Diploma Thesis

Mikalai Zhudro

September 25, 2010

Contents

1 Wavelets and Multiresolution Analysis ............................................. 5
  1.1 Wavelets construction on $\mathbb{R}$ ............................................ 6
  1.2 Wavelet construction on $\mathbb{R}^2$ ........................................... 13
  1.3 Wavelet construction on an interval ............................................ 15

2 Discrete wavelet transform ............................................................. 17
  2.1 Discrete Wavelet Transform in one dimension ............................ 18
  2.2 Periodized wavelet transform .................................................. 20
  2.3 Discrete Wavelet Transform in two dimensions ........................... 21

3 Discretization of operator equations ................................................ 22

4 Compactly supported wavelets .......................................................... 24
  4.1 Haar wavelet ............................................................................. 24
  4.2 Haar transform ......................................................................... 24
  4.3 Daubechies wavelets .................................................................. 26

5 Description of algorithms ..................................................................... 35
  5.1 Numerical integration .................................................................. 35
  5.2 Newton-Cotes ............................................................................ 35
  5.3 Gauss-Legendre quadrature rule ............................................... 36
  5.4 Adaptive integration .................................................................... 37
  5.5 Numerical algorithms .................................................................. 38
    5.5.1 Wavelet decomposition of a function ..................................... 39
    5.5.2 Discretization of an integral operator ..................................... 41
    5.5.3 Solving integral equation ...................................................... 44
6 Numerical results
   6.1 Haar wavelet ........................................ 45
   6.2 Daubechies wavelets .................................. 54

7 Conclusion .................................................. 65

A Tables of $h_k$ coefficients .......................... 68
B Table of $C^n_k$ coefficients .......................... 70
C Gauss-Legandre coefficients $w_i$ and $x_i$ ........... 70
Introduction

There are several methods of representation of an integral operator with constant limits of integration. In this work we perform a decomposition of an integral operator with respect to a family of compactly supported wavelets (Haar and Daubechies) Because we take an integral over a bounded interval namely $[0,1]$, some adaptations are needed to construct a wavelets basis on a bounded domain. We choose periodized wavelets and express the integral operator in a periodized wavelet basis. This will lead to a matrix in anisotropic wavelet basis. Next step will be to analyze decreasing of elements in this matrix and, of course, its condition number. We will compare condition numbers arising from the wavelet decomposition on different levels and for the different wavelet bases of Daubechies family. As a result, we propose a fast algorithm for the wavelet representation of an integral equation of the first kind.

In chapter we introduce the concept of Multiresolution Analysis (MRA) and wavelets. For constructing wavelet bases we need to construct first scaling function. Therefore all necessary theorem will be provided as well as three types of construction methods of scaling functions. We will determine wavelet basis for $\mathbb{R}, \mathbb{R}^2$ and wavelet construction on the interval $[0, 1]$. Finally periodized wavelet transform will be described.

Chapter 2 provides information about the Discrete Wavelet Transform (DWT). Formulas for discrete wavelet transform in case of Haar and Daubechies wavelet db4 will be given. Then we express them in terms of upsampling and downsampling operators. Next we introduce DWT in two-dimensional case.

In chapter 3 we develop a wavelet-based representation for the Fredholm operator equation of the first kind. First, we express unknown function in wavelet bases. As next step we rewrite our equation in matrix form.

Chapter 4 presents compactly supported wavelets - Haar and Daubechies wavelets. Formulas for two-dimensional Haar wavelet transform will be given. As construction of Daubechies wavelets requires a lot of preliminary work we will provide only base theory.

In Chapter 5 we present all numerical algorithms used in this master thesis. First, we have a close look at numerical integration by Newton-Cotes and
Gauss-Legendre quadrature rules. Besides we will briefly describe adaptive integration, which is used for integration of peaked functions. Then we will develop algorithms for discretization of right hand side, integral operator, solving integral equation via Haar and Daubechies wavelets.

Finally in chapter 6 numerical results are presented. We compare the running time, approximation error, condition numbers for different discretization scales as well as for different types of wavelets (dbN).

In chapter 7 we summarize the main ideas of the present work and the ideas for future research.
1 Wavelets and Multiresolution Analysis

First we want to introduce the concept of Multiresolution Analysis (MRA). The idea behind MRA is: we want to decompose a function $f$ in a discrete set of wavelet coefficients $d_{a,b} = (f, \psi_{a,b}) = \int_{-\infty}^{\infty} f(t) \psi_{a,b}(t) dt$ (1), where $a > 0, b \in \mathbb{R}$, such that for all $f \in L^2(\mathbb{R})$ a reconstruction from these coefficients is possible. We restrict ourself only for coefficients $(a, b)$ from the set $C = \{(2^{-j}, k 2^{-j}), (j, k) \in \mathbb{Z}^2\}$. Then corresponding wavelets functions will be given as $\psi_{j,k}(t) = 2^{j/2} \psi(2^j t - k)$. ($j$ called scaling values and $k$ called translations). If wavelets constitute orthonormal basis we can express the function $f$ as

$$f(t) = \sum_{j,k} d_{j,k} \psi_{j,k}(t)$$ (2)

But creating a mother wavelet such that the decomposition and the reconstruction are possible is quite a challenging problem. To overcome these difficulties once were introduced concept of Multiresolution Analysis of $L^2(\mathbb{R})$. Simply spoken, we divide the whole space $L^2(\mathbb{R})$ into different subspaces which correspond do different scaling values and these subspaces are generated by scaling function $\phi$.

Definition 1. [16]. Sequence of subspaces $V_j \in L^2(\mathbb{R})$ is called MRA of $L^2(\mathbb{R})$ if the following properties are satisfied

- $V_j \in V_{j+1}, \cap_{j \in \mathbb{Z}} V_j = \{0\}, \cup_{j \in \mathbb{Z}} V_j = L^2(\mathbb{R})$ (completeness)
- Scale invariance: $f(t) \in V_j \iff f(2t) \in V_{j+1}$
- Shift-invariance: $f(t) \in V_0 \iff f(t - k) \in V_0$
- Shift-invariant basis: $\exists$ function $\phi$ such that $\{\phi(t - k)\}_{k \in \mathbb{Z}}$ constitute a Riesz basis of $V_0$
- Shift-invariant basis: $\exists$ function $\phi$ such that $\{\phi(t - k)\}_{k \in \mathbb{Z}}$ constitute a orthonormal basis of $V_0$
Last two properties are interchangeable.

Now we are going to introduce concept of wavelets. By (mother wavelet) we can take every function $\psi$ which has following properties [12]:

1. $\psi \in L^2(\mathbb{R})$
2. $t\psi \in L^2(\mathbb{R})$
3. $\omega \hat{\psi}(\omega) \in L^2(\mathbb{R})$
4. $\hat{\psi}(0) = \int_{-\infty}^{\infty} \psi(t) dt = 0$

1.1 Wavelets construction on $\mathbb{R}$

The idea is the following- we want to decompose $V_{j+1}$ into direct sum of $V_j$ and $W_j$. and we want to have a function $\psi$ such that $\{2^{j/2}\psi(2^j t - k)\}_{k \in \mathbb{Z}}$ constitute a Riesz basis of $W_j$. The most straightforward way is orthogonal extension $V_{j+1} \perp W_j$. Wavelets construct in this way are called semi-orthogonal wavelets. A different approach is orthogonal wavelets - with an additional property that $\psi_{j,k}$ constitute a orthonormal basis of $L^2(\mathbb{R})$. In this work we consider only orthogonal wavelets. So far we achieve decomposition of $L^2(\mathbb{R})$ into subspaces $W_j$:

$$L^2(\mathbb{R}) = \ldots \bigoplus W_{-1} \bigoplus W_0 \bigoplus W_1 \bigoplus W_2 \ldots$$

Note that the decomposition spaces $V_j$ are called approximation subspaces and $W_j$ called detail subspaces.

From idea of Multyresolution Analysis for $j = 0$ we will get $V_1 = V_0 \bigoplus W_0$. For satisfying that it is necessary and sufficient to have:

1. $W_0 \subset V_1$
2. $V_0 \cap W_0 = \{0\}$
3. $\phi(2t - k) \in V_0 \bigoplus W_0 = V_1$

First condition implies that we will have for $\psi$ the following representation

$$\psi(t) = \sqrt{2} \sum_{k=-\infty}^{\infty} g_k \phi(2t - k)$$
Let’s denote by $P_j f$ the orthogonal projection on $V_j$:

$$P_j f = P_{V_j} f = \sum_{k \in \mathbb{Z}} (f, \phi_{j,k}) \phi_{j,k}$$

and by $Q_j f$ orthogonal projection on $W_j$.

$$Q_j f = P_{W_j} f = P_{j+1} f - P_j f = \sum_{k \in \mathbb{Z}} (f, \psi_{j,k}) \psi_{j,k}$$

If $j_0 < j_1$ we can express orthogonal decomposition on $V_{j+1}$ into

$$P_{j+1} f = P_{j_0} f + \sum_{j_0 \leq j < j_1} Q_j f$$  \hspace{1cm} (5)

which mean

$$\sum_k c_{j_1,k} \phi_{j_1,k} = \sum_k c_{j_0,k} \phi_{j_0,k} + \sum_{j_0 \leq j < j_1} \sum_k d_{j,k} \psi_{j,k}$$  \hspace{1cm} (6)

where $c_{j,k} = (f, \phi_{j,k})$ and $d_{j,k} = (f, \psi_{j,k})$.

Last formula can be seen as orthogonal change of basis on $V_{j_1}$, therefore these sets \{\phi_{j_1,k}\}_{k \in \mathbb{Z}} and \{\phi_{j_0,k}\}_{k \in \mathbb{Z}} U \{\psi_{j,k}\}_{j_0 \leq j < j_1, k \in \mathbb{Z}} represent orthogonal bases of $V_{j_1}$.

From the first property of MRA it follows that

$$\lim_{j \to +\infty} \|f - P_j f\|_{L^2(\mathbb{R})} = 0$$  \hspace{1cm} (7)

From (5) and (7) we see that \{\phi_{j_0,k}\}_{k \in \mathbb{Z}} U \{\psi_{j,k}\}_{j_0 \leq j < j_1, k \in \mathbb{Z}} is an orthogonal basis of $L^2(\mathbb{R})$.

Therefore, each function $f \in L^2(\mathbb{R})$ can be expressed as follows

$$f(t) = \sum_{k \in \mathbb{Z}} c_{j_0,k} \phi_{j_0,k}(t) + \sum_{j \geq j_0} \sum_{k \in \mathbb{Z}} d_{j,k} \psi_{j,k}(t)$$  \hspace{1cm} (8)

Note that $c_{j,k}$ called the approximation coefficients and $d_{j,k}$ - detail coefficients.

Question about construction of $\psi$ is still open. Is it possible knowing $\phi$ find $\psi_{j,k}$?

As we have scale-invariance property is it sufficient to create $\psi_{0,k}$ which will constitute orthonormal basis of $W_0$? To answer these questions we have to look more carefully at the scaling function.
We know that $\phi \in V_0 \subset W_0$ and $\{\sqrt{2}\phi(t-k)\}_{k \in \mathbb{Z}}$ is the orthonormal basis of $V_1$. We can find a sequence $h_k$ with $\sum_{k=-\infty}^{+\infty} |h_k|^2 < \infty$, such that

$$\phi(t) = \sqrt{2} \sum_{k=-\infty}^{+\infty} h_k \phi(2t - k)$$

(9)

This formula is called the dilatation equation or the two-scale relation.

Applying Fourier transform to the both sides of equation (9) we will get

$$\hat{\phi}(\omega) = \frac{\hat{\phi}(\omega/2)}{\sqrt{2}} \sum_{k=-\infty}^{+\infty} h_k e^{-ik\omega/2}$$

(10)

Let’s introduce a two scale symbol $H(z)$ for the scaling function $\phi$

$$H(z) = \sum_{k=-\infty}^{+\infty} h_k z^{-k}$$

(11)

Rewriting (10) we get

$$\hat{\phi}(\omega) = \frac{\hat{\phi}(\omega/2)}{\sqrt{2}} H(e^{-ik\omega/2}) = \hat{\phi}(0) \prod_{k=1}^{+\infty} H(e^{-ik\omega/2^k}) = H_0(\omega/2) \hat{\phi}(\omega/2)$$

(12)

where

$$H_0(\omega) = \frac{1}{\sqrt{2}} H(e^{-i\omega/2})$$

(13)

Function $H_0(\omega)$ is 2$\pi$ periodic function belonging to $L^2(\mathbb{R})$.

