
Input: rno is a number of recursive call.
    Resolution scale J
    r is a node number.
WaveTree(rno, J, r)
{
  if (J - rno <= 0)
     return
  Merge: V_{J-rno}^(r/2) = [V_{J-rno+1}^(r-1)  V_{J-rno+1}^(r)]
  if (r/2 is even)
    WaveTree(rno+1, J, r/2);
  else
    Update and Split: [T_{J-rno+1}^(r-1)  T_{J-rno+1}^(r)] = T_{J-rno}^(r/2+1) - 4A_E V_{J-rno}^(r/2) 1_{m/2^(J-rno)}
}
Step 3. Solve V_1^(1) from (4.12).
```

Algorithm 1

During the decomposition of (4.14), the right-hand side of the right child is split after updating it recursively as follows:

$$\mathbf{T}_k^{(r)} - 4\mathbf{A}_E\mathbf{V}_k^{(r-1)}\mathbf{1}_{m/2^k} = \left[\mathbf{T}_{k+1}^{(2r-1)} \quad \mathbf{T}_{k+1}^{(2r)}\right]. \tag{4.23}$$

This splitting and updating sequence is a preorder traversal of the perfect binary tree from root node ②. The unknown matrix $\mathbf{V}_1^{(2)}$ is obtained by merging all column vectors $\mathbf{V}_J^{(r)}$ ($r = 2^{J-1}+1,\ldots,2^J$). This sequence is a postorder traversal of the perfect binary tree from root node ②. To update (4.23), we need $\mathbf{V}_k^{(r-1)}$ which is obtained from the left child. Hence, to solve (4.22), it is necessary to update, split, and solve by using the following combined preorder and postorder traversal method.

The pseudocode of the proposed algorithm is as in Algorithm 1.

For example, at resolution scale $J = 4$, the proposed combined preorder and postorder traversal method is illustrated in Figure 2.

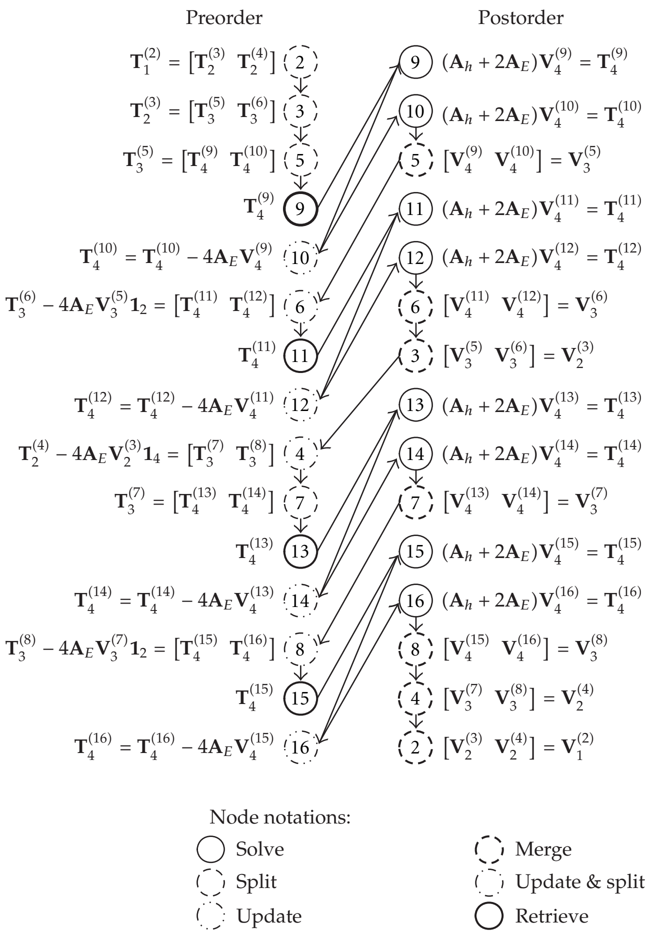

Figure 2: The combined preorder and postorder traversal for resolution scale $J = 4$.

In Figure 2, nodes 2, 3, 5, and 9 of preorder traversal are done at Step 1 and the remaining nodes are processed at Step 2. The computational efficiency of the proposed method is discussed in the next section.

