HAAR WAVELET APPROACH TO
ORDINARY DIFFERENTIAL EQUATIONS

A Thesis
Presented to the
Faculty of
California State Polytechnic University, Pomona

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science
In
Mathematics

By
Robert Chen
2016
THESIS:       HAAR WAVELET APPROACH TO
             ORDINARY DIFFERENTIAL EQUATIONS

AUTHOR:      Robert Chen

DATE SUBMITTED:   Spring 2016

Department of Mathematics and Statistics

Dr. Ryan Szypowski
Thesis Committee Chair
Mathematics & Statistics

Dr. Hubertus von Bremen
Mathematics & Statistics

Dr. John Rock
Mathematics & Statistics
ACKNOWLEDGMENTS

Thank you Ryan and all my colleagues from Cal Poly Pomona for your support!

This thesis is dedicated to my dearest grandmas.

Hallelujah

感謝Ryan和加州理工大學波莫那分校各位同事的支持！

此論文獻給我最親愛的兩位奶奶。

哈利路亞
ABSTRACT

The Haar wavelet was one of the first wavelet bases that was developed by Alfred Haar in 1909. Haar used these functions to give an example of an orthonormal system for the space of square-integrable functions on the unit interval $[0,1]$. We will further study his method by applying Haar wavelets as a numerical method to solve different types of ordinary differential equation, such as a higher order ODE and stiff ODE. Moreover, by comparing Haar’s method to other numerical method such as Heun’s method, RK-45, and Implicit Euler’s method, we can determine how well the Haar’s method works in solving ODE.
# Contents

1 Overview .................................................. 1  
   1.1 Introduction ........................................ 1  
   1.2 Popular Applications .............................. 1  

2 Wavelets ................................................. 3  
   2.1 Introduction to Wavelets ......................... 3  
   2.2 Orthogonality ...................................... 3  
   2.3 What are Wavelets? ............................... 4  
   2.4 Haar Wavelet Basis ............................... 5  

3 Haar Wavelet and Integration ....................... 10  
   3.1 Discrete Haar Wavelets ......................... 10  
   3.2 Integration of Haar Wavelet ................... 14  

4 Haar Wavelet Method for Solving Differential Equations 17  
   4.1 Applying Haar wavelets to Differential Equations 17  
   4.2 Numerical Analysis ................................ 18  

5 Conclusion ............................................... 37
List of Figures

2.1 Time vs Frequency ............................... 6
2.2 $h_{00}(x) \rightarrow h_{33}(x)$ ............................. 9
3.1 $H_{00}(t) \rightarrow H_{33}(t)$ ............................. 12
4.1 Haar VS Analytic solution (Example 1: Second order ODE) .................. 20
4.2 Haar VS Analytic: Error (Example 1: Second order ODE) .................. 21
4.3 Haar VS Analytic VS Heun: Solution (Example 1: Second order ODE) 28
4.4 Haar VS Heun: Time, Error (Example 1: Second order ODE) ............... 29
4.5 time vs error: Haar and Heun (Example 1: Second order ODE) ........... 30
4.6 Haar VS ode45 (Example 2: Forth order ODE) .......................... 31
4.7 Error between Haar and ode45 (Example 2: Forth order ODE) .......... 32
4.8 m vs error, time vs error: Heun and Haar (Example 3: Second order stiff
    ODE) .................................................. 33
4.9 Haar vs Implicit Euler vs Analytic (Example 3: Second order stiff ODE) 34
4.10 m vs error, m vs time, Haar and IEM (Example 3: Second order stiff ODE) 35
4.11 Time vs Error: Stiff (Example 3: Second order stiff ODE) ............... 36
List of Tables

4.1 Haar VS Analytic: Error (Example 1: Second order ODE) . . . . . . . 19
4.2 Error between Haar and ode45 (Example 2: Forth order ODE) . . . . 24
4.3 Haar vs Analytic, Haar vs Implicit Euler: Error (Example 3: Second
order stiff ODE) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26
Chapter 1

Overview

1.1 Introduction

Numerical methods for solving Ordinary Differential Equations (ODEs) have been well studied throughout the century, yet people are using different methods for different types of questions. The main factor that determines how good a method is comes down to the calculation time and accuracy. If a method solves ODEs within hundredth, or even thousandth, of a second, but gives tremendous error, one might consider other ways. On the other hand, if a method gives accuracy with error less than $10^{-6}$ but takes a day to calculate, one might consider this method inefficient as well. In the following chapters, we will introduce the Haar Wavelets and apply it as a numerical method to solve different types of ODEs.

