A Project report on Implementation of Wavelets to Solve Mathematical Problems

Sethurathienam Iyer S - 2013B4A7720G Rishabh Khurana - 2013B4A7785G

Submitted in the partial fulfillment of requirements of

MATH F376 - DESIGN PROJECT

Under the Supervision of

Dr. Amit Setia





BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI - K.K. BIRLA GOA CAMPUS August 2015 - December 2015

Contents

Acknowledgement			
Abstract		tract	
	1 l	Haar Approximation of functions	3
	1.1	Advantages of Wavelet Analysis	3
	1.2	Haar Approximation : Theory	4
	1.3	Haar Approximation: Implementation	4
2 Haar Approximation of Fractional Differential		ar Approximation of Fractional Differential Equations	6
	2.1	Theory	6
	2.2	Implementation	7
3	Err	Error Analysis	
	3.1	Theory	14
	3.2	Estimation of M	14
	3.3	Implementation	15
4	Ref	erences	16

Acknowledgement

We would like to thank our mentor and instructor in charge Dr. Amit Setia for providing us this wonderful opportunity to explore this field of fractional differential equations. We would also like to thank him for having faith in us and for his patience.

We would also like to thank all our friends who helped us through the completion of this project. Finally we would like to thank our family and friends for their continuous support and the confidence they bestowed upon us during this project.

Abstract

This report presents an overview of the techniques which are used to find out the numerical solutions of fractional differential equations using Haar wavelets.

We chose to use MATLAB as it is powerful high level language which provides interactive environment for numerical computations. Firstly, we discuss about approximating the functions using haar wavelets. We can approximate any functions over any interval with desired accuracy.

The following section discusses the theory of solving fractional differential equation followed by its implementation. We extend the idea to system of fractional differential and show it's implementation as well.

The concluding section discusses about the error analysis for numerical solution of fractional differential equations to conclude that the method we use to calculate the solution is indeed convergent.

1 Haar Approximation of functions

Wavelet analysis is a relatively new area in mathematics research. It has been applied widely in signal analysis, time-frequency analysis and numerical analysis.

Functions are expanded to summation of basis functions, and every basis function is achieved by compression and translation of a mother wavelet function with good properties of smoothness and locality, which makes people study the properties of integer and locality in the process of expressing functions.

1.1 Advantages of Wavelet Analysis

Wavelets were first introduced in seismology to provide a time dimension in seismic analysis that Fourier analysis lacked. Fourier analysis is ideal for studying stationary data, whose statistical properties are invariant over the time. Wavelets are designed to study these non stationary data in mind, and quickly become useful to a number of disciplines.

There are many type of wavelets such as Morlet wavelet, Daubechies, Haar wavelelts etc. In this project, we have used Haar wavelet for the functional approximation. (See Figure 1).

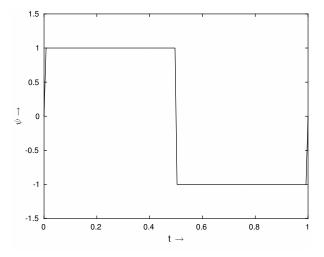


Figure 1: Haar Wavelet

1.2 Haar Approximation: Theory

Haar wavelet is a function defined on \mathbb{R} which is defined as

$$H(t) = \begin{cases} 1 & 0 \le t < \frac{1}{2} \\ -1 & \frac{1}{2} \le t < 1 \\ 0 & \text{elsewhere} \end{cases}$$

We can do two operations on it namely Compression/Dilation and shifting it along the x axis. We need to produce a sequence of orthogonal functions using these two operations for the necessary approximation.