From (12) we see if $\hat{\phi}(0) = 0$ then $\phi(t) = 0$. Therefore $\hat{\phi}(0) \neq 0$. Taking $\omega = 0$ in above formula we get $\sum_{k=-\infty}^{+\infty} h_k = \sqrt{2}$

**Theorem 1.1.** [8] Functions $\phi_k = \phi(t-k)$ constitute orthonormal basis of $V_0$ if and only if

$$\sum_{k=-\infty}^{+\infty} |\hat{\phi}(\omega + 2k\pi)|^2 = 1 \quad a.e. \quad (14)$$
Proof. Changing variables we can easily show that \((\phi_k, \phi_m) = (\phi, \phi_n)\). Then we have

\[
(\phi, \phi_n) = \int_{\mathbb{R}} \phi(t) \overline{\phi(t-n)} dt = \frac{1}{2\pi} \int_{\mathbb{R}} \tilde{\phi}(\omega) e^{-in\omega} \tilde{\phi}(\omega) d\omega =
\]

\[
= \frac{1}{2\pi} \int_{\mathbb{R}} e^{-in\omega} |\tilde{\phi}(\omega)|^2 d\omega = \frac{1}{2\pi} \left( \ldots + \int_{-\pi}^{0} + \int_{0}^{\pi} + \int_{\pi}^{2\pi} \right) =
\]

\[
= \frac{1}{2\pi} \int_{0}^{2\pi} e^{-in\omega} \sum_{k=-\infty}^{+\infty} |\tilde{\phi}(\omega + 2k\pi)|^2 = \delta_{0,n}
\]

As \(e^{-in\omega}\) constitute the orthonormal basis on \([0,2\pi]\) then the integral is equal \(\delta_{0,n}\) if and only if the sum in (14) is equal to 1 almost everywhere.

\[\square\]

**Lemma 1.2.** [8] Under condition of previous theorem

\[|H_0(\omega)|^2 + |H_0(\omega + \pi)|^2 = 1 \quad \text{a.e.} \quad (15)\]

**Proof.** Change \(\omega/2 = \xi\) and insert the definition of \(H_0\) in (14). We have \(\sum_{k=-\infty}^{+\infty} |H_0(\xi + \pi k) \tilde{\phi}(\xi + \pi k)|^2 = 1 \text{ a.e.}\). We divide the last sum in odd and even terms:

\[
\sum_{k=-\infty}^{+\infty} |H_0(\xi + 2\pi k) \tilde{\phi}(\xi + 2\pi k)|^2 + |H_0(\xi + 2\pi k + \pi) \tilde{\phi}(\xi + 2\pi k + \pi)|^2 = 1 \quad (16)
\]

As function \(H_0\) is \(2\pi\) periodic we get:

\[
\sum_{k=-\infty}^{+\infty} |H_0(\omega)|^2 |\tilde{\phi}(\xi + 2\pi k)|^2 + |H_0(\omega + \pi)|^2 |\tilde{\phi}(\xi + 2\pi k + \pi)|^2 = 1 \quad (17)
\]

From the last equation we get \(|H_0(\omega)|^2 + |H_0(\omega + \pi)|^2 = 1 \text{ a.e.}\) \[\square\]

So far we have found some conditions under which \(\phi\) constitute an orthonormal basis of \(V_0\). But how we can build \(\psi\) from \(\phi\) such that \(\psi\) will be orthonormal basis for \(W_0\) and \(\psi_{j,k}(t) = 2^{j/2} \psi(2^j t - k)\) will be orthonormal basis for \(W_j\), \(j \in \mathbb{Z}\)? We have a function \(H_0\). Based on this function we can construct other filters [16]:

- \(H(\omega) = \sqrt(2)H_0(\omega)\ ), the low pass decomposition filter,
- \(G(\omega) = e^{j\omega}H(\omega + \pi)\ ), the high pass decomposition filter,
\[ \tilde{H}(\omega) = \overline{H(\omega)}, \text{the low-pass reconstruction filter,} \]
\[ \tilde{G}(\omega) = e^{-j\omega}H(\omega + \pi), \text{the high-pass reconstruction filter,} \]

We want to decompose function into the detail and approximation parts. It is analogous to decompose signal in low-pass and high-pass parts. We have already function \( H(\omega) \) which is filter function of decomposition of the scaling function

\[ \phi(t) = \sum_k h_k \phi_{1,k}(t) \]

It is logical to assume that \( W_0 \) will be spanned by function \( \psi \) and coefficients of decomposition \( \psi \) by \( \phi_{1,k} \) form filter \( G(\omega) \). Next theorem will prove that our assumption is correct.

**Theorem 1.3.** [8] Assume that \( V_j, j \in \mathbb{Z} \) form an orthogonal MRA of \( L^2(\mathbb{R}) \). There there exists such function \( \psi \) such that \( \psi(t - k) \) build up orthonormal basis of \( W_0 \). We can get such \( \psi \) by

\[ \tilde{\psi}(\omega) = e^{j\omega/2}H_0(\omega/2 + \pi)\tilde{\phi}(\omega/2) \tag{18} \]

which is equivalent to

\[ \psi(t) = \sqrt{2} \sum_k (-1)^{k+1} h_{k-1} \phi(2t - k) \tag{19} \]

Proof of this theorem can be found in [8].

Immediately a one question arises-how one can construct scaling function \( \phi \) from \( h_k \). Next we present three methods of building \( \phi \) [16].

1. **Iteration method**
   Look carefully at the scaling equation:
   \[ \phi(t) = \sqrt{2} \sum h_k \phi(2t - k), \text{ where } \sum |h_k|^2 < \infty. \]
   We can rewrite it as an operator equation.
   Introduce two unitary operators

   Scaling operator \( A_2 \):

   \[ A_2(f)(x) = \sqrt{2} f(2t) \]
Shifting operator $T_{k/2}$:

$T_{k/2}(f)(x) = f(t - k/2), \quad k \in \mathbb{Z}$

We get $\phi(2t - k) = T_{k/2} \circ A_2(\phi)(t)$.

Therefore

$$\phi(t) = \sqrt{2} \sum_{k \in \mathbb{Z}} h_k T_{k/2} \circ A_2(\phi(t)) = F(\phi)(t) \quad (20)$$

where the operator $F : L^2(\mathbb{R}) \to L^2(\mathbb{R})$ is linear combination of $T_{k/2} \circ A_2$,

$F = \sqrt{2} \sum_{k \in \mathbb{Z}} h_k T_{k/2} \circ A_2$. If we write the scaling equation in the form of $\phi(t) = F(\phi)(t)$ we will see the scaling function $\phi$ is fixed point of operator $F$. So we can calculate $\phi(x)$ by iterating $\phi_{k+1} = F(\phi_k)$, $k \to \infty$

By a start point we can get, for example, Haar function. But question about convergence is still open. It is defined by proper choosing of $h_k$.

2. Recursive method

Assume that we know $h_k$ and the function $\phi$ has compact support in $[0, N]$. Then the scaling equation will look like

$$\phi(t) = \sqrt{2} \sum_{k=0}^{N} h_k \phi(2t - k) \quad (21)$$

Suppose we know the values of the scaling function in integers, then from last equation we can find the values in half-times integers, then in quarter-times integers and so on. This way we can get the values of $\phi$ in all points $t = k/2^j, k \in \mathbb{Z}, j = 1, 2, ...$ And because every real number can be approximated by numbers $k/2^j$ with any given accuracy, we can approximate $\phi$ with any required accuracy.

Let’s look at how to find the values of $\phi$ in integers. Substituting $t = k$ in the dilatation equation (9) we get

$$\phi(k) = \sum_{n=0}^{N} h_n \phi(2k - n) = \sum_{s=0}^{N} h_{2^j - s} \phi(s) \quad (22)$$

Last equation can be written in matrix form as $H\phi = \phi$, where $H \in M^{(N+1) \times (N+1)}$ having elements $H_{ks} = \sqrt{2} h_{2k - s}$. Therefore vector $\phi$ is an eigenvector for the corresponding eigenvalue 1.
3. Infinite multiplication

Third method uses scaling equation for Fourier transforms. $\hat{\phi}(\omega) = H_0(\omega/2)\hat{\phi}(\omega/2)$. If in the last equation we put $H_0(\omega/4)\hat{\phi}(\omega/4)$ instead of $\hat{\phi}(\omega/2)$ we will get

$$
\hat{\phi}(\omega) = H_0(\omega/2)H_0(\omega/4)\hat{\phi}(\omega/4),
$$
Continuing this procedure we get $\hat{\phi}(\omega) = c \prod_{j=1}^{+\infty} H_0(\omega/2^j)$, where $c$ constant is limit of $\hat{\phi}$ in 0. If function $\hat{\phi}$ is continuous and normalized by $\hat{\phi}(0) = 1$ then

$$
\hat{\phi}(\omega) = \prod_{j=1}^{+\infty} H_0(\omega/2^j) \quad (23)
$$

Using formula (23) we can find scaling function from $H_0$. Will this infinite multiplication converge?

**Theorem 1.4.** [8] If $\sum_{k} |h_k| < \infty$ and $H_0(0) = 1$ than infinite multiplication

$$
\hat{\phi}(\omega) = \prod_{j=1}^{+\infty} H_0(\omega/2^j) \quad (24)
$$

converges uniformly and absolutely on a compact set.

**Proof.**

$$
|H_0(\omega)| = 1 + |H_0(\omega) - 1| = 1 + |H_0(\omega) - H_0(0)|
= 1 + \frac{1}{\sqrt{2}} \left| \sum_{k \in \mathbb{Z}} h_k(e^{-jk\omega} - 1) \right|
\leq 1 + \sqrt{2} \left| \sum_{k \in \mathbb{Z}} |h_k| |\sin(k\omega/2)| \right|
= 1 + \sqrt{2} \left| \sum_{k \in \mathbb{Z}} |h_k| |\sin(k\omega/2)|/|k\omega/2| \right|
\leq 1 + C|\omega| \leq e^{C|\omega|}
$$

Then

$$
\prod_{j=1}^{+\infty} |H_0(\omega/2^j)| \leq \prod_{j=1}^{+\infty} e^{C|\omega/2^j|} = \exp(\prod_{j=1}^{+\infty} C|\omega/2^j|) \leq C|\omega| \leq e^{C|\omega|}
$$

\[\square\]
1.2 Wavelet construction on $\mathbb{R}^2$

For some applications (eg. integral equations) however we need Multiresolution Analysis on $L^2(\mathbb{R}^2)$. The simplest way is the tensor product of wavelet functions $\psi_{j,k}(t) = 2^{j/2}\psi(2^jt - k)$ which substitute an orthonormal basis of $L^2(\mathbb{R})$. Wavelet functions for $L^2(\mathbb{R}^2)$ given by

$$\Psi_{i,k}^{j,l}(x,y) = \psi_{i,k}(x)\psi_{j,l}(y) \quad (25)$$

Then $\{\Psi_{i,k}^{j,l}(x,y); i, k, j, l \in \mathbb{Z}\}$ constitutes an orthonormal basis of $L^2(\mathbb{R}^2)$.

The second approach is the tensor product of two one-dimensional Multiresolution Analyses. It is based on the fact that the space $L^2(\mathbb{R}^2)$ is direct sum of the spaces $L^2(\mathbb{R})$ and $L^2(\mathbb{R})$

$$L^2(\mathbb{R}^2) = L^2(\mathbb{R}) \bigoplus L^2(\mathbb{R}) \quad (26)$$

Let the functions $\phi$ and $\psi$ generate an orthogonal Multiresolution Analysis of $L^2(\mathbb{R}^2)$. Define $V_0$ as the tensor product of $V_0$ and $V_0$. Then $\{\phi(x - k_2)\phi(y - k_3); k_1, k_2 \in \mathbb{Z}\}$ constitute an orthonormal basis of $V_0$. Other spaces $V_j$ are scaled versions of $V_0$

$$F \in V_0 \iff F(2^j \cdot, 2^j \cdot) \in V_j \quad (27)$$

So we have created Multiresolution Analysis of $L^2(\mathbb{R}^2)$ with properties

$$V_j \subset V_{j+1} \quad (28)$$

$$\bigcap_{j \in \mathbb{Z}} V_j = \{0\} \quad (29)$$

$$\bigcup_{j \in \mathbb{Z}} V_j = L^2(\mathbb{R}^2) \quad (30)$$

As we have already mentioned functions $\{\phi(x - k_2)\phi(y - k_3); k_1, k_2 \in \mathbb{Z}\}$ constitute an orthonormal basis of $V_0$, then

$$\Phi_{j,k_1,k_2} = 2^j \phi(2^jx - k_1, 2^jy - k_2), k_1, k_2 \in \mathbb{Z} \quad (31)$$
is an orthonormal basis of $V_j$.