1.2 Popular Applications

A wavelet is a mathematical function useful in signal processing and image compression. Wavelets, or the wavelet transformation, convert an image into a series of mathematical...
expressions, usually matrices, which requires less space to store. To view the compressed image, one must decode the file and apply the inverse transform. By applying a discrete Haar wavelet transform, we can use the output to identify the edges in the image. In term of signal processing, wavelets are often used to denoise signals. Though image compression and signal processing are two main application of wavelets, we will discuss wavelets in numerical methods for solving differential equations.
Chapter 2

Wavelets

2.1 Introduction to Wavelets

The Fourier transform is a useful tool when we want to analyze a signal in the frequency domain. However, if we take the Fourier transform over the whole interval, we can not identify specifically at what instant the frequency rises. The short-time Fourier transform uses a sliding window to find spectrogram, which gives information on both time and frequency, but the length of window limits the resolution of frequency. With the wavelet transform, we can change the scale to have it analyze the signal at different scale.

2.2 Orthogonality

Why is orthogonality an important property when constructing a basis? For example, if \( \{v_1, v_2, v_3\} \) is a basis for \( \mathbb{R}^3 \), we can write any \( v \in \mathbb{R}^3 \) as a linear combination of \( v_1, v_2, \) and \( v_3 \) in a unique way; that is \( v = a_1v_1 + a_2v_2 + a_3v_3 \) where \( a_1, a_2, a_3 \in \mathbb{R} \). While we know that \( a_1, a_2, a_3 \) are unique, we do not have a way of finding them without doing some explicit calculations. On the other hand, if \( \{w_1, w_2, w_3\} \) is an orthonormal basis for
\( \mathbb{R}^3 \), we can write any \( v \in \mathbb{R}^3 \) as

\[
v = (v \cdot w_1)w_1 + (v \cdot w_2)w_2 + (v \cdot w_3)w_3.
\]

In this case, we have an explicit formula for the unique coefficients in the linear combination.

The Fourier basis,

\[
b = \{1, \cos(x), \sin(x), \cos(2x), \sin(2x), \cos(3x), \sin(3x), \ldots\}
\]

is a orthogonal basis of \( L^2(-\pi, \pi) \); that is,

\[
\int_{-\pi}^{\pi} f(x)g(x)dx = 0, \text{ for any } f(x), g(x) \in b, \text{ and } f(x) \neq g(x).
\] (2.2)

Thus, we have a simple way to calculate the Fourier coefficients \( a_n \) and \( b_n \),

\[
a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos\left(\frac{n\pi x}{p}\right) dx, \quad n \geq 0,
\]

\[
b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin\left(\frac{n\pi x}{p}\right) dx, \quad n \geq 1.
\] (2.3)

A Fourier series of a function is

\[
s(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} [a_n \cos(nx) + b_n \sin(nx)]
\] (2.4)

In the later part of this chapter, we will construct the Haar basis and see that it is also an orthogonal basis.

### 2.3 What are Wavelets?

Wavelets are defined in terms basis functions, with two characteristics, location and scale. In general, a wavelet basis function is represented by the function

\[
\phi_{a,b}(t) = \frac{1}{\sqrt{a}} \phi \left( \frac{t-b}{a} \right),
\] (2.5)
where \( a \) is the scale parameter that determines the width of the function, and \( b \) is the location parameter that determines where we are localized, or where the support is. The function \( \phi \) is called *mother wavelet*. To understand more about how wavelets works, we will demonstrate with a time-signal cartoon (Fig. 2.1). For a wavelet function with small \( a \), it has a narrower window that gives nice time localization and can capture high frequency. Then we slide that window across time by varying \( b \) to capture all the high frequencies first. After we have done that, we will remove the high frequencies we have captured and make \( a \) larger. When \( a \) is large, it has a wider window and can capture low frequencies, but we lose some of the time localization. We keep making \( a \) larger every time until it captures the whole graph.

The graph on the right of fig. 2.1 gives another elaboration to this. In order to capture high frequency, we will need large time steps. After that, we decrease the number of time steps which captures lower frequencies but less time localization. Then repeat the process until the time step covers the whole signal. We will start by building the Haar wavelets.