For i=1,2,..., write $i=2^j+k$ with j=0,1,... and $k=0,1,2,...,2^j-1$. define $h_i(t)=2^{\frac{j}{2}}H(2^jt-k)$. It may be shown that the sequence $(h_i)_0^\infty$ is an orthogonal system in $L^2[0,1]$. Therefore, for $u\in C[0,1]$, the series $\sum_i < u, h_i > h_i$ converges uniformly to u. Therefore a function u(t) can be decomposed as $u(t)=\sum_0^\infty c_i h_i(t)$ where $c_i=\int_0^1 u(t)h_i(t)dt$. However, in practice, only k terms are considered, where k is power of 2. So we can safely write u(t) as approximately $\sum_0^{k-1} c_i h_i(t)$ which can be represented as $u_k(t)$.

1.3 Haar Approximation: Implementation

The Implementation of the above algorithm is as follows:

```
Listing 1: approx.m
```

```
%— Functional approximation using haar wavelets --%
function [ap,z,err] = approx()
tic
[wl,x]=wpfun('db1',63);
%— wl is 63 X 130 matrix now --%
y=x;
%— this is the function to approximate --%
for i=1:63
    g=y.*wl(i,:);
    s=sum(g(1:length(x)));
    c(i)=s/length(x);
```

```
%— calculating <ui(x) fi(x)> --%
ap=0;
for i=1:63
         ap=ap+c(i).*wl(i,:);
end
%— f(x) = summation i from 0 to k-1 ci * hi --%
z=x;
plot(x,ap); hold on; plot(x,y); xlabel('x'); ylabel('y');
title('Functional_Approximation_using_Haar_wavelets_');
%— compare the approximation --%
err=sqrt(abs((ap-y).*(ap-y)));
%— computes the error --%
toc
end
```

The corresponding output in MATLAB is as follows:

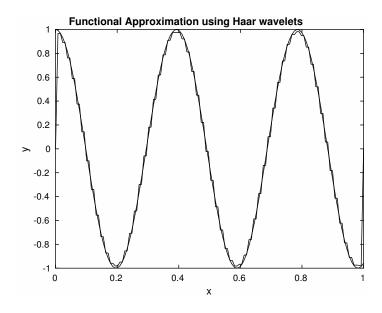


Figure 2: Plot of the solution

2 Haar Approximation of Fractional Differential Equations

In this section, we would first look at the theory of approximating fractional differential equations and then we would show the MATLAB implementation as well as output.

2.1 Theory

Consider the fractional differential equation with variable coefficients :

$$\mathbf{D}_*^{\alpha}\mathbf{u}(\mathbf{t}) + \mathbf{a}(\mathbf{t})\mathbf{u}(\mathbf{t}) = \mathbf{f}(\mathbf{t}), \mathbf{u^{(i)}} = \mathbf{0}, \mathbf{i} = \mathbf{0}, \mathbf{1}, ..., \mathbf{m-1}, \mathbf{0} < \mathbf{m-1} < \alpha \leq \mathbf{m}$$

By approximating $D_*^{\alpha}u(t)$, we have

$$\mathbf{D}_*^{\alpha}\mathbf{u}(\mathbf{t}) \equiv \mathbf{D}_*^{\alpha}\mathbf{u}_\mathbf{k}(\mathbf{t}) = \sum_{i=0}^{k-1}\mathbf{c}_i\mathbf{h}_i(\mathbf{t}) = \mathbf{c}^\mathbf{T}\mathbf{H}_\mathbf{k}(\mathbf{t})$$

Therefore, u(t) can be expressed in terms of operational matrix of fractional integration of haar wavelets as

$$u(t) \equiv u_k(t) = c^T P_{k \times k}^{\alpha} H_k(t)$$
, where

 $P_{k\times k}^{\alpha} = H_{k\times k} F_{\alpha} H_{k\times k}^{-1},$

$$F_{\alpha} = \frac{1}{m^{\alpha}} \frac{1}{\Gamma(\alpha+2)} \begin{bmatrix} 1 & \epsilon_{1} & \epsilon_{2} & \dots & \epsilon_{m-1} \\ 0 & 1 & \epsilon_{1} & \dots & \epsilon_{m-2} \\ 0 & 0 & 1 & \dots & \epsilon_{m-3} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix},$$