Let define $W_j$ as the orthogonal complement $V_j$ to $V_{j-1}$. We have:

$$V_{j-1} = V_j \oplus V_{j-1} = (V_j \oplus W_j) \oplus (V_j \oplus W_j) =$$

$$= (V_j \oplus V_j) \oplus [(W_j \oplus V_j) \oplus (V_j \oplus W_j) \oplus (W_j \oplus W_j)] = V_j \oplus W_j$$

Therefore an orthonormal basis of $W_j$ is generated by $\psi(x)\phi(y)$ (for $W_j \oplus V_j$), $\phi(x)\psi(y)$ (for $V_j \oplus V_j$) and $\psi(x)\psi(y)$ (for $W_j \oplus W_j$). So we can define three types of wavelets:

- $\Psi^h(x, y) = \phi(x)\psi(y)$
- $\Psi^v(x, y) = \psi(x)\phi(y)$
- $\Psi^d(x, y) = \psi(x)\psi(y)$

where $h,v,d$ stands for horizontal, vertical and diagonal.

Then

$$\{\Psi^\lambda_{j,k_1,k_2} : k_1, k_2 \in \mathbb{Z}, \lambda \in \{h, v, d\}\}$$

is an orthonormal basis of $W_j$ and

$$\{\Psi^\lambda_{j,k_1,k_2} : j, k_1, k_2 \in \mathbb{Z}, \lambda \in \{h, v, d\}\}$$

constitutes an orthonormal basis for $L^2(\mathbb{R}^2) = \bigoplus_{j \in \mathbb{Z}} W_j$.

Wavelet coefficient of the signal $f(x, y)$ will be of four types:

1. First - approximating coefficients

$$cA_1 = (f, \phi_{j,k,n}) = 2^j \int_{\mathbb{R}} f(x, y) \phi(2^j x - k) \phi(2^j y - n) dx dy$$

others three types of detail(wavelet) coefficients give by:

2. horizontal coefficients

$$cH_1 = (f, \Psi^h_{j,k,n}) = 2^j \int_{\mathbb{R}} f(x, y) \phi(2^j x - k) \psi(2^j y - n) dx dy$$

3. diagonal coefficients

$$cD_1 = (f, \Psi^d_{j,k,n}) = 2^j \int_{\mathbb{R}} f(x, y) \psi(2^j x - k) \phi(2^j y - n) dx dy$$
4. and finally, vertical coefficients

\[ c_{V1} = (f, \Psi_j^{v,k,n}) = 2^j \int_{\mathbb{R}} f(x,y) \psi(2^j x - k) \psi(2^j y - n) dxdy \]

Having \( C_0 \) coefficients, one can calculate \( \{c_{A1}, c_{H1}, c_{V1}, c_{D1}\} \) by two-dimensional discrete wavelet transform formulas [8], then \( \{c_{A2}, c_{H2}, c_{V2}, c_{D2}\} \) and so on.

There \( C_0 \) is defined by

\[ C_0 = (f, \phi_{o,k,n}) = \int_{\mathbb{R}} f(x,y) \phi(x-k) \phi(y-n) dxdy. \]

And the scheme of the signal decomposition can be viewed as

\[ C_0 \rightarrow (c_{A1}, c_{H1}, c_{V1}, c_{D1}) \rightarrow (c_{A2}, c_{H2}, c_{V2}, c_{D2}, c_{H1}, c_{V1}, c_{D1}) \rightarrow \ldots \]

1.3 Wavelet construction on an interval

Wavelet decomposition on an interval is quite a challenging task. One has to construct Multiresolution Analysis for \( L^2([0, 1]) \).

Several construction strategies were proposed to generate wavelets on an interval [12], [1]. The main problem is to retain their vanishing moments [4.3]. All constructions come from the extension a function \( f \) on \([0, 1]\) to whole \( \mathbb{R} \).

In this work we chose to use periodized wavelets on the interval. The idea behind is to extend function outside the interval \([0, 1]\) periodically to \( \mathbb{R} \).

Periodization is defined as follows:

\[ f^\text{per} = \sum_{n=-\infty}^{+\infty} f(t + n) \quad (34) \]

Therefore wavelets and scaling functions will be

\[ \phi_{j,k}(x) = \sum_{n=-\infty}^{+\infty} \phi(x + n) \quad , \psi_{j,k}(x) = \sum_{n=-\infty}^{+\infty} \psi(x + n) \]

After substituting \( \psi_{j,k}(t) = 2^{j/2} \psi(2^j t - k) \) in the previous equations we get periodized wavelets

\[ \psi_{j,k}^\text{per}(x) = \frac{1}{\sqrt{2^j}} \sum_{n=-\infty}^{+\infty} \psi \left( \frac{x - 2^j k + n}{2^j} \right) \quad (35) \]

For every \( j \leq 0 \) there are \( 2^{-j} \) different wavelets indices \( 0 \leq k < 2^{-j} \).
Theorem 1.5. [12] For any \( J \leq 0 \)

\[
\begin{align*}
\{ \psi_{j,k}^{\text{per}} \}_{-\infty < j \leq J, 0 \leq k < 2^{-j}} , \{ \phi_{J,k}^{\text{per}} \}_{0 \leq k < 2^{-J}}
\end{align*}
\]  

constitute an orthonormal basis of \( L^2([0,1]) \).

We will prove Theorem 1.5 with the help of the following lemma

Lemma 1.6. [12] Let \( x(t), y(t) \in L^2(\mathbb{R}) \). If \(( (x, y(\cdot + k)) = 0, \forall k \in \mathbb{Z} \) then

\[
\int_0^1 x(t) y^{\text{per}}(t) dt = 0 \quad (37)
\]

We prove it by using formula (34)

\[
\int_0^1 x^{\text{per}}(t) y^{\text{per}}(t) dt = \int_{-\infty}^{+\infty} x(t) y^{\text{per}}(t) dt = \sum_{k=-\infty}^{+\infty} \int_{-\infty}^{+\infty} x(t) y(t + k) dt = 0
\]

Functions \( \psi_{j,k} \) and \( \phi_{j,k} \) are orthogonal in \( L^2(\mathbb{R}) \). For these we have \(( x, y(\cdot + k) = 0 \). There \( x(t), y(t) \) are wavelets or scaling functions. As a consequence of the lemma (1.6) we see that \( \psi^{\text{per}} \) and \( \phi^{\text{per}} \) are orthogonal in \( L^2([0,1]) \)

For proving that periodized wavelet basis is wavelet basis in \( L^2([0,1]) \) we extend function \( f \) with zero outside the interval \([0,1]\) and decompose function \( f \) using a wavelet basis of \( L^2(\mathbb{R}) \).

\[
f = \sum_{j=-\infty}^{J} \sum_{k=-\infty}^{+\infty} (f, \psi_{j,k}) \psi_{j,k} + \sum_{k=-\infty}^{+\infty} (f, \phi_{J,k}) \phi_{J,k} \quad (38)
\]

We will periodize such extension of the function by (34). For \( t \in [0,1] \) we have \( f^{\text{per}}(t) = f(t) \). \( f^{\text{per}}(t) = 0 \), if \( t \) outside \([0,1]\). Periodizing \( f \) by (34) shows that the function \( f \) is decomposed by wavelet basis of \( L^2([0,1]) \)

The concept of periodized wavelets has one problem in introducing of high amplitude wavelet coefficients near \( t = 0 \) and \( t = 1 \). If \( f(0) \neq f(1) \) we will get discontinuities at the boundaries. This can be seen from the periodic extension of the function and using

\[
\int_0^1 f(t) \psi_{j,k}^{\text{per}}(t) dt = \int_{-\infty}^{+\infty} f^{\text{per}}(t) \psi_{j,k}(t) dt \quad (39)
\]

In case \( f(0) \neq f(1) \), \( f^{\text{per}} \) will have discontinuities at \( t = 0 \) and \( t = 1 \). This will lead to high amplitude wavelet coefficients near boundaries.
2 Discrete wavelet transform

For the calculation of approximation and wavelets coefficients one can numerically find integrals

\[ c_{j,k} = \int_{-\infty}^{+\infty} f(t) \phi_{j,k}(t) dt \]  \hspace{1cm} (40)

and

\[ d_{j,k} = \int_{-\infty}^{+\infty} f(t) \psi_{j,k}(t) dt \]  \hspace{1cm} (41)

However, finding big number of approximation and wavelet coefficients is a time and resource consuming task. Moreover, different integration methods should be adapted for different function types: oscillating, smooth, discontinuous, with high amplitudes, etc. Based on the two-scale relation or the dilatation equation (9) several methods were proposed to overcome such difficulties.

Mallat [12] proposed algorithm for calculation of coefficients of wavelet decomposition without integration using only algebraic operation based on discrete convolution (53), (54).

Supposed we know coefficients for the first level of decomposition for some \( j = J \), \( c_{A_0} = \{c_{j,k}\} \). Problem is to do the first step of decomposition (5), (6)

\[ P_j(f) = \sum_{k \in \mathbb{Z}} c_{j-1,k} \phi_{j-1,k} + \sum_{k \in \mathbb{Z}} d_{j-1,k} \psi_{j-1,k} \]  \hspace{1cm} (42)

without integration.

Let’s express basis functions \( \psi_{j-1,k} \) and \( \phi_{j-1,k} \) by functions \( \phi_{j,k} \) using (4):

\[ \psi_{j-1,k} = \sum_n h_n \phi_{j,n-2k}, \phi_{j-1,k} = \sum_n g_n \phi_{j,n-2k} \]  \hspace{1cm} (43)

where \( \{h_n\}_{n \in \mathbb{Z}} \) and \( \{g_n\}_{n \in \mathbb{Z}} \) are wavelet filters of \( \psi \) and \( \phi \). Then we get

\[ c_{j-1,k} = (f, \phi_{j-1,k}) = (f, \sum_n h_n \phi_{j,n-2k}) = \sum_n h_n c_{j,n-2k} \]  \hspace{1cm} (44)

and

\[ d_{j-1,k} = (f, \psi_{j-1,k}) = (f, \sum_n g_n \psi_{j,n-2k}) = \sum_n g_n c_{j,n-2k} \]  \hspace{1cm} (45)
Finally we get the formulas $c_{j-1,k} = \sum_n h_n c_{j,n-2k}$ and $d_{j-1,k} = \sum_n g_n c_{j,n-2k}$.

The last formulas lead to fast algorithms of calculation of wavelets coefficients (cascade algorithms, Mallat algorithms).

**Note**

Question about initial coefficients.

We still have a problem with calculation of initial coefficients $c_{A_0} = \{c_J, k\}$.

There are several possibilities.

- Direct calculation of integral $c_{J,k} = \int_{\mathbb{R}} f(t) \phi_{J,k} dt$ is a quite time-consuming operation and secondly, if the support of $\phi_{J,k} dt$ is small we cannot provide the desired accuracy.

- Function $f$ is defined as an array of values $y_i = f(x_i)$ in some points $x_i = i \ast h$ and $h = 2^{-J}$. Then the most obvious choice of $c_{J,k}$ is to take function values - $c_{J,k} = y_k$.

### 2.1 Discrete Wavelet Transform in one dimension

There are many ways to implement Discrete Wavelet Transform. We will describe implementation using filter banks and direct implementation. Directly applying the dilatation equation (9) we get formulas for Fast Wavelet Transform.

**Haar decomposition algorithm**

\[
\begin{aligned}
c_j^k &= (c_{j+1}^{2k} + c_{j+1}^{2k+1})/\sqrt{2} \\
d_j^k &= (c_{j+1}^{2k} - c_{j+1}^{2k+1})/\sqrt{2} \text{ for } k = 0, 1, \ldots, 2^j - 1
\end{aligned}
\]  
(46)

**Haar reconstruction algorithm**

\[
\begin{aligned}
c_{j+1}^{2k} &= (c_j^k + c_j^k)/\sqrt{2} \\
d_{j+1}^{2k} &= (c_j^k - c_j^k)/\sqrt{2} \text{ for } k = 0, 1, \ldots, 2^j - 1
\end{aligned}
\]  
(47)

**The Daubechies wavelet db4**

**db4 decomposition algorithm**
\[
\begin{align*}
\begin{cases}
arrow{j}{c^k_j} &= C_0 c^2k_{j+1} + C_1 c^2k_{j+1} + C_2 c^2k_{j+1} + C_3 c^2k_{j+1} + 2 \\
\arrow{j}{d^k_j} &= C_3 c^2k_{j+1} - C_2 c^2k_{j+1} + C_1 c^2k_{j+1} - C_0 c^2k_{j+1}
\end{cases}
\end{align*}
\]

(db4 reconstruction algorithm)

\[
\begin{align*}
\begin{cases}
\arrow{j+1}{c^2k_{j+1}} &= C_3 c^2k_{j} - C_0 d^k_{j} + C_1 c^k_{j} - C_2 d^k_{j} \\
\arrow{j+1}{c^2k_{j+1}} &= C_2 c^k_{j} + C_1 d^k_{j} + C_0 c^k_{j+1} + C_2 d^k_{j+1}
\end{cases}
\end{align*}
\]

where

\[
\begin{align*}
\begin{cases}
C_0 &= \frac{1+\sqrt{3}}{2}, C_1 = \frac{3+\sqrt{3}}{4}\sqrt{2} \\
C_1 &= \frac{3-\sqrt{3}}{4}\sqrt{2}, C_2 = \frac{1-\sqrt{3}}{4}\sqrt{2}
\end{cases}
\end{align*}
\]

Other implementation is implementation using filter banks. We have to introduce first some notation.