2.4 Haar Wavelet Basis

The first orthogonal basis of wavelets was developed by a Hungarian mathematician Alfred Haar. [3] Up until this day, Haar wavelets still remain one of the most used wavelets in the field of signal and image processing. The construction of these functions are relatively simple. In this construction, we assume the interval of interest is \([0, 1]\). Equation (2.6) is the general form of a Haar wavelet, \( i \) denotes the generation of the function, also known as the scaling parameter. Every generation cuts the unit interval into \( 2^{i-1} \) steps. \( j \) denotes the location of the function, where \( i \in \mathbb{N} \) and \( j = 0, 1, \ldots, 2^{i-1} - 1 \).
Figure 2.1: Time vs Frequency
\[ h_{ij}(x) = 2^{(i-1)/2}h_{10}(2^{j-1}x - j), \]  

(2.6)

with

\[ h_{00}(x) = 1, x \in [0, 1], \]

and

\[ h_{i0}(x) = \begin{cases} 1 & \text{for } 0 < x < \frac{1}{2} \\ -1 & \text{for } \frac{1}{2} < x < 1 \\ 0 & \text{otherwise} \end{cases} \]

(2.7)

We can see that \( h_{00} \) and \( h_{10} \) are orthogonal by multiplying them together and integrating from 0 to 1. Integrating from 0 to \( \frac{1}{2} \) we get 1, and from \( \frac{1}{2} \) to 1 we get -1 (Fig. 2.2). To continue building the next wavelet functions, we will compress \( h_{10}(x) \) horizontally and stretch the function vertically,

\[ h_{20}(x) = \sqrt{2}h_{10}(4x) \]

\[ h_{21}(x) = \sqrt{2}h_{10}(2x - 1) \]

(2.8)

(2.9)

We continue to do the same to get wavelet functions with higher resolution (Equation (2.10)).

\[ h_{30}(x) = 2h_{10}(4x) \]

\[ h_{31}(x) = 2h_{10}(4x - 1) \]

\[ h_{32}(x) = 2h_{10}(4x - 2) \]

\[ h_{33}(x) = 2h_{10}(4x - 3) \]

(2.10)
As we keep constructing $h_{ij}$ as $i$ grows big, the interval on which $h_{ij}$ is nonzero, called the support, gets narrower and narrower. At some point, we can throw them away at a minimal loss of information. For example, $h_{6k}$ in equation (2.11) has a step size of $\frac{1}{32}$. If we keep repeating the procedure, $h_{15k}$ would have a step size of $\frac{1}{2^{15}} = \frac{1}{32768}$ at which we may ignore without losing resolution, depending on the application.

\[
h_{4k} = 2\sqrt{2}h_{10}(8x - k) \quad k = 0,1,\ldots,7
\]
\[
h_{5k} = 4h_{10}(16x - k) \quad k = 0,1,2,\ldots15
\]
\[
h_{6k} = 4\sqrt{2}h_{10}(32x - k) \quad k = 0,1,2,\ldots31
\]

By the construction of Haar wavelets, these functions are orthogonal to each other. That is,

\[
<h_{\alpha}(x), h_{\beta}(x)> = \int h_{\alpha}(x)h_{\beta}(x)dx = \delta_{\alpha\beta} = \begin{cases} 0 & \alpha = \beta \\ 1 & \alpha = \beta \end{cases}
\]

(2.12)
Figure 2.2: $h_{00}(x) \rightarrow h_{33}(x)$
Chapter 3

Haar Wavelet and Integration

3.1 Discrete Haar Wavelets

The use of orthogonal functions to construct operational matrices for solving identification and optimization problems of dynamic systems was initially established in 1975, when the Walsh-type operational matrix was constructed by Chen and Hsiao [2]. The main idea of this technique is to convert a differential equation into an algebraic equation, and hence the solution process will be much simplified.

In section 2.4, we have been using $x$ as the independent variable, but since time is the more used variable in terms of dynamic systems, we will use $t$ as our independent variable. The fundamental idea starts from integrating a matrix $\phi(t)$, and we will approximate it using the following,

$$
\int_{0}^{t} \phi(\tau)d\tau \approx P\phi(t)
$$

(3.1)
where

\[
\phi(t) = \begin{bmatrix}
\phi_0(t) \\
\phi_1(t) \\
\vdots \\
\phi_{m-1}(t)
\end{bmatrix}.
\]

\(\phi_i(t)\)'s are the basic orthogonal functions on a certain interval \([a, b]\), which in the case of Haar wavelets, the interval is \([0, 1]\). And matrix \(P\) is uniquely determined based on the orthogonal functions \(\phi_i(t)\).