5. An illustrated example

In this section, an example is presented to illustrate the proposed algorithm. We consider a singular linear system of (4.1) with

$$\mathbf{E} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{A} = \begin{bmatrix} -33 & 0 & 1.0 & 0 \\ 0 & 1 & 0 & 1.0 \\ 0 & 621.4 & -28.27 & 0 \\ 0 & -327.1 & 12.72 & 1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 52.65 \\ -23.69 \end{bmatrix}, \tag{5.1}$$

and $\mathbf{X}_0 = \begin{bmatrix} 0 & 0.5 & 1.0 & 0 \end{bmatrix}^{\mathrm{T}}$. And we assume that $u(t)$ is a unit step function. In the cases of $J = 4$ and 8, the simulation results are depicted in Figures 3 and 4, respectively.

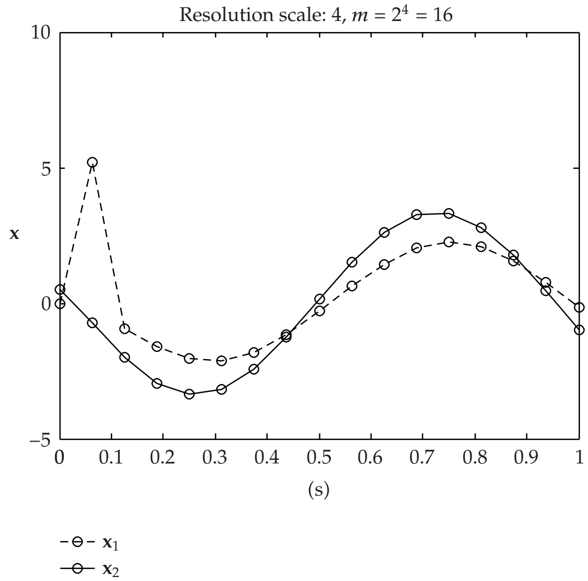

Figure 3: Case for resolution scale $J = 4$.

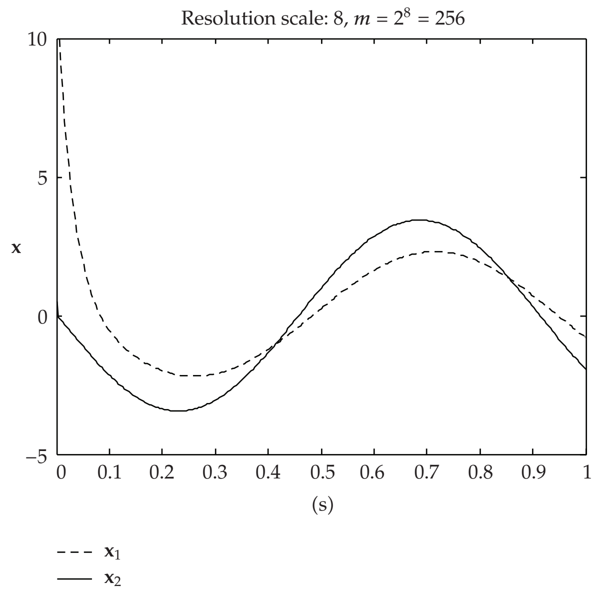

Figure 4: Case for resolution scale $J = 8$.

From these figures, it is clear that the solution accuracy is improved when the resolution scale is increased. However, it requires more computational time.