Let’s \( s(n) \) be signal

**Definition 2.** The downsampling operator is defined by

\[
(\downarrow)n = s(n) \tag{51}
\]

**Definition 3.** The upsampling operator is defined by

\[
(\uparrow)n = \begin{cases} 
  s(n/2) & \text{if } n \text{ is even} \\
  0 & \text{if } n \text{ is odd}
\end{cases} \tag{52}
\]

**Definition 4.** The convolution of a signal \( s \) and filter \( h \) is defined by

\[
s \ast h[n] = \sum_k s(k)h(n-k) = \sum_k s(n-k)h(k) \tag{53}
\]

**Definition 5.** The circular convolution of a signal \( \tilde{s} \) of length \( N \) and filter \( \tilde{h} \) is defined by

\[
s \otimes h[n] = \sum_{k=1}^{N} s(k)h(n-k) = \sum_{k=1}^{N} s(n-k)h(k) \tag{54}
\]
where \( s = \overline{s}[n \mod N] \) and \( h = \overline{h}[n \mod N] \)

We have 2 formulas

\[
c_{j-1,k} = \sum_n h_n c_{j,n-2k}
\]

and

\[
d_{j-1,k} = \sum_n g_n c_{j,n-2k}
\]

These formulas can be viewed as convolution of the signal and filters and then downsampling:

\[
c_{j-1} = (\downarrow)(c_j \ast h), \quad d_{j-1} = (\downarrow)(d_j \ast g)
\]

For reconstruction we have

\[
c_{j,k} = \sum_n h_{k-2n} c_{j-1,n} + \sum_n g_{k-2n} c_{j-1,n}
\]

Here upsampling of \( c_{j,n} \) is followed by filtering and summation:

\[
c_{j+1} = (\uparrow)(c_j) \ast h + (\uparrow)(d_j) \ast g
\]

### 2.2 Periodized wavelet transform

We have to calculate \( d_{j,k} = (f, \psi^{\text{per}}_{j,k}) \) and \( c_{j,k} = (f, \phi^{\text{per}}_{j,k}) \). Then we have \( d_{j,k} = (f^{\text{per}}, \psi_{j,k}) \) and \( c_{j,k} = (f^{\text{per}}, \phi_{j,k}) \). We will apply convolution and have to take into account the periodicity of \( f \). Therefore approximation and wavelet coefficients are periodical with period \( 2^j \) and each convolution should be changed into circular convolution. And decomposition and reconstruction formulas we be given by:

\[
c_{j-1} = (\downarrow)(c_j \otimes h), \quad d_{j-1} = (\downarrow)(d_j \otimes g)
\]

\[
c_{j+1} = (\uparrow)(c_j) \otimes h + (\uparrow)(d_j) \otimes g
\]
2.3 Discrete Wavelet Transform in two dimensions

There are two basic types of two-dimensional wavelet transform. First one corresponds to anisotropic wavelet basis. Second one corresponds to non-anisotropic wavelet bases. Anisotropic wavelet bases is defined as following: Wavelet functions of $L^2(\mathbb{R}^2)$ with periodic boundary conditions will be

$$
\Psi_{i_1,i_2;j_1,j_2}(x,y) = \psi_{i_1,j_1}(x)\psi_{j_2,i_2}(y)
$$

(58)

Anisotropic wavelet coefficients are computed from $f$ by

$$
(\Psi_{i_1,i_2;j_1,j_2}(x,y), f) = (\overline{\Psi}_{i_1,i_2;j_1,j_2}(x,y), f)
$$

(59)

where $J < i_2, i_2 \leq 0, \forall 0 \leq j_1 \leq 2^{-i_1}, \forall 0 \leq j_2 \leq 2^{-i_2}$ and $\overline{\Psi}_{i_1,i_2;j_1,j_2}(x,y)$ generates a two-dimensional anisotropic basis of $L^2(\mathbb{R}^2)$ which is also a tensorial basis

$$
\Psi_{i_1,i_2;j_1,j_2}[x,y] = \psi_{i_1,i_2}[x]\psi_{j_1,j_2}[y]
$$

(60)

where $\psi_{i,j}$ is one-dimensional discrete wavelet vector.
3 Descretization of operator equations

We want to solve the Fredholm integral equation of the first kind

\[ T(f) = \int_0^1 k(x, y)f(y)dy = g(x) \] (61)

where \( f, g \in L^2([0, 1]) \) and \( k(x, y) \in L^2([0, 1]) \).

Suppose we have an orthonormal basis of \( L^2([0, 1]) \) of Haar or periodized Daubechies wavelets. In such a basis we take a union of wavelet subspaces \( V_j \) then

\[ \bigcup_{j \in \mathbb{Z}} V_j = L^2([0, 1]), \quad V_j \subset V_{j+1} \]

We take \( \psi_{0,0} \cup \{ \psi_{j,k} \}_{0 \leq k < 2^j} = \bigcup_{i \in I} \psi_i \) as orthonormal basis of \( V_j \) in case of Haar wavelet. In case of Daubechies periodized wavelets, we take \( \{ \psi_{j,k}^{\text{per}} \}_{0 < j \leq J, 0 \leq k < 2^j} \cup \{ \phi_{j,k}^{\text{per}} \}_{0 \leq k < 2^j} \) as orthonormal wavelet basis of \( V_J \).

Let us define a projection on the space \( V_j \) as follows:

\[ P_{V_j} f = \sum_{i \in I} (f, \psi_i) \psi_i = \sum_{i \in I} d_i \psi_i \] (62)

where \( d_i = (f, \psi_i) \) are wavelet coefficients.

Applying the projection operator \( P_{V_j} \) to both sides of the equation and taking instead of \( f \), its projection on \( V_J \), we get:

\[ P_{V_j} \int_0^1 k(x, y) \sum_{i \in I} d_i \psi_i(y)dy = P_{V_j} g \] (63)

As \( \int_0^1 k(x, y) \sum_{i \in I} d_i \psi_i dy - g \) belongs to the orthogonal complement of \( V_J \), the previous equation is equivalent to:

\[ \left( \int_0^1 k(x, y) \sum_{i \in I} d_i \psi_i(y)dy - g, \psi_j \right) = 0 \] (64)

for every \( j \in J \).

Taking inner product, we get:

\[ \int_0^1 \int_0^1 k(x, y) \sum_{i \in I} d_i \psi_i(y)dy \psi_j(x)dx - (g, \psi_j) = 0 \] (65)

for every \( j \in I \).
Interchanging sum and integration and taking out coefficients, we get:

\[
\sum_{i \in I} \int_0^1 \int_0^1 k(x, y) \psi_i(y) dy \psi_j(x) dx - \langle g, \psi_j \rangle = 0 \quad (66)
\]

for every \( j \in I \).

Let us define the operator matrix \( K \) as
\[
K_{i,j} = \int_0^1 \int_0^1 k(x, y) \psi_i(y) dy \psi_j(x) dx
\]
and the vector \( g \) as \( g_j = \langle g, \psi_j \rangle \) and the vector of coefficients \( d \) where \( i, j \in I \).

Therefore the integral equation can be written as follows:

\[
Kd = g \quad (67)
\]

As we can notice, the coefficients \( g_j = \langle g, \psi_j \rangle \) are coefficients of wavelet decomposition of the function \( g(x) \) and the matrix entries \( K_{i,j} \) are coefficients of wavelet decomposition of the function \( k(x, y) \).
4 Compactly supported wavelets

4.1 Haar wavelet

In this chapter we introduce two types of compactly supported wavelets - Haar wavelet and Daubechies wavelets. The simplest and earliest wavelet is the Haar wavelet. It was proposed in 1909 by Alfred Haar.

The scaling function \( \phi(t) \) is characteristic function of the interval \([0, 1)\)

\[
\phi(t) = \chi_{[0,1)} = \begin{cases} 
1 & \text{for } t \in [0, 1) \\
0 & \text{otherwise}
\end{cases}
\]  

(68)

Haar wavelet function is given as follows :

\[
\phi(t) = \begin{cases} 
1 & \text{for } t \in [0, 1/2) \\
-1 & \text{for } t \in [1/2, 1) \\
0 & \text{otherwise}
\end{cases}
\]  

(69)

Haar wavelet has the smallest support among orthogonal wavelets. However it has poor approximation properties since it has only one vanishing moment. [sec:db] In 1 section one can see pictures of functions approximated by Haar wavelet.

4.2 Haar transform

As we have explicit representation of Haar wavelet scaling functions, we can calculate wavelet coefficients in one-dimensional or two-dimensional case by numerical integration. We would like to recall first that \( \psi_{j,k}(t) = 2^{j/2}\psi(2^j t - k) \) and \( \phi_{j,k}(t) = 2^{j/2}\phi(2^j t - k) \).

Then \( \psi_{j,k}(t) \) is

\[
\psi_{j,k}(t) = \begin{cases} 
2^{j/2} & \text{for } k/2^j \leq t < (2k + 1)/2^j \\
-2^{j/2} & \text{for } (2k + 1)/2^j \leq t < (k + 1)/2^j \\
0 & \text{otherwise}
\end{cases}
\]  

(70)

and \( \phi_{j,k}(t) \) can be expressed as

\[
\phi_{j,k}(t) = \chi_{[k/2^j,(k+1)/2^j)} = \begin{cases} 
1 & \text{for } k/2^j \leq t < (k + 1)/2^j \\
0 & \text{otherwise}
\end{cases}
\]  

(71)
Then approximation coefficients of finest level can be calculated as follows
\[
c_{J,k} = \int_\mathbb{R} \phi_{J,k}(t) f(t) dt = 2^{J/2} \int_\mathbb{R} \phi(2^J t - k) f(t) dt = 2^{J/2} \int_{k/2^J}^{(k+1)/2^J} f(t) dt \quad (72)
\]

Calculation of the operator matrix entries is little bit more complicated.
Functions \( \{ \phi_{j,k} \}_{k \in \mathbb{Z}} \cup \{ \psi_{j,k} \}_{k \in \mathbb{Z}} \) are Haar basis of \( L^2(\mathbb{R}) \). We need to calculate
\[
K_{i,k,j,l} = \int \int k(x,y) \psi_{i,k}(y) dy \psi_{j,l}(x) dx.
\]

First element of matrix \( K_{0,0,0,0} \) will be
\[
\int_0^1 \int_0^1 k(x,y) \psi_{0,0}(y) dy \phi_{0,0}(x) dx = \int_0^1 \int_0^1 k(x,y) dy dx
\]

On each scale \( j \) we will have \( 2^j \) wavelets \( \psi_{j,k} \) \( k = 0 \ldots 2^j - 1 \).

Entries of operator matrix which corresponds to \( \phi_{0,0}(x) \) can be calculated as follows:
\[
\int_0^1 \int_0^1 k(x, y) \psi_{j,k}(y) dy \phi_{0,0}(x) dx = \int_0^1 \int_0^1 k(x, y) \psi_{j,k}(y) dy dx = 2^{j/2} \left( \int_{(2k+1)/2^{j+1}}^{1} k(x, y) dx dy - \int_{(k+1)/2^{j}}^{(2k+1)/2^{j+1}} k(x, y) dx dy \right)
\]

Entries of operator matrix which corresponds to \( \phi_{0,0}(y) \) can be calculated as follows:

\[
\int_0^1 \int_0^1 k(x, y) \phi_{0,0}(y) dy \psi_{j,k}(x) dx = \int_0^1 \int_0^1 k(x, y) dy \psi_{j,k}(x) dx = 2^{j/2} \left( \int_{(2k+1)/2^{j+1}}^{1} k(x, y) dx dy - \int_{(k+1)/2^{j}}^{(2k+1)/2^{j+1}} k(x, y) dx dy \right)
\]

Other entries can be calculated by formula:

\[
\int_0^1 \int_0^1 k(x, y) \psi_{i,k}(y) dy \psi_{j,l}(x) dx = C \left( \int_{2^{l+1}/2^{j+1}}^{(2k+1)/2^{j+1}} \int_{k/2^i}^{(2k+1)/2^{j+1}} k(x, y) dx \right.\left. dy - \int_{(l+1)/2^i}^{2k+1} \int_{2^{l+1}/2^{j+1}}^{(2k+1)/2^{j+1}} k(x, y) dx dy \right)
\]

where \( C = 2^{(l+j)/2}, dx dy = dx dy \). All these integrals can evaluated by numerical integration. We used iterated integration based on Gauss-Legendre quadrature rule and Newton-Cote quadrature rule.