In order to solve differential equation models of dynamic systems, usually one needs to perform integration to solve the problem. Let us consider the Haar wavelet function from section 2.4, and the integration of these functions (Fig. 3.1). Define \(H_{ij}(t)\) by

\[
H_{ij}(t) = \int_0^t h_{ij}(\tau)d\tau \quad (3.2)
\]

We also know, from earlier, by the construction of Haar wavelets, all of them are orthogonal to each other (Equation (2.12)). Functions with this property are well suited as a transform basis. Given any function \(y(t) \in L^2(0, 1)\), or square integrable function \(y(t)\) over the interval \([0, 1]\), \(y(t)\) can be expanded as a series of Haar wavelets [2]:

\[
y(t) = c_0 h_{00}(t) + c_1 h_{10}(t) + c_2 h_{20}(t) + c_3 h_{21}(t) + \ldots \quad (3.3)
\]

\[
c_i = 2^i \int_0^1 y(t) h_i(t) dt \quad (3.4)
\]

Similar to Fourier series, the more terms we have equation (3.3), the better our approximation is going to be. The first four Haar wavelets can be expressed as the following:

\[
h_{00}(t) = [1 \quad 1 \quad 1 \quad 1]
\]

\[
h_{10}(t) = [1 \quad 1 \quad -1 \quad -1]
\]

\[
h_{20}(t) = [1 \quad -1 \quad 0 \quad 0]
\]

\[
h_{21}(t) = [0 \quad 0 \quad 1 \quad -1]
\]

11
Figure 3.1: $H_{00}(t) \rightarrow H_{33}(t)$
The number of entries in each function from (3.5) signifies the interval $[0, 1]$ is cut into 4 equal pieces, for instance, $h_{20}(t) = [1 -1 0 0]$ represent the wavelet function that is 1 from $t = 0$ to $t = \frac{1}{4}$, -1 from $t = \frac{1}{4}$ to $t = \frac{1}{2}$, 0 from $t = \frac{1}{2}$ to $t = \frac{3}{4}$, and 0 from $t = \frac{3}{4}$ to $t = 1$. We obtain the Haar matrix of $4^{th}$ order when we combine everything in equation (3.5) into a matrix:

$$
H_4(t) \approx \begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & 1 & -1 & -1 \\
1 & 0 & 0 & 0 \\
0 & 1 & -1 & 0
\end{bmatrix}
$$

(3.6)

**Example**

We want to transform a piecewise constant function $y(t) = [9 1 2 0]$. The Haar coefficient can be found using equation (3.4), but here we want to propose an alternative method. We want to transform $y(t)$ such that

$$
y(t) = c^T H_4(t)
$$

(3.7)

where $c$ is the Haar coefficient vector. It is very convenient to find $c$ using matrix inversion:

$$
H_4^{-1}(t) = \frac{1}{4} \begin{bmatrix}
1 & 1 & 2 & 0 \\
1 & 1 & -2 & 0 \\
1 & 0 & 0 & 2 \\
1 & -1 & 0 & -2
\end{bmatrix}
$$

(3.8)

$$
c^T = y(t)H_4^{-1} = [3 2 4 1].
$$

(3.9)

Therefore, we have the transformation

$$
y(t) = 3h_{00}(t) + 2h_{10}(t) + 4h_{20}(t) + h_{21}(t)
$$

(3.10)
3.2 Integration of Haar Wavelet

One tool that is often used to solve differential equation models of dynamic systems is integration. The technique we are introducing is using a similar approach. We will start by looking at the integration of the first four Haar wavelets:

\[
\int_{0}^{t} h_{00}(t) \, dt = t, \quad 0 \leq t < 1 \quad \simeq \frac{1}{8} [1 \ 3 \ 5 \ 7] \quad (3.11)
\]

\[
\int_{0}^{t} h_{10}(t) \, dt = \begin{cases} \frac{t}{2}, & 0 \leq t < \frac{1}{2} \\ \frac{1}{2} - t, & \frac{1}{2} \leq t < 1 \end{cases} \quad \simeq \frac{1}{8} [1 \ 3 \ 3 \ 1] \quad (3.12)
\]

\[
\int_{0}^{t} h_{20}(t) \, dt = \begin{cases} \frac{t}{4}, & 0 \leq t < \frac{1}{4} \\ \frac{3}{4} - t, & \frac{1}{4} \leq t < \frac{1}{2} \end{cases} \quad \simeq \frac{1}{8} [1 \ 1 \ 0 \ 0] \quad (3.13)
\]

\[
\int_{0}^{t} h_{21}(t) \, dt = \begin{cases} \frac{t - \frac{1}{2}}{4}, & \frac{1}{2} \leq t < \frac{3}{4} \\ \frac{3}{4} - t, & \frac{3}{4} \leq t < 1 \end{cases} \quad \simeq \frac{1}{8} [0 \ 0 \ 1 \ 1] \quad (3.14)
\]