In (4.7), the LU factorization of $\mathbf{I}_m \otimes \mathbf{E} + \mathbf{P}_m^{\mathrm{T}} \otimes \mathbf{A}$ involves $O(m^3p^3)$ flops. The cost of the proposed algorithm is the sum of the cost of *WinSolver*, $O((m/2)p^3 + (m/2)p^2)$, and the cost of *WinTree*, $O(\sum_{k=1}^{J-2}(2^{J-1}p^2 + (2^{J+2k-2} - 2^{J-2})p))$ (see Appendix B). Since $m = 2^J$, the costs

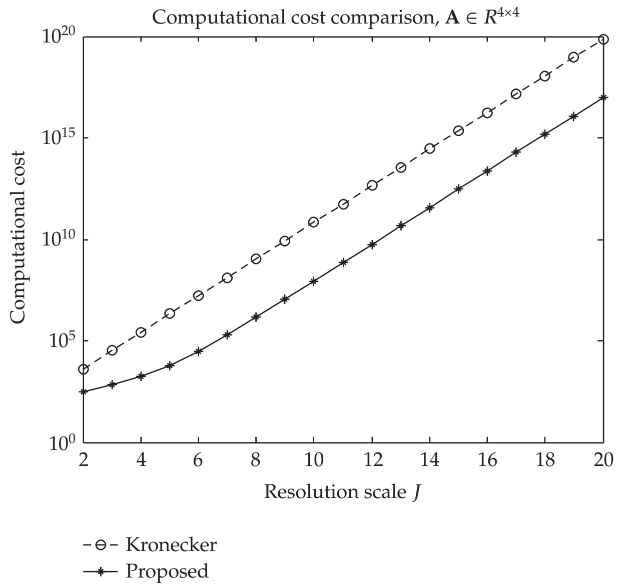

Figure 5: Log plot of flop counts for the Kronecker product method and the proposed method.

Table 1: Flop counts for various sizes of the matrix $\mathbf{A}$ and resolution scales.

J	$\mathbf{A} \in R^{4\times4}$				$\mathbf{A} \in R^{20\times20}$			
	Kronecker method	Proposed algorithm			Kronecker method	Proposed algorithm		
		WinSolver	*WinTree*	Total		*WinSolver*	*WinTree*	Total
2	4096	160	0	160	512000	16800	0	16800
5	2097152	1280	3360	4640	262144000	134400	32160	166560
10	6.871×10^{10}	40960	89534464	89575424	8.589×10^{12}	4300800	448983040	453283840
12	4.398×10^{12}	163840	5.726×10^{9}	5.727×10^{9}	5.497×10^{14}	17203200	2.864×10^{10}	2.867×10^{10}
15	2.251×10^{15}	1310720	2.932×10^{12}	2.932×10^{12}	2.814×10^{17}	137625600	1.466×10^{13}	1.466×10^{13}
18	1.152×10^{18}	10485760	1.501×10^{15}	1.501×10^{15}	1.441×10^{20}	1101004800	7.506×10^{15}	7.506×10^{15}
20	7.378×10^{19}	83886080	4.194×10^{16}	4.194×10^{16}	9.223×10^{21}	4.404×10^{9}	4.803×10^{17}	4.803×10^{17}

of *WinSolver* and $O(m^3p^3)$ can be rewritten as $O(2^{J-1}p^3 + 2^{J-1}p^2)$ and $O(2^{3J}p^3)$, respectively. Thus, the total cost of the proposed algorithm is

$$O\left(2^{J-1}p^3 + 2^{J-1}p^2 + \sum_{k=1}^{J-2}\left(2^{J-1}p^2 + \left(2^{J+2k-2} - 2^{J-2}\right)p\right)\right) \text{ flops.} \tag{5.2}$$

Table 1 and Figure 5 show that the computational cost of the proposed algorithm is significantly less than the Kronecker product method, and that the flop counts are increasing rapidly with resolution scale. As the resolution scale grows, the flop counts of *WinTree* is increasing more rapidly than that of *WinSolver* since the sizes of matrices $\mathbf{1}_m$, $\mathbf{T}_m$, and $\mathbf{V}_m$ increase exponentially.

Table 2

Level	Size of $\mathbf{1}_{m/2^{J-rno}}$	Size of $\mathbf{T}_{J-rno}^{(r/2+1)}$ and $\mathbf{V}_{J-rno}^{(r/2)}$	Times	Computational cost
2	$2^{J-2}\times 2^{J-2}$	$p\times 2^{J-2}$	1	$p2^{2J-3}+(p^2+p)2^{J-1}-p2^{J-1}$
3	$2^{J-3}\times 2^{J-3}$	$p\times 2^{J-3}$	2	$p2^{2J-5}+(p^2+p)2^{J-2}-p2^{J-2}$
:	:	:	:	:
$J-k$	$2^k\times 2^k$	$p\times 2^k$	2^{J-k-2}	$2^{k+1}p^2+2^k(2^{2k}-1)p\times 2^{J-k-2}$
:	:	:	:	:
$J-2$	4×4	$p\times 4$	2^{J-4}	$2^3p^2+2^2(2^4-1)p\times 2^{J-4}$
$J-1$	2×2	$p\times 2$	2^{J-3}	$4p^2+2^1(2^2-1)p\times 2^{J-3}$

6. Conclusions

An efficient computational method was presented for state space analysis of singular systems via Haar wavelets. The problem was formulated as a generalized Sylvester matrix equation. We presented an explicit expression for the inverse of the Haar matrix and a combined preorder and postorder traversal algorithm to solve the problem more effectively. The full-order generalized Sylvester matrix equation was solved in terms of the solutions of simple linear matrix equations by the proposed algorithm. The efficiency of the proposed method was demonstrated by a numerical example.