### 4.3 Daubechies wavelets

First of all we want to introduce concept of vanishing moments and analyze its impact on regularity.

**Definition 6.** Function \( \psi(x) \) has \( N \) vanishing moments if for every \( k = 0 \ldots N - 1 \) following holds

\[
\int_{\mathbb{R}} x^k \psi(x) = 0 \tag{73}
\]
Vanishing moments have big importance in wavelet decomposition. If a wavelet has \( N \) vanishing moments, wavelets coefficients \( d_{j,k} \) of smooth function \( f \) rapidly decrease while \( j \) increases.

Let \( d_{j,k} = (f, \psi_{j,k}) \). Suppose \( f \in C^{N-1}(\mathbb{R}) \). Decompose it using Taylor’s series in neighborhood of \( x_0 \) belonging to support \( \psi_{j,k} \) we get

\[
f(x) = f(x_0) + f'(x_0)(x - x_0) + \ldots + \frac{f^{(N-2)}(x_0)}{(N-1)!}(x - x_0)^{N-1} + r(x)(x - x_0)^{N-1},
\]

where \( r(x) \to 0 \) while \( x \to x_0 \). As function \( \psi \) has \( N \) vanishing moments wavelets coefficients \( d_{j,k} \) (as inner product of wavelets and members of Taylor expansion) will be equal zero and only last member will left

\[
d_{j,k} = (f, \psi_{j,k}) = (r(x)(x - x_0)^{N-1}, \psi_{j,k})
\]

If \( \psi(x) \) has compact support with the increasing of \( j \), support of \( \psi_{j,k}(x) = 2^j \psi(2^j x - k) \) becomes smaller (by \( 2^j \) comparing to beginning). In formula (82) \( x_0 \) belongs to support of \( \psi \). Having in mind that last member \( (r(x)(x - x_0)^{N-1}, \psi_{j,k}) \) is infinitesimal of higher order than \( (x - x_0)^{N-1} \), we get that \( d_{j,k} \) will be smaller than less support of \( \psi_{j,k} \). That means the coefficients are rapidly decreasing while \( j \) increase. If function \( f \) is smooth then wavelets coefficients will be mainly zeros for small scales.

Following theorems connects vanishing moments with decay of wavelet function.

**Theorem 4.1.** [8] Let function \( \psi(t) \in C^{N-1} \) and its derivatives \( \psi^l \) be bounded for \( l \leq N - 1 \) and have following decay:

\[
|\psi(t)| \leq \frac{C}{(1 + |t|^{N+\epsilon})}
\]

If functions \( \psi_{j,k}(t) = 2^j \psi(2^j t - k) \) constitute orthonormal families in \( L^2(\mathbb{R}) \) then

\[
\int_{\mathbb{R}} t^l \psi(t) dt = 0
\]
Conclusion 4.2. Let function $\psi(t) \in \mathbb{C}^{N-1}$ has compact support. If functions $\psi_{j,k}(t) = 2^{j/2}\psi(2^j t - k)$ constitute orthonormal families in $L^2(\mathbb{R})$ then

$$\int_{\mathbb{R}} l^j \psi(t) \, dt = 0$$

As we can see if orthogonal wavelet $\psi \in \mathbb{C}^{N-1}$ with compact support, it has $N$ vanishing moments. Note, if the orthogonal wavelet has compact support, it can not belong to $\mathbb{C}^\infty$.

How will the function $H_0(13)$ and its coefficients look like, if $\psi$ has compact support? Answer can be found in the following theorems.

Theorem 4.3. Let the function $\psi$ constitutes an orthonormal basis of $W_0$ with corresponding MRA. And let function $\psi$ and its derivatives $\psi^l$ be bounded for $l \leq N - 1$ and have following decay:

$$|\phi(t)|, |\psi(t)| \leq \frac{C}{(1 + |t|^{N+\epsilon})}$$  \hspace{1cm} (77)

then symbol $H_0(\omega)$ of $\psi$ can be decomposed as

$$H_0(\omega) = (1 + e^{-i\omega})^N L(\omega)$$  \hspace{1cm} (78)

where $L(\omega)$ is $2\pi$ periodic function belonging to $\mathbb{C}^{N-1}$

Theorem 4.4. Let the function $\psi$ and $\phi$ generates an orthonormal basis of wavelets with corresponding MRA. If function $\psi$ has $N$ vanishing moments and

$$|\phi(t)|, |\psi(t)| \leq \frac{C}{(1 + |t|^{N+\epsilon})}$$  \hspace{1cm} (79)

then filter coefficients $\{h_k\}$ of scaling function have properties:

- $\sum_n h_n = \sqrt{2}$
- $\sum_n (-1)^n n^k h_n = 0, k = 0, 1, \ldots, N - 1$
We want to construct an orthogonal wavelets basis with compact support and having \( m \) vanishing moments.

As we know, \( \phi \) can be reconstructed from the formula (23):

\[
\hat{\phi}(\omega) = \prod_{k=1}^{\infty} H_0(\omega/2^k) \tag{80}
\]

and such function should satisfy:

\[
|H_0(\omega)|^2 + |H_0(\omega + \pi)|^2 = 1 \tag{81}
\]

We will look first for the function \( M_0(\omega) = |H_0(\omega)|^2 \) which satisfies:

\[
M_0(\omega) + M_0(\omega + \pi) = 1 \tag{82}
\]

We have

\[
M_0(\omega) = |H_0(\omega)|^2 = \left( \frac{|1 + \cos(\omega) - i \sin(\omega)|^2}{4} \right)^N |T(\omega)|^2 = \left( \frac{1 + \cos(\omega)}{2} \right)^N |T(\omega)|^2 = \left( \cos^2(\omega/2) \right)^N L(\omega)
\]

where \( L(\omega) = |T(\omega)|^2 \) is also trigonometric polynomial of \( \cos(\omega) \).

Indeed, trigonometric polynomial \( T(\omega) \) with powers of \( e^{-i\omega} \) has real coefficients. Therefore \( T(\omega) = \overline{T(-\omega)} \) and then \( L(\omega) = |T(\omega)|^2 = T(\omega)T(-\omega) \) is \( 2\pi \) periodic function and will be polynomial of \( \cos(\omega) \). And because \( 1 + \cos \omega = 2 \sin^2(\omega/2) \), this polynomial can be written as polynomial \( P(\sin^2(\omega/2)) \). Then we have

\[
M_0(\omega) = \left( \cos^2(\omega/2) \right)^N P(\sin^2(\omega/2)) = \left( 1 - \sin^2(\omega/2) \right)^N P(\sin^2(\omega/2))
\]

and hence

\[
M_0(\omega + \pi) = \left( \sin^2(\omega/2) \right)^N P(1 - \sin^2(\omega/2))
\]

So we are searching for the function \( M_0(\omega) \) which looks like

\[
M_0(\omega) = \left( 1 - \sin^2(\omega/2) \right)^N P(\sin^2(\omega/2)) \tag{83}
\]

Lets define \( y \) as \( \sin^2(\omega/2) \).

Combining all this together we get

\[
(1 - y)^N P(y) + y^N P(1 - y) = 1 \tag{84}
\]
For finding such polynomial \( P(y) \) using the last equation we need the following lemma

**Lemma 4.5.** [8] If \( P_1(y) \) and \( P_2(y) \) are polynomials of power \( n \) and \( m \) without common zeros, then exist unique polynomials \( Q_1(y) \) and \( Q_2(y) \) of powers \( n-1 \) and \( m-1 \) such that

\[
P_1(y)Q_1(y) + P_2(y)Q_2(y) = 1 \tag{85}
\]

Let’s use this lemma for our polynomials \( P_1 = (1-y)^N \) and \( P_2 = y^N \). Then there exists unique polynomials \( Q_1(y) \) and \( Q_2(y) \) of power less than \( N \) such that

\[
(1-y)^N Q_1(y) + y^N Q_2(y) = 1 \tag{86}
\]

Changing \( y \) to \( 1-y \) we get \( y^N Q_1(1-y) + (1-y)^N Q_2(1-y) = 1 \). From uniqueness of \( Q_1 \) and \( Q_2 \) we get that \( Q_2(y) = Q_1(1-y) \). Then

\[
(1-y)^N Q_1(y) + y^N Q_1(1-y) = 1 \tag{87}
\]

Let’s find \( Q_1 \) explicitly

\[
Q_1(y) = \frac{1}{(1-y)^N (1-y^N Q_1(1-y))}
\]

Express first multiplicator by the binomial theorem:

\[
\frac{1}{(1-y)^N} = \sum_{k=0}^{\infty} \frac{(N+k-1)(N+k-1)\ldots(N+1)(N)}{k!} y^k = \sum_{k=0}^{\infty} C_{N+k-1}^k y^k
\]

then

\[
Q_1(y) = \frac{1}{(1-y)^N (1-y^N Q_1(1-y))} = \sum_{k=0}^{N-1} C_{N+k-1}^k y^k + O(y^N) = \sum_{k=0}^{N-1} C_{N+k-1}^k y^k
\]

because the power of \( Q_1 \) less than \( N \).

Then the desired solution \( P(y) \) of equation (84) of power \( N-1 \) will look like

\[
P_{N-1}(y) = \sum_{k=0}^{N-1} C_{N+k-1}^k y^k = 1 + Ny + \frac{N(N-1)}{2!} y^2 + \frac{N(N-1)(N-2)}{3!} y^3 + \ldots \tag{88}
\]
So far we find $M_0(\omega) = |H_0(\omega)|^2$. For finding $H_0$ we have to take square root of $M_0(\omega)$. And way to do that will be explain in following

**Lemma 4.6.** [8] Let $A(\omega)$ be a positive trigonometric polynomial having representation

$$A(\omega) = \sum_{k=0}^{N} a_k \cos k\omega, a_k \in \mathbb{R}$$

then there exists trigonometric polynomial $B(\omega)$ :

$$B(\omega) = \sum_{k=0}^{N} b_k e^{jk\omega}, b_k \in \mathbb{R}$$

such that $|B(\omega)|^2 = A(\omega)$

**Proof.** Function $A(\omega)$ can be represented as a polynomial of $\cos \omega$, $A(\omega) = p_A(\cos \omega)$. Since $|B(\omega)|^2 = A(\omega)$, we need to take square root on $A(\omega)$. If $z = e^{-i\omega}$ then $\cos \omega = (e^{-j\omega} + e^{j\omega})/2 = (z + z^{-1})/2$. Therefore

$$A(\omega) = p_A(\cos \omega) = p_A(\frac{z + z^{-1}}{2}) = p_A(z)$$

Polynomial $p_A(z)$ has real coefficients and $p(z) = p(z^{-1})$ therefore their complex roots will look like $\{z_j, z_j^{-1}, z_j^{-2}\}$ and real will look like $\{r_j, r_j^{-1}\}$ So we can write

$$p_A(x) = cz^{-N} \prod_j (z - z_j)(z - z_j^{-1})(z - z_j^{-2}) \prod_j (z - r_j)(z - r_j^{-1})$$

There we have to separate case when $z_k = e^{ia^k} = \overline{z_k^{-1}}$ and instead of quads complex roots we will have pairs. On the unit circle $z = e^{-i\omega}$ we have

$$|(e^{i\omega} - z_0)(e^{i\omega} - \overline{z_0^{-1}})| = |z_0|^{-1}|e^{i\omega} - z_0|^2$$

Therefore

$$A(\omega) = |A(\omega)| = |P_A(e^{-i\omega})| = |c||\prod_j \frac{1}{|z_j|}(e^{-i\omega} - z_j)(e^{-i\omega} - \overline{z_j^{-1}})|^2| \prod_k (e^{-i\omega} - e^{ia_k})(e^{-i\omega} - e^{-ia_k})|^2 \prod_l \frac{1}{|r_l|} = |e^{-i\omega} - r_l|^2 = |B(\omega)|^2$$
And we can take $B(\omega)$ as

$$B(\omega) = |c|^{1/2} \prod_j \frac{(e^{-i\omega} - z_k)(e^{-i\omega} - \bar{z}_j^{-1})}{|z_j|} \prod_k (e^{-i\omega} - e^{i\alpha_k})(e^{-i\omega} - e^{-i\alpha_k}) \prod_l \frac{e^{-i\omega} - r_l}{|r_l|^{1/2}}$$

(89)

Remarks

1) $B(\omega)$ is constructed from $A(\omega)$ but not uniquely. For example, one can multiply $(B(\omega))$ by $e^{-jk\omega}, k \in \mathbb{Z}$.