By writing equation (3.11) to (3.14) together in a matrix, we obtain

\[
\int_{0}^{t} H_4(t) \, dt \simeq \frac{1}{8} \begin{bmatrix} 1 & 3 & 5 & 7 \\ 1 & 3 & 3 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \quad (3.15)
\]

Let the integrals be expanded into Haar series, such that,

\[
\int_{0}^{t} H_4(t) \, dt = P_4 H_4(t) \quad (3.16)
\]

gives

\[
P_4 = \int_{0}^{t} H_4(t) \, dt H_4^{-1}(t) \quad (3.17)
\]
and

\[
P_4 = \frac{1}{2 \times 4} \begin{bmatrix}
4 & -2 & -1 & -1 \\
2 & 0 & -1 & 1 \\
1/2 & 1/2 & 0 & 0 \\
1/2 & -1/2 & 0 & 0
\end{bmatrix}
\] (3.18)

By following similar procedure, we can build another integration matrix, \( P_8 \), with higher resolution:

\[
P_8 = \frac{1}{16} \begin{bmatrix}
8 & -4H_1 \\
4H_1^{-1} & 0 & -2H_2 \\
4H_2^{-1} & 0 & -2H_2 \\
H_4^{-1} & 0 & 0
\end{bmatrix} = \frac{1}{16} \begin{bmatrix}
16P_4 & -H_4 \\
16P_4 & -H_4
\end{bmatrix}
\] (3.19)

In general for an \( m \)th-order system with \( m = 2^j, j \in \mathbb{N} \), the \( P_m \) is given as

\[
P_m = \frac{1}{2m} \begin{bmatrix}
2mP_{m/2} & -H_{m/2} \\
H_{m/2}^{-1} & 0
\end{bmatrix}
\] (3.20)

A general proof of the construction of \( P_m \) can be found in [2]. Here, we will verify \( P_m \) for \( m = 1, 2, 4, 8, \ldots \). The Haar matrix \( H_m \) is defined by \( m \) Haar functions in row vector form as in equation (3.6). Specifically,

\[
H_m(t) = \begin{bmatrix}
h_{00}(t) \\
h_{10}(t) \\
\vdots \\
h_{j,j-1}(t)
\end{bmatrix}
\] (3.21)

\[
H_1 = [1], \quad H_1^{-1} = [1]
\] (3.22)

\[
H_2 = \begin{bmatrix}
1 & 1 \\
1 & -1
\end{bmatrix}, \quad H_2^{-1} = \frac{1}{2} \begin{bmatrix}
1 & 1 \\
1 & -1
\end{bmatrix}
\] (3.23)
The operation matrix $P_m$ is the Haar transform coefficient matrix of these integrals defined in equation (3.16).

\[ P_m = \begin{bmatrix} \int_0^t H_m(\tau)d\tau \end{bmatrix} H_m^{-1} \]  

(3.24)

\[ P_1 = \begin{bmatrix} \frac{1}{2} \end{bmatrix}, \quad P_2 = \frac{1}{4} \begin{bmatrix} 2 & -1 \\ 1 & 0 \end{bmatrix} \]

\[ P_4 = \frac{1}{16} \begin{bmatrix} 8 & -4 & -2 & -2 \\ 4 & 0 & -2 & 2 \\ 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix} \]

(3.25)

\[ P_8 = \frac{1}{64} \begin{bmatrix} 32 & -1 & -8 & -8 & -4 & -4 & -4 & -4 \\ 16 & 0 & -8 & 8 & -4 & -4 & 4 & 4 \\ 4 & 4 & 0 & 0 & -4 & 4 & 0 & 0 \\ 4 & -4 & 0 & 0 & 0 & 0 & -4 & 4 \\ 1 & 1 & 2 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & -2 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 2 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & -2 & 0 & 0 & 0 & 0 \end{bmatrix} \]
Chapter 4