Appendices

A. Formula for $\mathbf{C}_m$

In this appendix, we derive a formula for $\mathbf{C}_m$. By using (3.13), (3.9), and (3.10), we can write

$$\begin{aligned}
\mathbf{C}_m &= \mathbf{H}_m^{-1}\mathbf{P}_m\mathbf{H}_m \\
&= \left[\mathbf{H}_{m/2}^{-1}\otimes\begin{bmatrix}0.5\\0.5\end{bmatrix} \quad \mathbf{I}_{m/2}\otimes\begin{bmatrix}0.5\\-0.5\end{bmatrix}\right]\begin{bmatrix}\mathbf{P}_{m/2} & -\dfrac{1}{2m}\mathbf{H}_{m/2}\\ \dfrac{1}{2m}\mathbf{H}_{m/2}^{-1} & \mathbf{0}_{m/2}\end{bmatrix}\begin{bmatrix}\mathbf{H}_{m/2}\otimes[1\ \ 1]\\ \mathbf{I}_{m/2}\otimes[1\ \ -1]\end{bmatrix} \\
&= \left[\left(\mathbf{H}_{m/2}^{-1}\otimes\begin{bmatrix}0.5\\0.5\end{bmatrix}\right)\left[\mathbf{P}_{m/2} \quad -\frac{1}{2m}\mathbf{H}_{m/2}\right] + \left(\mathbf{I}_{m/2}\otimes\begin{bmatrix}0.5\\-0.5\end{bmatrix}\right)\left[\frac{1}{2m}\mathbf{H}_{m/2}^{-1} \quad \mathbf{0}_{m/2}\right]\right] \\
&\quad\times\begin{bmatrix}\mathbf{H}_{m/2}\otimes[1\ \ 1]\\ \mathbf{I}_{m/2}\otimes[1\ \ -1]\end{bmatrix} \\
&= \left(\left[\left(\mathbf{H}_{m/2}^{-1}\otimes\begin{bmatrix}0.5\\0.5\end{bmatrix}\right)\mathbf{P}_{m/2} \quad -\frac{1}{2m}\left(\mathbf{H}_{m/2}^{-1}\otimes\begin{bmatrix}0.5\\0.5\end{bmatrix}\right)\mathbf{H}_{m/2}\right]\right. \\
&\qquad\left. + \left[\frac{1}{2m}\mathbf{I}_{m/2}\otimes\begin{bmatrix}0.5\\-0.5\end{bmatrix}\mathbf{H}_{m/2}^{-1} \quad \mathbf{0}_{m/2}\right]\right)\begin{bmatrix}\mathbf{H}_{m/2}\otimes[1\ \ 1]\\ \mathbf{I}_{m/2}\otimes[1\ \ -1]\end{bmatrix} \\
&= \left[\left(\mathbf{H}_{m/2}^{-1}\otimes\begin{bmatrix}0.5\\0.5\end{bmatrix}\right)\mathbf{P}_{m/2} \quad -\frac{1}{2m}\left(\mathbf{H}_{m/2}^{-1}\otimes\begin{bmatrix}0.5\\0.5\end{bmatrix}\right)\mathbf{H}_{m/2}\right]\begin{bmatrix}\mathbf{H}_{m/2}\otimes[1\ \ 1]\\ \mathbf{I}_{m/2}\otimes[1\ \ -1]\end{bmatrix} \\
&\quad + \left[\frac{1}{2m}\mathbf{I}_{m/2}\otimes\begin{bmatrix}0.5\\-0.5\end{bmatrix}\mathbf{H}_{m/2}^{-1} \quad \mathbf{0}_{m/2}\right]\begin{bmatrix}\mathbf{H}_{m/2}\otimes[1\ \ 1]\\ \mathbf{I}_{m/2}\otimes[1\ \ -1]\end{bmatrix}
\end{aligned}$$

$$= \left(\mathbf{H}_{m/2}^{-1} \otimes \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\right) \mathbf{P}_{m/2}\left(\mathbf{H}_{m/2} \otimes \begin{bmatrix} 1 & 1 \end{bmatrix}\right) - \frac{1}{2m}\left(\mathbf{H}_{m/2}^{-1} \otimes \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\right) \mathbf{H}_{m/2}\left(\mathbf{I}_{m/2} \otimes \begin{bmatrix} 1 & -1 \end{bmatrix}\right)$$