2) Such procedure of finding $B(\omega)$ from $A(\omega) = |B(\omega)|^2$ is called spectral factorization. Taking square roots of $A(\omega)$ is selection of half of roots of $A(\omega)$ and construction of $\prod_j (z - z_k)$.

3) For every quad of complex roots $\{z_j, \bar{z}_j, z_j^{-1}, \bar{z}_j^{-1}\}$ we take a pair $\{z_j, \bar{z}_j\}$ or $\{z_j^{-1}, \bar{z}_j^{-1}\}$ such that both roots are inside the unit circle on the complex plane. For each pair of $\{r_j, r_j^{-1}\}$ we select one pair inside or outside the unit circle.

4) If we have orthogonal wavelet $\phi$ with compact support, function $H_0(\omega)$ is trigonometric polynomial satisfying $|H_0(\omega)|^2 + |H_0(\omega + \pi)|^2 = 1$. If we require $N$ vanishing moments for $\psi$ function $H_0(\omega)$ will have representation

$$H_0(\omega) = \left(1 + \frac{e^{-i\omega}}{2}\right)^N T(\omega)$$

where $T(\omega)$ is trigonometric polynomial. Let $M_0(\omega) = |H_0(\omega)|^2$ then $M_0(\omega) = (1 - \sin^2(\omega/2))^N P(\sin^2(\omega/2))$ were $P$ is polynomial of power $N - 1$. Function $H_0(\omega)$ is constructed from spectral factorization of polynomial $M_0(\omega)$. Having in mind the $\sin^2\omega/2 = (1 - \cos\omega/2) = (1 - (z + z^{-1})/2)/2$ where $z = e^{-i\omega}$ we get

$$H_0(\omega) = \left(1 + \frac{z}{2}\right)^N \prod_{j=1}^{N-1} (z - z^j)$$

where $j$ is index for chosen roots.

Below we can see plot of function $\phi$ (Figure 2) and $\psi$ (Figure 3) of db2 wavelet.

Examples of exact calculation of $h_k$ can be found in [17].

For practical computation we will use already calculated tables [12] of values
Figure 2: Graph of scaling function of db2 wavelet

of $h_k$ for dbN wavelets $N = 2 \ldots 10$. 
Figure 3: Graph of wavelet function of db2 wavelet
5 Description of algorithms

5.1 Numerical integration

For calculating approximation and wavelets coefficients, we need to evaluate the integrals $\int f(t)\phi(t)dt$ and $\int f(t)\psi(t)dt$. So we approximate it numerically by

$$I = \int_a^b f(t)dt \approx \int_a^b P(t)dt$$  \hspace{1cm} (90)

where $P$ is the approximation function. This process is called numerical integration or quadrature rule. In our program, we use two types of quadrature rules- Newton-Cotes and Gauss-Legendre quadrature [10].

5.2 Newton-Cotes

We first discretize interval $[a, b]$ into $n$ equidistant intervals with lengths $h = (b - a)/n$ and $n + 1$ points $x_i = a + kh$. Then we will interpolate the definite integral by formula

$$\int_a^b f(t)dt = (b - a) \sum_{k=0}^{n} C_n^k f(a + kh)$$  \hspace{1cm} (91)

This formula is called Newton-Cotes formula. Coefficients $C_n^k$ are:

$$C_n^k = \frac{A_k}{b - a} = \frac{1}{b - a} \int_a^b \frac{p(x)}{(x - a - kh)p'(a + kh)}dx$$

where $p(x) = (x - a)(x - a - h)\ldots(x - a - nh)$. Letting $x = a + th$ and substituting into the last formula we get

$$C_n^k = \frac{(-1)^{n-k}}{nk!(n-k)!} = \int_0^n t(t-1)(t-2)\ldots(t-k+1)(t-k-1)\ldots(t-n)dt$$

$C_n^k$ are called Cartesian numbers. Their values can be found in Appendix B. There are two types of Newton-Cotes formula.

- Closed-type Newton-Cotes quadrature Rule If we include points $a, b$ in our rule we get

$$\int_a^b = I_a^b(f) = (b - a) \sum_{k=0}^{n} w_k f(x_k)$$  \hspace{1cm} (92)
where weights are given in Appendix B, and $\sum_{k=0}^{n} w_k = 1$.

- Open-type Newton-Cotes quadrature Rule
  If we exclude points $a, b$ in our rule we get
  \[
  \int_{a}^{b} f(t) = (b - a) \sum_{k=0}^{n-1} w_k f(a + ik)
  \]
where weights are given in Appendix B.

5.3 Gauß-Legendre quadrature rule

In Gaussian quadrature rule, we express the function $f(t)$ approximately as $f(t) = W(t)g(t)$

\[
\int_{-1}^{1} f(t) dt = \int_{-1}^{1} W(t)g(t) dt \approx \sum_{i=1}^{n} w_i g(x_i)
\]

where $W(t)$ is called weight function and $x_i$ are not equally spaced. Special selections of $W(t)$ lead to different types of quadrature rule. If we select $W(t) = (1 - t)^{\alpha}(1 + t)^{\beta}$, the rule is called Gauß-Jacobi rule.

If we select $W(t) = 1$, the rule is called Gauß-Legendre rule.

For Gauß-Legendre rule, we have

\[
\int_{a}^{b} f(y) dy = \frac{b - a}{2} \sum_{i=1}^{n} w_i f(y_i)
\]

where $w_i$ are weights and $y_i = \left(\frac{b-a}{2}\right)x_i + \frac{b-a}{2}$. The weights are given by $w_i = \frac{2}{(1-x_i^2)[P_n'(x_i)]^2}$. Here, $P_n(x)$ is the Legendre polynomial and $x_i$ is i-th zero of $P_n$. Legendre polynomials compose an orthogonal system in $L^2[-1, 1]$ created by polynomials $x^i$. The have explicit representation:

\[
P_n(t) = \frac{1}{2^n n!} \frac{d^n}{dt^n}[(t^2 - 1)^n]
\]

Applying the binomial theorem to $(t^2 - 1)^n$ and differentiating $n$ times, one can calculate
\[ P_n(t) = \sum_{j=0}^{N} (-1)^j \frac{(2n - 2j)!}{2^n j!(n - j)!(n - 2j)!} t^{n-2j} \quad \text{(97)} \]

where \( N = n/2 \) for even \( n \) and for odd \( N = (n - 1)/2 \).

For practical purposes, one can use precalculated values of \( x_i \) and \( w_i \). Example of a such table can be found in Appendix D.

### 5.4 Adaptive integration

One approach to calculate integrals is an automatic integration. If we do not match the desired error estimate, we will keep halving stepsize until we get one. It may work, but will required more work, than any approaches specially adapted to the particular function. A better approach is adaptive integration where integration domain is rearranged to respond specific behavior of integrand.[9]

First, we calculate integral on the whole interval. If we do not match the desired error tolerance, then divide the original interval into two parts and calculate integrals again in each area. Then if the sum of errors is again exceeds the error tolerance, we half the part with the largest error and continue till we met the required tolerance.

Such approach has some problems. Firstly, we may not meet the error tolerance. For example, if have noisy data. So this approach will keep halving the integration domains and in each parts will have a big error. It will continue until it breaks down and shows an error that we can not half anymore.

Second problem that we will get some error estimate from adaptive integration but it might be completely wrong. For example, we have some peaks in our domain of integration but it these are not included in our halving interval.

Let us now describe recursive adaptive integration algorithm. Suppose that \( \text{basicInt}(\text{fun}, a, b, \text{tol}) \) is function which evaluates integral of function \( \text{fun} \) on interval \([a, b]\) with predescribed tolerance via Newton-Cote or Gauss-Legendre quadrature rule. Then we can write the adaptive integration algorithm as follows. We denote by \( \text{adaptInt} \) function that does adaptive integration.
middlePoint = (a + b)/2
Q = basicInt(fun, a, b)
Q1 = basicInt(fun, a, middlePoint)
Q2 = basicInt(fun, middlePoint, b)
Qnew = Q1 + Q1
if Q - Qnew ≤ tol then
    integral = Qnew
else
    adaptInt(fun, a, middlePoint, tol)
    adaptInt(fun, a, middlePoint, tol)
    integral = Q1 + Q2
end if

5.5 Numerical algorithms

In this work we use two types of compactly supported wavelets - Haar and Daubechies wavelets. (Section 4) In the case of Daubechies wavelets, we use periodized wavelet transform (subsection 2.2). Therefore we describe algorithms for solving integral equation of the first kind with the help of these two methods. For wavelet decomposition of a signal, one has to find approximation coefficients on the finest level. There are three cases - two with Haar wavelet and one with Daubechies:

1. Function is given explicitly. So we can calculate $c_{J,k}$ with help of numerical integration either with Newton-Cotes formulas or Gauss-Legendre.

2. Function is given by discrete values on equidistant grid. Therefore we can use only Newton-Cotes quadrature.
3. In the case of Daubechies wavelets we can use a function given by
discrete values and we take values of the function at selected points as
the initial wavelet coefficients.

For other possibilities of choice of computation of initial coefficients, we refer
the reader to section 2.

5.5.1 Wavelet decomposition of a function

For wavelet decomposition of the right-hand side, we used Haar wavelet trans-
form and Daubechies wavelet transform with periodized adaptation on the
boundary.

1. Haar wavelet transform algorithm

- Decomposition algorithm
  Let \( f \) be a function belonging to \( L^2([0,1]) \) and \( J \) fixed integer.
  First of all, we compute \( 2^J \) coefficients \( c_{J,k} \) using methods from
  section 2. Beginning from \( c_{J,k} \), we can compute all approximation
  coefficients \( c_{j,k} \) and wavelet coefficients \( d_{j,k} \) with help of the fol-
  lowing algorithm.

  \[
  \text{for } j = 0 \text{ to } J - 1 \text{ do} \\
  \quad \text{for } k = 0 \text{ to } 2^j - 1 \text{ do} \\
  \quad \quad c_j^k \leftarrow (c_{2j+1}^k + c_{2j+1}^{k+1}) \sqrt{2} \\
  \quad \quad d_j^k \leftarrow (c_{2j+1}^k - c_{2j+1}^{k+1}) \sqrt{2} \\
  \quad \text{end for} \\
  \text{end for} \\
  \]

  Let’s compute the computational cost of this algorithm. We see
  that \( d_j^k \) are not used in the next step of the algorithm. We need
  only \( 2^j \) values of \( c_j^k \) to compute \( c_{j-1}^k \) and \( d_{j-1}^k \). The Computational
  cost of algorithm at step \( j \) is evaluating \( 2^j \) coefficients of \( c_j^k \) and
  \( d_j^k \). So it will be \( 2^{j+1} \) operations. Altogether it will be:

  \[ 2^{J+1} + \ldots + 2^j + \ldots + 2^2 + 1 = 4 \times 2^J \]

- Reconstruction algorithm
  Assume that we know all the approximation coefficients \( c_0^0 \) or the
mean values of \( f \) on \([0, 1]\) and wavelet coefficients \( d_j^k, j = 0 \ldots J-1 \).

Then we can compute all \( c_j^k \) by the following algorithm:

\[
\text{for } 0 \text{ to } j = J - 1 \text{ do}
\]
\[
\quad \text{for } k = 0 \text{ to } 2^j - 1 \text{ do}
\]
\[
\quad \quad c_{2^j+1}^{2k} \leftarrow (c_j^k + d_j^k) \sqrt{2}
\]
\[
\quad \quad d_{2^j+1}^{2k+1} \leftarrow (c_j^k - d_j^k) \sqrt{2}
\]
\[
\quad \text{end for}
\]
\[
\text{end for}
\]

Again the computational cost of this algorithm is \( 2^{j+1} \) operations.