Haar Wavelet Method for Solving Differential Equations

4.1 Applying Haar wavelets to Differential Equations

Let us consider a linear system

\[ \dot{x}(t) = Ax(t) + Bu(t), \quad x(0) = x_0 \]  \hspace{1cm} (4.1)

with \( n \) states \( x(t) \) and \( p \) inputs \( u(t) \) that can be described in the state equation with properly dimensioned matrices \( A \) and \( B \). Assume that \( u(t) \) is square integrable in the interval \( 0 \leq t < 1 \). It can be expressed as a Haar series expansion

\[ u(t) = GH(t) \]  \hspace{1cm} (4.2)

where \( G \) is \( p \times m \), \( m \) being the size of matrix \( H \) which determines the resolution of the Haar matrix (3.6). \( G \) can be obtained from equation (3.4) for different \( i \) as a vector of coefficients of the Haar wavelets.
Furthermore, we also want to expand $\dot{x}(t)$ into a Haar series:

$$\dot{x}(t) = FH(t) \quad (4.3)$$

After integration, we have

$$x(t) = \int_0^t \dot{x}(\tau) d\tau + x_0 = \int_0^t H(\tau) d\tau + x_0 = FPH(t) + x_0. \quad (4.4)$$

Since $P$, $H$, and $x_0$ are known, if we can solve for $F$, then we have $x(t)$. By substituting equation (4.2), (4.3), and (4.4) into equation (4.1), we obtain

$$FH(t) = AFPH(t) + Ax_0 + BGH(t)$$

$$\Rightarrow [F - AFP]H(t) = \{Ax_0, 0, 0, \ldots, 0\} + BG \equiv G_1 \quad (4.5)$$

The zero entries in $H$ and $P$ will greatly simplify the solution processes. Equation (4.5) can also be expressed as

$$A^{-1}F - FP = A^{-1}G_1. \quad (4.6)$$

For nonsingular matrix $A$, this is a Lyapunov matrix equation [2]. By using `lyap(A^{-1}, -P, A^{-1}G_1)` from MATLAB, we can obtain $F$ directly with $A$ and $G_1$ being given.

### 4.2 Numerical Analysis

MATLAB has a sparse function that converts a full matrix to sparse form by squeezing out any zero elements. This function will save up a lot of computation time due to the large amount of zero entries in matrix $P$ and $H$.

**Example**

Consider the differential equation

$$y'' + 4y' + 3y = 1 \quad (4.7)$$
with initial condition

\[ y(0) = 0, \quad y'(0) = 1. \]

In order to apply the Haar’s method, we want to represent equation (4.7) as an equation of matrices.

Let \( x = \begin{bmatrix} y \\ y' \end{bmatrix} \), we can rewrite equation (4.7) as

\[ \dot{x} = \begin{bmatrix} 0 & 1 \\ -3 & -4 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \quad (4.8) \]

with \( u(t) = GH. \) (Equation (4.2))

The analytic solution is

\[ y(t) = \frac{1}{3} - \frac{e^{-3t}}{3} \]

First, we want to know how well Haar’s method works in terms of error. Though there are many ways to calculate error using different norms, we will be using the infinity norm to calculate the error, \( e. \)

\[ ||e||_{\infty} = \max\{e\} \]

Graph 4.1 gives some idea of how fit the Haar’s method is as we increase \( m \) by a power of 2. Table 4.1 shows the error between the Haar’s method and the analytic solution at different value of \( m. \) As we increase \( m \) from \( 2^p \) to \( 2^{p+1}, \) the error tends to drop by half, this means the Haar’s method for solving differential equation is first order.

<table>
<thead>
<tr>
<th>m</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>0.0909</td>
<td>0.0526</td>
<td>0.0286</td>
<td>0.0149</td>
<td>0.0076</td>
<td>0.0039</td>
</tr>
</tbody>
</table>

Table 4.1: Haar VS Analytic: Error (Example 1: Second order ODE)
Figure 4.1: Haar VS Analytic solution (Example 1: Second order ODE)
However, Haar’s method is still a first-order. There are other methods that has higher order than the Haar’s method, for example, Heun’s method is a second-order RK method. Let us compare Haar’s method and Heun’s method. Figure 4.3 compares the Haar’s method and Heun’s method to the analytic solution. Graphically, Heun’s method is a little better than Haar at $m = 8$. Let us compare the error and time between these two methods. Figure 4.4 shows the time and error difference between Haar and Heun’s method. In terms of time as $m$ increases, Haar remains fairly fast because as $m$ increases, the matrices $P$ and $H$ are much ”sparser”, or has more zero entries. Sparse matrix multiplication in Matlab is still fast even when the size of the matrices increases. Whereas the time taken for Heun’s method increases in a somewhat linear fashion.
The graph on the right of figure 4.4 shows the error for both methods as $m$ increases. When $m = 8$, the error is very close for both methods, but as we increase $m$, we can see that Heun’s method has an advantage. Heun’s error decreases much faster than Haar’s as $m$ increases. This is due to the order of these two methods.