$$+ \frac{1}{2m}\mathbf{I}_{m/2} \otimes \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix} \mathbf{H}_{m/2}^{-1}\left(\mathbf{H}_{m/2} \otimes \begin{bmatrix} 1 & 1 \end{bmatrix}\right). \tag{A.1}$$

Since $(\mathbf{A} \otimes \mathbf{B})\mathbf{C} = (\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes 1) = (\mathbf{AC} \otimes \mathbf{B})$ and $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC} \otimes \mathbf{BD})$, the above equation is rewritten as

$$\begin{aligned}
\mathbf{C}_m &= \left((\mathbf{H}_{m/2}^{-1}\mathbf{P}_{m/2}) \otimes \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\right)\left(\mathbf{H}_{m/2} \otimes \begin{bmatrix} 1 & 1 \end{bmatrix}\right) - \frac{1}{2m}\left((\mathbf{H}_{m/2}^{-1}\mathbf{H}_{m/2}) \otimes \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\right)\left(\mathbf{I}_{m/2} \otimes \begin{bmatrix} 1 & -1 \end{bmatrix}\right) \\
&\quad + \frac{1}{2m}\left((\mathbf{I}_{m/2}\mathbf{H}_{m/2}^{-1}) \otimes \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix}\right)\left(\mathbf{H}_{m/2} \otimes \begin{bmatrix} 1 & 1 \end{bmatrix}\right) \\
&= (\mathbf{H}_{m/2}^{-1}\mathbf{P}_{m/2}\mathbf{H}_{m/2}) \otimes \left(\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\begin{bmatrix} 1 & 1 \end{bmatrix}\right) - \frac{1}{2m}(\mathbf{I}_{m/2}) \otimes \left(\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}\begin{bmatrix} 1 & -1 \end{bmatrix}\right) \\
&\quad + \frac{1}{2m}(\mathbf{I}_{m/2}) \otimes \left(\begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix}\begin{bmatrix} 1 & 1 \end{bmatrix}\right) \\
&= \mathbf{C}_{m/2} \otimes \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} + \frac{1}{2m}\mathbf{I}_{m/2} \otimes \left(\begin{bmatrix} -0.5 & 0.5 \\ -0.5 & 0.5 \end{bmatrix} + \begin{bmatrix} 0.5 & 0.5 \\ -0.5 & -0.5 \end{bmatrix}\right) \\
&= \mathbf{C}_{m/2} \otimes \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{bmatrix} + \frac{1}{2m}\mathbf{I}_{m/2} \otimes \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \\
&= \begin{bmatrix} \frac{1}{2}\mathbf{C}_{m/2} & \frac{1}{m}\mathbf{1}_{m/2} \\ \mathbf{0}_{m/2} & \frac{1}{2}\mathbf{C}_{m/2} \end{bmatrix}.
\end{aligned} \tag{A.2}$$

B. Flop counts of the combined preorder and postorder traversal algorithm

In this appendix, we show that the computational cost for the combined preorder and postorder traversal algorithm described in Section 4.2 can be obtained as follows:

(1) WinSolve

Solve for $\mathbf{V}_J^{(r)}$ the system $(\mathbf{A}_h + 2\mathbf{A}_E)\mathbf{V}_J^{(r)} = \mathbf{T}_J^{(r)}$: $O(p^3)$.