2. Daubechies periodized wavelet transform algorithm

Let \( f \) be a function belonging to \( L^2([0,1]) \) and \( J \) fixed integer. We will take values of function \( f \) on interval \([0,1] \) at \( t = k/2^j \) points as the initial coefficients \( c_j^k \). Suppose that we know wavelets decomposition low-pass and high pass filters and reconstruction low-pass and high-pass filters [16]. In our pseudocode we define them as

- Low-pass decomposition filter -getDecLowPassFilter(waveletName)
- High-pass decomposition filter -getDecHighPassFilter(waveletName)
- Low-pass reconstruction filter -getRecLowPassFilter(waveletName)
- High-pass reconstruction filter -getRecHighPassFilter(waveletName)

Since we use periodized wavelets basis, we need to extend the function periodically.

Let us denote the periodical extension of a signal \( s \) by length \( n \) as \( \text{extendPeriodically}(s,n) \) and the convolution of two signals \( s_1 \) and \( s_2 \) by \( \text{conv}(s_1,s_2) \). Downsampling and upsampling shall be denoted by \( \text{down} \) and \( \text{up} \) and function values by \( \text{signal} \)

Finally, we can describe periodized wavelet transform (2.2) algorithms as follows:

- **Decomposition algorithm**

\[
\text{cfir} \leftarrow \text{getDecLowPassFilter}(\text{waveletName})
\]
\[
\text{dfir} \leftarrow \text{getDecHighPassFilter}(\text{waveletName}) \\
y \leftarrow \text{signal} \\
\text{for } 0 \text{ to } j = J - 1 \text{ do} \\
\hspace{1em} c \leftarrow \text{circularConvolution}(y, \text{cfir}) \\
\hspace{1em} d \leftarrow \text{circularConvolution}(y, \text{cfir}) \\
\hspace{1em} y \leftarrow c \\
\text{end for}
\]

where \text{circularConvolution}(\text{signal}, \text{filter}) is a function which calculate circular convolution

\[
\text{filterSize} \leftarrow \text{length}(\text{filter}) \\
\text{periodicExtension} \leftarrow \text{getsextendPeriodically}(\text{signal}, \text{filterSize}/2) \\
\text{circConv} \leftarrow \text{conv}(\text{periodicExtension}, \text{filter}) \\
\text{circConv} \leftarrow \text{circConv}(\text{filterSize} : \text{circConvSize} - \text{filterSize} + 1) \\
\text{circConv} \leftarrow \text{down}(\text{circConv})
\]

- **Reconstruction algorithm**

Reconstruction algorithm with periodized wavelet is the same as reconstruction algorithm with ordinary wavelet. Algorithm is described in following pseudocode.

\[
\text{cfir} \leftarrow \text{getRecLowPassFilter}(\text{waveletName}) \\
\text{dfir} \leftarrow \text{getRecHighLowPassFilter}(\text{waveletName}) \\
\text{for } j=0 \text{ to } J - 1 \text{ do} \\
\hspace{1em} d \leftarrow \text{getWaveletCoefficients}(j) \\
\hspace{1em} c \leftarrow \text{up}(c) \\
\hspace{1em} d \leftarrow \text{up}(c) \\
\hspace{1em} c \leftarrow \text{circularConvolution}(c, \text{cfir}) \\
\hspace{1em} d \leftarrow \text{circularConvolution}(d, \text{cfir}) \\
\hspace{1em} c \leftarrow c + d \\
\text{end for}
\]

5.5.2 Discretization of an integral operator

Integral operator decomposition algorithm with Daubechies periodized wavelets

Let \( k(x, y) \) belongs to \( L^2([0, 1]^2) \) and \( J \) is fixed integer. We discretized
Integral operator decomposition algorithm with Haar wavelets

Let the kernel function \( k(x, y) \) evaluated on wavelet bases \( \psi(x)\psi(y) \) we can use the following simple algorithm:

\[
\begin{align*}
&i \gets \text{getRowNumber}(D) \\
&j \gets \text{getColumnNumber}(D) \\
&\text{for } j = 0 \text{ to } J - 1 \text{ do} \\
&\quad \text{tempTable}(i,j) \gets \text{periodizedWaveletTransform}(D(:, j), J, \text{waveletName}) \\
&\text{end for} \\
&\text{for } i = 0 \text{ to } J - 1 \text{ do} \\
&\quad \text{tensor}(i,:) \gets \text{periodizedWaveletTransform}(\text{tempTable}(i,:), J, \text{waveletName}) \\
&\text{end for}
\end{align*}
\]

According to the formulas provided in 4.2, the matrix entries \( K_{0,0,j,k} \) which correspond to \( \phi_{0,0}(y) \), can be calculated as follows:

\[
\begin{align*}
\text{integral1} &\gets \text{numInt2D}(\text{kernel}, 0, 1, k \cdot 2^{-i}, (k + 1/2) \cdot 2^{-i}) \\
\text{integral2} &\gets \text{numInt2D}(\text{kernel}, 0, 1, (k + 1/2) \cdot 2^{-i}, (k + 1) \cdot 2^{-i}) \\
\text{return} &\quad 2^{i/2} \times (\text{integral1} - \text{integral2})
\end{align*}
\]

According to the formulas provided in 4.2, the matrix entries \( K_{j,l,0,0} \) which correspond to \( \phi_{0,0}(x) \), can be calculated as follows:

\[
\begin{align*}
\text{integral1} &\gets \text{numInt2D}(\text{kernel}, l \cdot 2^{-j}, (l + 1/2) \cdot 2^{-j}, 0, 1) \\
\text{integral2} &\gets \text{numInt2D}(\text{kernel}, (l + 1/2) \cdot 2^{-j}, (l + 1) \cdot 2^{-j}, 0, 1) \\
\text{return} &\quad 2^{j/2} \times (\text{integral1} - \text{integral2})
\end{align*}
\]

Calculation of ordinary entries \( K_{i,k,j,l} \) is more complicated:

\[
\begin{align*}
\text{integral1} &\gets \text{numInt2D}(\text{kernel}, l \cdot 2^{-j}, (l + 1/2) \cdot 2^{-j}, k \cdot 2^{-i}, (k + 1/2) \cdot 2^{-i}) \\
\text{integral2} &\gets \text{numInt2D}(\text{kernel}, (l + 1/2) \cdot 2^{-j}, (l + 1) \cdot 2^{-j}, k \cdot 2^{-i}, (k + 1/2) \cdot 2^{-i}) \\
\text{integral3} &\gets \text{numInt2D}(\text{kernel}, l+1/2)2^{-j}, (l+1)2^{-j}, k2^{-i}, (k+1/2)2^{-i}) \\
\text{integral4} &\gets \text{numInt2D}(\text{kernel}, (l+1/2)2^{-j}, (l+1)2^{-j}, (k+1/2)2^{-i}, (k+1/2)2^{-i})
\end{align*}
\]
\[1) \ast 2^{-i})\]
\[\text{return } 2^{i/2} \ast 2^{i/2}(\text{integral1} - \text{integral2} - \text{integral3} + \text{integral4})\]

Now we can evaluate the first row and column of the operator matrix \text{firstRow}(kernel, j, l) and \text{firstColumn}(kernel, i, k) respectively as well as the ordinary matrix entries \text{matrixEntry}(kernel, i, k, j, l).

Firstly, we will evaluate the first column of operator matrix.

\[K_{1,1} \leftarrow \text{numInt2D}(kernel, 0, 1, 0, 1)\]
\[\text{for } i=0 \text{ to } J-1 \text{ do}\]
\[\quad \text{for } k=0 \text{ to } 2^i - 1 \text{ do}\]
\[\quad \quad n \leftarrow 2^i + k\]
\[\quad \quad K_{n+1,1} \leftarrow \text{firstColumn}(kernel, i, k)\]
\[\text{end for}\]
\[\text{end for}\]

Then we can calculate the first row of operator matrix.

\[\text{for } j=0 \text{ to } J-1 \text{ do}\]
\[\quad \text{for } l=0 \text{ to } 2^j - 1 \text{ do}\]
\[\quad \quad n \leftarrow 2^j + l\]
\[\quad \quad K_{1,n+1} \leftarrow \text{firstRow}(kernel, j, l)\]
\[\text{end for}\]
\[\text{end for}\]

And finally left entries of operator matrix K

\[\text{for } i=0 \text{ to } J-1 \text{ do}\]
\[\quad \text{for } k=0 \text{ to } 2^i - 1 \text{ do}\]
\[\quad \quad \text{for } j=0 \text{ to } J-1 \text{ do}\]
\[\quad \quad \quad \text{for } l=0 \text{ to } 2^j - 1 \text{ do}\]
\[\quad \quad \quad \quad n1 \leftarrow 2^i + k\]
\[\quad \quad \quad \quad n2 \leftarrow 2^j + l\]
\[\quad \quad \quad \quad K_{n1+1,n2+1} \leftarrow \text{matrixEntry}(kernel, i, k, j, l)\]
\[\quad \quad \text{end for}\]
\[\quad \text{end for}\]
\[\text{end for}\]
5.5.3 Solving integral equation

Combining these altogether, we can now solve an integral equation. First we calculate the initial coefficients via numerical integration or as discrete points. Then we do wavelet decomposition of the right-hand side. Then we obtain wavelet decomposition of the integral equation. Solving a linear system or using regularization methods, we can get wavelet coefficients of the solution. Applying the inverse wavelet transform, we will find the solution.

\[
\int_0^1 \int_0^1 k(x, y) f(y) d(y) = g(y)
\]

\[
\begin{align*}
\text{initialCoeff} & \leftarrow \text{calculateInitialCoeff}(g, J, \text{waveletName}) \\
\text{rhsCoeff} & \leftarrow \text{waveletDecompose}(\text{initialCoeff}, J, \text{waveletName}) \\
K & \leftarrow \text{tensorMatrix}(\text{kernel}, J, \text{waveletName}) \\
\text{solutionCoeff} & \leftarrow \text{solveLinerSystem}(K, \text{rhsCoeff}) \\
\text{solution} & \leftarrow \text{waveletDecompose}(\text{solutionCoeff}, J, \text{waveletName})
\end{align*}
\]
6 Numerical results

As we use for our decomposition, two types of wavelets: Haar and Daubechies, we divide this chapter into two parts. In the first part we analyze the approximation properties of the Haar wavelet. We use three kernel functions with corresponding solutions and the right-hand sides of corresponding equations for analyzing the approximation error arising from different types of wavelet bases -Haar and Daubechies. Information about the condition number is quite helpful while solving liner systems so we try to look at the condition numbers for different types of wavelets. We want to compute an operator matrix in reasonable time, so we provide information about running time for different types of wavelets. Finally, we look at how the entries of the operator matrix behave.

6.1 Haar wavelet

We select several functions and will use them for analyzing of approximation properties.

1. \( f \in C^\infty \). \( f(t) = e^t \).
2. \( f \) is discontinuous at 0. \( f(t) = \frac{1}{t} \), \( f \notin L^2([0, 1]) \).
3. \( f \) is polynomial. \( f(t) = t^3 - 3/2t + 1 \).
4. \( f \in C^2([0, 1]) \) but \( f \notin C^3([0, 1]) \). \( f(t) = |t - 1/2|^3 \).
5. \( f \in C^9([0, 1]) \) but \( f \notin C^{10}([0, 1]) \) and has a shape close to Haar shape, \( f(t) = |t - 1/2|^{10} \).

First of all, we want to see the approximation properties of the Haar wavelet. According to theory, it doesn’t have good approximation properties. In Table 1, we can see \( L^2 \) error norms on different scales \( J \) for the above mentioned functions. For each scale, we use \( 2^{10} \) points for approximation. Integrations are done with tolerance 10^{-6} and Gauss-Legendre quadrature rule of order 4 is used. Function numbers 1 – 5 correspond to the upper list. The last row of this table is multiplied by 10^{-3}.

As we can see from the table 1, functions with better regularities are approximated better than the others. However, smooth functions are not so nicely approximated by the Haar wavelet. As we can notice for function 5,
functions which have shapes close to the Haar function shape, are approximated the best. We were not able to approximate the function \( f(t) = 1/t \) because of stack limit errors in Matlab. Approximation with wavelets is not possible for this function because it does not belong to \( L^2([0,1]) \).

The pictures 5,6,7 show the approximations of the functions 1 ,4 and 6 by the Haar system.