It is hard to compare the efficiency of these two methods. Haar is better in computation time, but worse in error reducing. In order to better analyze the two methods, let’s look at time and error together.

In figure 4.5, the Haar’s method is better if the user is looking for an accuracy of at most $10^{-4}$. From the figure, we can see that Haar and Heun are going to cross at somewhere below $\text{Error} = 10^{-4}$. Therefore, if we want a better accuracy than $10^{-4}$, Heun’s method is more efficient than Haar’s method.
Example

Now let’s look at a higher order differential equation:

\[ y^{(IV)} + 2y''' - 3y'' + 3y' - 5y = k \sin t \quad (4.9) \]

with initial condition

\[ y(0) = 0, \quad y'(0) = 1, \quad y''(0) = 0, \quad y'''(0) = 0. \]

Similarly, we want to rewrite equation (4.9) as an equation of matrices.

Let \( x = \begin{bmatrix} y \\ y' \\ y'' \\ y''' \end{bmatrix} \), we can rewrite equation (4.9) as

\[
\dot{x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 5 & -3 & 3 & -2 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 0 \\ k \end{bmatrix} u(t) \quad (4.10)
\]

with \( u(t) = GH \). Since now we have \( k \sin t \) on the right hand side of equation (4.9), matrix \( G \) will be the Haar coefficients of the function \( \sin t \), the constant \( k \) is captured in equation (4.10). To do this, we pick the mesh, \( m \), first, and then take the value value of \( \sin t \) and the endpoint of every mesh to get the function value in a vector, \( W \), and find \( G \) using

\[
G = WH_m^{-1} \quad (4.11)
\]

The analytic solution takes a long time to obtain, instead, we will compare Haar’s method to MATLAB’s built-in solver ode45. Since ode45 is of medium order, we should be able to use ode45 as an analytic solution to this question.
<table>
<thead>
<tr>
<th>m</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error(Haar - ode45)</td>
<td>0.1245</td>
<td>0.0625</td>
<td>0.0312</td>
<td>0.0156</td>
<td>0.0078</td>
<td>0.0039</td>
</tr>
</tbody>
</table>

Table 4.2: Error between Haar and ode45 (Example 2: Fourth order ODE)

We can see from figure 4.6 again that Haar’s method works nicely when $m$ is large. It is very close to what ode45 does as $m$ increases. Table 4.2 the maximum different between Haar’s method and ode45 at different values of $m$. The difference is halved every time we double $m$, therefore, again, concluding that the Haar’s method is of order 1.
Example
A stiff ordinary differential equation often poses troubles for explicit method such as Heun’s method. The solution can be unstable unless if the step size is taken to be very small. The idea is that some terms in the equation lead to a rapid variation in the solution, such as vanishing at a fast rate. For example,

\[ y'' + 1001y' + 1000y = 0 \]  \hspace{1cm} (4.12)

with initial condition \( y(0) = 1, y'(0) = 0 \) is a stiff ODE problem with analytic solution

\[ y = \frac{1}{999}(1000e^{-t} - e^{-1000t}). \]  \hspace{1cm} (4.13)

The term \( e^{-1000t} \) decreases to 0 much faster than \( e^{-t} \), and that will be a trouble when we try to solve this using Heun’s method.

From left of figure 4.8, we can see that when error of Heun’s method increases rapidly when \( m \) increases until \( m \approx 10^3 \). Whereas the error for Haar’s method is consistently small when compared to Heun’s method. Right of figure 4.8 shows that Haar’s method is definitely a better choice than Heun’s method in terms of run time and accuracy. However, since Heun’s method is explicit, we know Heun’s method is not our first choice when it comes to solving stiff ODE, it would not be fair to compare Haar’s method to Heun’s.