Update $\mathbf{T}_J^{(r+1)}$ according to $\mathbf{T}_J^{(r+1)} = \mathbf{T}_J^{(r+1)} - 4\mathbf{A}_E\mathbf{V}_J^{(r)}$: $O(p(2p-1)+p) = O(2p^2)$.

Solve for $\mathbf{V}_J^{(r+1)}$ the system $(\mathbf{A}_h + 2\mathbf{A}_E)\mathbf{V}_J^{(r+1)} = \mathbf{T}_J^{(r+1)}$: $O(p^3)$.

The total iteration number of "for ($r = 2^{J-1} + 1;\ r < 2^J;\ r = r + 2$)" is $m/4$. Thus, *WinSolve* involves $O((m/4)(p^3+2p^2+p^3)) = O((m/2)(p^3+\mathrm{p}^2))$ flops.

(2) WinTree

`Update and split` $\mathbf{T}_{J-\mathrm{rno}}^{(r/2+1)} - 4\mathbf{A}_E\mathbf{V}_{J-\mathrm{rno}}^{(r/2)}\mathbf{1}_{m/2^{J-\mathrm{rno}}}$ (see Table 2).

Therefore, the computational cost for *WinTree* can be calculated by

$$O\left(\sum_{k=1}^{J-2}\left(2^{k+1}p^2 + 2^k\left(2^{2k}-1\right)p\right) \times 2^{J-k-2}\right) = O\left(\sum_{k=1}^{J-2}\left(2^{J-1}p^2 + \left(2^{J+2k-2} - 2^{J-2}\right)p\right)\right). \tag{A.1}$$

References

[1] C. Cattani, "Haar wavelet-based technique for sharp jumps classification," *Mathematical and Computer Modelling*, vol. 39, no. 2-3, pp. 255–278, 2004.

[2] C. Cattani, "Wavelet approach to stability-of-orbits analysis," *International Applied Mechanics*, vol. 42, no. 6, pp. 721–727, 2006.

[3] C. F. Chen and C. H. Hsiao, "Haar wavelet method for solving lumped and distributed-parameter systems," *IEE Proceedings: Control Theory and Applications*, vol. 144, no. 1, pp. 87–94, 1997.

[4] C. F. Chen and C.-H. Hsiao, "Wavelet approach to optimising dynamic systems," *IEE Proceedings: Control Theory and Applications*, vol. 146, no. 2, pp. 213–219, 1999.

[5] F. L. Lewis, "A survey of linear singular systems," *Circuits, Systems, and Signal Processing*, vol. 5, no. 1, pp. 3–36, 1986.

[6] F. L. Lewis and B. G. Mertzios, "Analysis of singular systems using orthogonal functions," *IEEE Transactions on Automatic Control*, vol. 32, no. 6, pp. 527–530, 1987.

[7] R. Kalpana and S. R. Balachandar, "Haar wavelet method for the analysis of transistor circuits," *AEU - International Journal of Electronics and Communications*, vol. 61, no. 9, pp. 589–594, 2007.

[8] C. F. van Loan, "The ubiquitous Kronecker product," *Journal of Computational and Applied Mathematics*, vol. 123, no. 1-2, pp. 85–100, 2000.

[9] A. Haar, "Zur Theorie der orthogonalen Funktionensysteme," *Mathematische Annalen*, vol. 69, no. 3, pp. 331–371, 1910.

[10] Z. Gajic and M. T. J. Qureshi, *Lyapunov Matrix Equation in System Stability and Control*, vol. 195 of *Mathematics in Science and Engineering*, Academic Press, San Diego, Calif, USA, 1995.

[11] B.-S. Kim, I.-J. Shim, B. K. Choi, and J. H. Jeong, "Wavelet based control for linear systems via reduced order Sylvester equation," in *Proceedings of the 3rd International Conference on Cooling and Heating Technologies (ICCHT '07)*, pp. 239–244, Tokyo, Japan, July 2007.

[12] F. Carrano and W. Savitch, *Data Structures and Abstractions with Java*, Prentice Hall, Upper Saddle River, NJ, USA, 2003.