$$\xi_{k} = (k+1)^{\alpha+1} - 2k^{\alpha+1} + (k-1)^{\alpha+1}.$$

$$\xi_k = (k+1)^{\alpha+1} - 2k^{\alpha+1} + (k-1)^{\alpha+1}$$
.
Also,

$$H_{k \times k} = [H_k(\frac{1}{2k})H_k(\frac{3}{2k})...H_k(\frac{2k-1}{2k})],$$
 where

$$H_k(t) = [h_0(t), h_1(t), ..., h_{k-1}(t)]^T$$

Here, $P_{k \times k}$ is Operational Matrix of fractional integration of Haar wavelets.

By Substituting these approximations in original equation, we get
$$c^T H(t) + a(t).c^T P_{k \times k}^{\alpha} H_k(t) = f(t)$$
 where $H = [H_k(t_0)H_k(t_1)...H_k(t_{k-1})].$

This is a system of algebraic equations which can be solved easily using MATLAB.

2.2 Implementation

In this section, we would show all the code listings necessary to implement the single equation solver.

The Following code tries to find out the numerical solution of the differential equation

$$D_*^{\frac{1}{3}}u(t) + t^{1/3}u(t) = \frac{3}{\Gamma(\frac{2}{3})}t^{\frac{2}{3}} + t^{\frac{4}{3}}, u(0) = 0$$

1. differential_solver.m

```
alpha=1/3; k=64;
af=0(x) x^{(1/3)};
ff=0(x) (x^{(4/3)}+(((x^{(2/3)})*1.5)/gamma(2/3)));
A=return_a_matrix(af,k);
F=return_fvalue_vector(ff,k);
F=F';
H=return_operational_matrix(k,0);
P=return_p_matrix(alpha,k);
E=P*H;
E=E*A;
E=E+H;
C=F/E;
Y=[];X=[];
for i=1:k
    [h,x]=return_h_column(i,k);
    y=C*P*h;
    X = [X, X];
    Y = [Y, y];
end
q=0:.1:1;
NY=interp1(X,Y,q);
plot (q, NY);
```

```
2. return_a_matrix.m
function [ A ] = return_a_matrix( f,k )
v=[];
for i=1:k
    v=[v, f((2*i-1)/(2*k))];
end
A=diag(v);
end
3. return_fvalue_vector.m
function [ fvalue ] = return_fvalue_vector( f,k)
fvalue=[];
for i=1:k
   fvalue=[fvalue; f(((2*i-1)/(2*k)))];
end
end
4. return_operational_matrix.m
function [ H ] = return_operational_matrix( k, flag)
   H=[];
   f=0(x, j, k)
 ((x<((0.5+k)/2^j))&(x>=(k/2^j)))-(x<((1+k)/2^j)&(x>=((0.5+k)/2^j)));
    [J K] = generate_j_k(k-1);
   for i=(1-flag):(k-flag)
       h=1;
       for j=1:k-1
           h=[h; ((2^{(j)/2}))*f(((2*i-1)/(2*(k-flag))), J(j), K(j)))];
       end
        H = [H, h];
    end
end
```

```
5.return_p_matrix.m
function [ P ] = return_p_matrix( alpha,k )
P=[];
H=return_operational_matrix(k,0);
F=return_f_matrix(alpha,k);
P=H*F;
P=P/H;
end
6. return_h_column.m
function [ H,X ] = return_h_column( i,k )
    f=0(x, j, k)
  ((x<((0.5+k)/2^{j}))&&(x>=(k/2^{j})))-(x<((1+k)/2^{j})&&(x>=((0.5+k)/2^{j}))); \\
    [J K] = generate_j_k(k-1);
    H=1;
    X=(2*i-1)/(2*(k));
    for j=1:k-1
        H=[H;((2^{(j)/2}))*f(X,J(j),K(j)));
    end
end
7.generate_j_k.m
function [J,K] = generate_j_k( n )
    j=0; k=0; J=[]; K=[];
    for i=1:n
    k=0;
    while i = 2^j+k
        k=k+1;
        if k==2^j
             j=j+1; k=0;
             break
        end
    end
    J = [J, j];
    K = [K, k];
    end
```