Implicit Euler’s method (IEM), or backwards Euler’s method, is similar to the standard Euler’s method, but differs in that it is an implicit method. Implicit methods find a solution by solving an equation involving both the current state of the system and the later one. For instance, the forward, or standard, Euler’s method

\[ \frac{1}{h}(y_{i+1} - y_i) = Ay_i \]

yields

\[ y_{i+1} = (I + hA)y_i \]
for $i = 0, 1, 2, \ldots$

Whereas, the backwards, or implicit, Euler method

$$\frac{1}{h}(y_{i+1} - y_i) = Ay_{i+1}$$

yields

$$y_{i+1} = (I - hA)^{-1}y_i$$

for $i = 0, 1, 2, \ldots$ Implicit Euler’s method has order one and is L-stable, which makes it a good choice when we want to solve a stiff equation.

Based on figure 4.9, Haar’s method and implicit Euler’s method seem to work fine for this stiff equation even when $m$ is small. Unlike Heun’s method where the error is huge if $m$ is not big enough. Now, it would be more reasonable to compare Haar’s method and IEM in details.

<table>
<thead>
<tr>
<th>m</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>Haar Error</td>
<td>0.1102</td>
<td>0.0579</td>
<td>0.0294</td>
<td>0.0145</td>
<td>0.0073</td>
<td>0.0038</td>
</tr>
<tr>
<td>Implicit Euler Error</td>
<td>0.0418</td>
<td>0.0219</td>
<td>0.0112</td>
<td>0.0057</td>
<td>0.0029</td>
<td>0.0014</td>
</tr>
<tr>
<td>Haar Error/IEM Error</td>
<td>2.6394</td>
<td>2.6453</td>
<td>2.6177</td>
<td>2.5456</td>
<td>2.5565</td>
<td>2.6398</td>
</tr>
</tbody>
</table>

Table 4.3: Haar vs Analytic, Haar vs Implicit Euler: Error (Example 3: Second order stiff ODE)

Table 4.3 tells us that IEM is definitely a better choice in terms of minimizing error at different step sizes. Haar Error is about 2.6 times Implicit Euler Error, which is somewhat significant. Now let’s look at the graph 4.10, Haar’s time curve is below Euler’s curve for all possible values of $m$. Here we can really see the power of MATLAB’s sparse mode on saving computational time. IEM is better in accuracy whereas Haar’s method
is fast in time. In order to further compare these two methods, figure 4.11 shows both methods in terms of their time and error, although IEM is good on accuracy, but when we want efficiency, Haar’s method is a better one. Haar’s method can be more accurate without significant additional computation time. On the other hand, if we can one more digit of accuracy with IEM, it takes 10 times longer to go from a $10^{-2}$ to $10^{-3}$. Since both methods are first order, Haar’s method has the upper hand of applying its method in MATLAB with sparse mode to reduce a lot of the computation time needed.
Figure 4.3: Haar VS Analytic VS Heun: Solution (Example 1: Second order ODE)
Figure 4.4: Haar VS Heun: Time, Error (Example 1: Second order ODE)
Figure 4.5: time vs error: Haar and Heun (Example 1: Second order ODE)
Figure 4.6: Haar VS ode45 (Example 2: Fourth order ODE)
Figure 4.7: Error between Haar and ode45 (Example 2: Fourth order ODE)
Figure 4.8: $m$ vs error, time vs error: Heun and Haar (Example 3: Second order stiff ODE)
Figure 4.9: Haar vs Implicit Euler vs Analytic (Example 3: Second order stiff ODE)
Figure 4.10: m vs error, m vs time, Haar and IEM (Example 3: Second order stiff ODE)
Figure 4.11: Time vs Error: Stiff (Example 3: Second order stiff ODE)
Chapter 5

Conclusion

The Haar wavelet orthogonal functions and their integration matrices have been introduced to solve higher order ODE and stiff ODE. The sparsity of matrices $P$ and $H$ brings down the computational time dramatically thanks to the sparse function in MATLAB. The method is of first order; however, due to the fast computational time, we can make step sizes smaller to get better accuracy without adding any significant amount of computational time as seen in several numerical examples.

In a theoretical point of view, Haar’s method is simply derived and proved. Out of many well-known wavelets, Haar’s is considered the simplest one. This method has two main advantages: (i) it is suitable for non-stiff and stiff ODE. Regardless of the types of problem, Haar’s method is stable in terms of error reducing versus step sizes. (ii) Fast computational time as mentioned earlier. This approach of Haar’s can also be extended to other wavelets, such as Daubechies’ wavelets. Daubechies wavelets are much more complicated in terms of construction compared to the Haar wavelets. Also an orthogonal basis, Daubechies wavelets can be applied in a similar way.
Bibliography