end

8. return_f_matrix.m

```
function [ F ] = return_f_matrix( alpha,k )
e=@(x)(((x+1)^(alpha+1))-(2*x^(alpha+1))+((x-1)^(alpha+1)));
F=eye(k);

for i=1:k
    for j=(i+1):k
        F(i,j)=e(j-i);
    end
end

F=F./(k^alpha);
F=F./(gamma(alpha+2));
```

end

The output is as shown below in comparison with the exact solution u(t) = t.

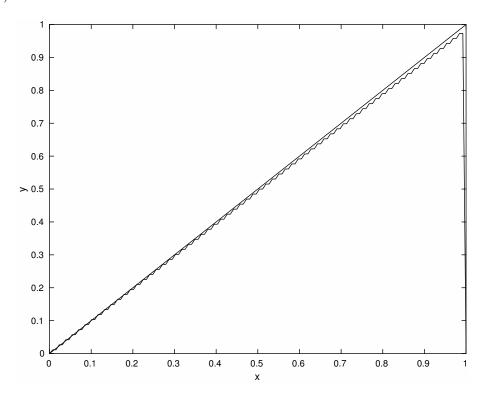


Figure 3: Plot of the solution

```
We can extend this idea to solve system of equations as well, and the following code solves the given system of equations
```

```
D_*^{\alpha 1} u_1(t) = 2u_2^2(t), 0 < \alpha 1 \le 1
D_*^{\alpha 2} u_2(t) = t u_1(t), 0 < \alpha 2 \le 1
D_*^{\alpha 3} u_3(t) = u_2(t)u_3(t), 0 < \alpha 3 \le 1
with the initial conditions u_1(0) = 0, u_2(0) = 1, u_3(0) = 1.
system_solver.m
k=2;
alpha=1;
f1=@(x,u1,u2,u3) 2*u2^2;
f2=@(x,u1,u2,u3) x*u1;
f3=@(x,u1,u2,u3) u2*u3;
A=sym('a',[1 k]);
B = sym('b', [1 k]);
C = sym('c', [1 k]);
P=return_p_matrix(alpha,k);
LHSA=[];
for i=1:k
    LHSA=[LHSA; A*return_h_column(i,k)];
end
LHSB=[];
for i=1:k
     LHSB=[LHSB; B*return_h_column(i,k)+1];
end
LHSC=[];
for i=1:k
    LHSC=[LHSC;C*return_h_column(i,k)+1];
end
RHSA=[];
RHSB=[];
RHSC=[];
for i=1:k
     JB=A*P*return_h_column(i,k);
     JC=B*P*return_h_column(i,k);
     JD=C*P*return_h_column(i,k);
```

```
J1=f1((2*i-1)/(2*k),JB,JC,JD);
J2=f2((2*i-1)/(2*k),JB,JC,JD);
J3=f3((2*i-1)/(2*k),JB,JC,JD);
RHSA=[RHSA;J1];
RHSB=[RHSB;J2];
RHSC=[RHSC;J3];
end

LHS=[LHSA;LHSB;LHSC];
RHS=[RHSA;RHSB;RHSC];

EQN=[];
for i=1:3*k
    EQN=[EQN,LHS(i)==RHS(i)];
end

VAR=[A,B,C];
S=vpasolve(EQN,VAR);
```

Figure 4,5,6 shows the plot of the solution for k=2.

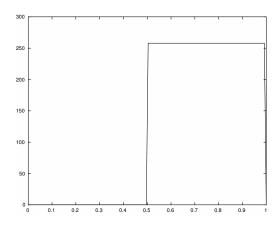


Figure 4: Plot of the solution: u1

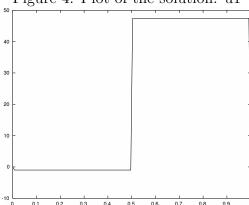


Figure 5: Plot of the solution: u2

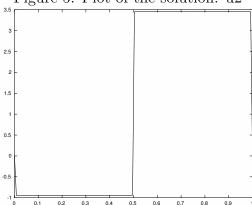


Figure 6: Plot of the solution: u3

3 Error Analysis

In this section, we would see the error analysis of the approximation algorithm we are using and we show that as we increase the value of k, the error would tend to zero.

3.1 Theory

For the analysis, we assume that $u^{(m)}(t)$ is continuous and bounded on (0,1). Mathematically,

$$\exists M > 0, \forall t \in (0,1), |u^{(m)}(t)| \le M$$

We need to calculate $D_*^{\alpha}u(t) - D_*^{\alpha}u_k(t)$ and find out it's limit as $k \to \infty$. For this purpose, we use the following theorem.

Theorm 6.1 Suppose that the function $D_*^{\alpha}u_k(t)$ obtained by haar wavelets are the approximation of $D_*^{\alpha}u(t)$, then we have exact upper bound as follows;

$$||D_*^{\alpha} u(t) - D_*^{\alpha} u_k(t)||_E \le \frac{M}{\Gamma(m-\alpha).(m-\alpha)} \frac{1}{[1-2^{2(\alpha-m)}]^{\frac{1}{2}}} \frac{1}{k^{m-\alpha}}$$

where
$$||u(t)||_E = (\int_0^1 u^2(t)dt)^{\frac{1}{2}}$$
.

We can see clearly that the error tends to zero as $k \to \infty$. This concludes that Haar wavelets method is convergent when it is used to solve numerical solution of fractional differential equations.

3.2 Estimation of M

The integration of Haar wavelets is expandable into Haar series with Haar coefficient matrix $Q_{k\times k}$, which is also called operational matrix of integration.

So, if we approximate $u^{(m)}(t)$ as $b^T H_k(t)$, then we can approximate u(t) as $b^T Q_{k \times k}^m H_k(t)$ by repeatedly integrating and approximating.

Writing in Matrix form,

$$u^T = b^T Q_{k \times k}^m H_{k \times k}.$$

Solving this equation, we can find b^T and therefore, we can estimate

 $u^{(m)}(x_i)$ for each i. We can safely take the upper bound as

$$\epsilon + \max_{0 \le i \le l} u^{(i)}(x_i).$$

3.3 Implementation

This sections shows the implementation to calculate the error bounds in MATLAB.

```
1. Error.m
```

```
k=8; l=2; m=1; alpha=1/4;
Q=return_operational_matrix(k,0);
C = sym('c', [1 k]);
Y=load('Y');
Y=Y.Y;
LHS=[];
for i=1:k
    LHS=[LHS,C*Q*return_h_column(i,k)];
end
[c1 c2 c3 c4 c5 c6 c7 c8] = vpasolve(Y==LHS,C)
c1=double(c1);
c2=double(c2);
c3=double(c3);
c4=double(c4);
c5=double(c5);
c6=double(c6);
c7=double(c7);
c8=double(c8);
C=[c1, c2, c3, c4, c5, c6, c7, c8];
Z=[];
for i=1:1
    Z=[Z,C*return_h\_column_x((i-1)/(l),k)];
end
E=return_error_bound(max(Z),k,m,alpha);
2. return_error_bound.m
function [ E ] = return_error_bound(M,k,m,alpha )
```

4 References

- 1. Y.Chen et al. Error analysis of fractional differential equation by Haar wavelets method, Journal of Computational Science 3 (2012) 367 373.
- 2. K.P._Ramachandran, Insight into wavelets