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1558</td>
<td>0.0571</td>
<td>0.0205</td>
<td>0.0072</td>
<td>0.0025</td>
<td>0.0008</td>
<td>0.0002</td>
<td>0.0001</td>
<td>1.1102e-016</td>
</tr>
<tr>
<td>3</td>
<td>0.0914</td>
<td>0.0327</td>
<td>0.0115</td>
<td>0.0040</td>
<td>0.0014</td>
<td>0.0005</td>
<td>0.0001</td>
<td>3.2369e-005</td>
<td>3.8164e-017</td>
</tr>
<tr>
<td>4</td>
<td>0.0519</td>
<td>0.0201</td>
<td>0.0074</td>
<td>0.0026</td>
<td>0.0009</td>
<td>0.0003</td>
<td>0.0001</td>
<td>2.1558e-005</td>
<td>8.6736e-018</td>
</tr>
<tr>
<td>5</td>
<td>0.3086</td>
<td>0.2237</td>
<td>0.1060</td>
<td>0.0437</td>
<td>0.0164</td>
<td>0.0056</td>
<td>0.0018</td>
<td>0.0004</td>
<td>2.7105e-017</td>
</tr>
</tbody>
</table>

Table 1: Approximation error of the Haar wavelet

Now we want to see if we are approximating integral operators in the right way. For known solutions of the integral equation we get the operator matrix multiplied by wavelet coefficients of the solution and check if we get wavelet coefficients of the right-hand side. We select for our test, three kernels with corresponding solutions and the right-hand sides:
Figure 6: Function \(|t - 1/2|^3\) approximated at scale \(J = 6\) with the Haar wavelet

1. \(k(x, y) = e^{\mid x-y \mid}\), right-hand side \(g(x) = x^2 - x + 1\), solution is \(f(y) = (1 + x - x^2)/2\)

2. \(k(x, y) = \cos(x - y)\), right-hand side \(g(x) = \cos(1 - x) - \cos(x) + \sin(1 - x)\), solution is \(f(y) = y\)

3. \(k(x, y) = (x - y)\), right-hand side \(g(x) = x/3 - 1/4\), solution is \(f(y) = y^2\)

We present calculation time, error norm \(K(f) - g\), the condition number of the operator matrix \(K\) for different kernels in the tables 2, 3, 4.

The figures 9, 8, 10 illustrate how the condition number behaves for different kernels.

To see the errors for the different kernels, let us look at the figure 11. The running times can be viewed in the figure 12.

Distribution of the operator matrix entries can be viewed from in the following figures: 19 and 14 for \(e^{\mid x - y \mid}\) kernel, 20 and 16 for \(\cos(x - y)\) kernel, 21 and 18 for \((x - y)\) kernel.

We can see that the calculation of operator matrix using numerical integration takes a lot of time. We may advice to use discrete values of the function instead of calculation of initial coefficients and calculate the entries.
<table>
<thead>
<tr>
<th>$J$</th>
<th>computation time(s)</th>
<th>error</th>
<th>condition number</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.122863</td>
<td>9.0980e-004</td>
<td>88.9425</td>
</tr>
<tr>
<td>3</td>
<td>0.265241</td>
<td>0.0035</td>
<td>493.6098</td>
</tr>
<tr>
<td>4</td>
<td>0.606568</td>
<td>0.0039</td>
<td>2.0743e+003</td>
</tr>
<tr>
<td>5</td>
<td>1.600803</td>
<td>0.0041</td>
<td>8.8254e+003</td>
</tr>
<tr>
<td>6</td>
<td>4.990421</td>
<td>0.0042</td>
<td>3.5838e+004</td>
</tr>
<tr>
<td>7</td>
<td>18.319657</td>
<td>0.0042</td>
<td>1.4443e+005</td>
</tr>
<tr>
<td>8</td>
<td>70.142678</td>
<td>0.0042</td>
<td>5.8049e+005</td>
</tr>
<tr>
<td>9</td>
<td>276.833808</td>
<td>0.0042</td>
<td>2.3795e+006</td>
</tr>
</tbody>
</table>

Table 2: Running time, error and the condition number of the operator matrix for the first kernel

<table>
<thead>
<tr>
<th>$J$</th>
<th>computation time(s)</th>
<th>error</th>
<th>condition number</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.047573</td>
<td>0.0014</td>
<td>2.9424e+016</td>
</tr>
<tr>
<td>3</td>
<td>0.317088</td>
<td>3.4892e-004</td>
<td>1.1627e+018</td>
</tr>
<tr>
<td>4</td>
<td>0.623430</td>
<td>8.7715e-005</td>
<td>6.8629e+017</td>
</tr>
<tr>
<td>5</td>
<td>1.079848</td>
<td>2.2401e-005</td>
<td>1.6680e+018</td>
</tr>
<tr>
<td>6</td>
<td>4.275434</td>
<td>1.0345e-005</td>
<td>4.3423e+019</td>
</tr>
<tr>
<td>7</td>
<td>17.128518</td>
<td>3.9334e-005</td>
<td>2.2853e+019</td>
</tr>
<tr>
<td>8</td>
<td>68.300118</td>
<td>1.2576e-004</td>
<td>7.6392e+019</td>
</tr>
<tr>
<td>9</td>
<td>277.027298</td>
<td>4.1644e-004</td>
<td>1.2382e+020</td>
</tr>
</tbody>
</table>

Table 3: Running time, error and the condition number of the operator matrix for the second kernel

<table>
<thead>
<tr>
<th>$J$</th>
<th>computation time(s)</th>
<th>error</th>
<th>condition number</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.100598</td>
<td>0.0052</td>
<td>1.3204e+035</td>
</tr>
<tr>
<td>3</td>
<td>0.320891</td>
<td>0.0013</td>
<td>1.7363e+051</td>
</tr>
<tr>
<td>4</td>
<td>0.320891</td>
<td>3.2552e-004</td>
<td>8.0565e+067</td>
</tr>
<tr>
<td>5</td>
<td>1.073480</td>
<td>8.1380e-005</td>
<td>1.6188e+034</td>
</tr>
<tr>
<td>6</td>
<td>4.226835</td>
<td>1.9878e-005</td>
<td>1.1799e+019</td>
</tr>
<tr>
<td>7</td>
<td>16.841857</td>
<td>4.2079e-006</td>
<td>1.5124e+020</td>
</tr>
<tr>
<td>8</td>
<td>68.242899</td>
<td>7.6523e-006</td>
<td>1.9135e+021</td>
</tr>
<tr>
<td>9</td>
<td>270.081683</td>
<td>1.4800e-004</td>
<td>8.8890e+021</td>
</tr>
</tbody>
</table>

Table 4: Running time, error and the condition number of the operator matrix for the third kernel
Figure 7: Function $t^3 - 3/2t + 1$ approximated at scale $J = 6$ with the Haar wavelet

Figure 8: Condition numbers of $e^{x-y}$ kernel for the Haar wavelet at different scales
Figure 9: Condition numbers of $\cos(x - y)$ kernel for the Haar wavelet at different scales

Figure 10: Condition numbers of $x - y$ kernel for the Haar wavelet at different scales
of the operator matrix by two-dimensional Haar wavelet transform. We can notice that matrix entries are becoming smaller while scale is increasing. The
matrices of the integral operator have the same block structure, where each
Figure 15: Plot of operator matrix of $\cos(x - y)$ kernel for the Haar wavelet

Figure 16: Logarithmic plot of absolute values of operator matrix of $\cos(x - y)$ kernel for the Haar wavelet

block corresponds to different scales in $x$ and $y$ directions.
Figure 17: Plot of operator matrix of $x - y$ kernel for Haar the wavelet

Figure 18: Logarithmic plot of absolute values of operator matrix of $x - y$ kernel for the Haar wavelet

### 6.2 Daubechies wavelets

For Daubechies wavelets, we calculate the running time for different dbN wavelets $n = 1, 2, 3$ as well as approximation error. But our goal in this
<table>
<thead>
<tr>
<th>$J$</th>
<th>time(s) db1</th>
<th>time(s)db2</th>
<th>time(s)db3</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.009877</td>
<td>0.007587</td>
<td>0.012677</td>
<td>0.1349</td>
</tr>
<tr>
<td>3</td>
<td>0.015824</td>
<td>0.016181</td>
<td>0.019213</td>
<td>0.0887</td>
</tr>
<tr>
<td>4</td>
<td>0.031828</td>
<td>0.033053</td>
<td>0.039420</td>
<td>0.0614</td>
</tr>
<tr>
<td>5</td>
<td>0.073895</td>
<td>0.076270</td>
<td>0.090139</td>
<td>0.0432</td>
</tr>
<tr>
<td>6</td>
<td>0.187627</td>
<td>0.199861</td>
<td>0.220393</td>
<td>0.0305</td>
</tr>
<tr>
<td>7</td>
<td>0.528603</td>
<td>0.537938</td>
<td>0.603664</td>
<td>0.0216</td>
</tr>
<tr>
<td>8</td>
<td>1.702586</td>
<td>1.678969</td>
<td>1.833927</td>
<td>0.0152</td>
</tr>
<tr>
<td>9</td>
<td>7.387363</td>
<td>7.605183</td>
<td>7.574171</td>
<td>0.0108</td>
</tr>
<tr>
<td>10</td>
<td>41.687030</td>
<td>41.812049</td>
<td>44.192067</td>
<td>0.0076</td>
</tr>
</tbody>
</table>

Table 5: Running time for db1,db2,db3 and error for the first kernel

<table>
<thead>
<tr>
<th>$J$</th>
<th>time(s) db1</th>
<th>time(s)db2</th>
<th>time(s)db3</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.007148</td>
<td>0.007854</td>
<td>0.081657</td>
<td>0.2234</td>
</tr>
<tr>
<td>3</td>
<td>0.014711</td>
<td>0.019451</td>
<td>0.022050</td>
<td>0.1542</td>
</tr>
<tr>
<td>4</td>
<td>0.030687</td>
<td>0.033597</td>
<td>0.041804</td>
<td>0.1078</td>
</tr>
<tr>
<td>5</td>
<td>0.073311</td>
<td>0.073910</td>
<td>0.090232</td>
<td>0.0758</td>
</tr>
<tr>
<td>6</td>
<td>0.190062</td>
<td>0.195917</td>
<td>0.226694</td>
<td>0.0534</td>
</tr>
<tr>
<td>7</td>
<td>0.519222</td>
<td>0.526337</td>
<td>0.596252</td>
<td>0.0377</td>
</tr>
<tr>
<td>8</td>
<td>1.651121</td>
<td>1.707447</td>
<td>1.802958</td>
<td>0.0267</td>
</tr>
<tr>
<td>9</td>
<td>7.266632</td>
<td>7.337076</td>
<td>7.572320</td>
<td>0.0189</td>
</tr>
<tr>
<td>10</td>
<td>42.937364</td>
<td>43.101450</td>
<td>43.645364</td>
<td>0.0133</td>
</tr>
</tbody>
</table>

Table 6: Running time for db1,db2,db3 and error for the second kernel

The case is to determine how condition number of operator matrix will change if we apply periodic Daubechies wavelet transform $dbN$ $n = 1, 2, 3$. We will use the same kernel function as in the previous subsection. First of all, we calculate the running times for the kernel functions and approximation error. See tables 5, 6 and 6. Approximation error for different Daubechies wavelet transforms $dbN$ $n = 1, 2, 3$ stay the same so we will include only one column in the table. We supposed that the approximation errors for different $dbN$ wavelets stay the same because periodic wavelet transform does not influence eigenvalues.

We can see that calculation time from scale 2 to scale 10 grows exponentially. For different types of Daubechies wavelets, we get almost the same running time. We can notice that $dbN$ has less running time than $dbNM$.
Table 7: Running time for db1, db2, db3 and error for the third kernel

\[ N < M. \] The running time grows as \( N \) increasing. And for big \( N \) we have much more calculation time.

How will the condition number propagates for different types of Daubechies wavelets? Graphs of conditions numbers for \( dBN, n = 2, 3, 4 \) and kernels 1, 2, 3 can be seen in figures 19, 20, 21.

![Condition number graph](image)

Figure 19: Condition propagation for \( exp|\frac{x-y}{\sigma}| \) kernel and different dbN, \( N = 2, 3, 5 \)
Figure 20: Condition propagation for $\cos(x - y)$ kernel and different $\text{dbN}, N = 2, 3, 5$

Figure 21: Condition propagation for $(x - y)$ kernel and different $\text{dbN}, N = 2, 3, 5$
From these graphs we can conclude that kernel functions having form $k(x, y) = f(|x-y|)$ and $k(x, y) = |f(x) - f(y)|$ have the same condition number for different $N$. We can notice that condition number is much smaller for some number of $N$. Therefore we can assume that for every function exists number $N$ with minimal condition number of the operator matrix.

Let us analyze how the elements of the operator matrix behave. As we know from [12], periodized wavelets have only one vanishing moment, so the number of nonzero elements in the operator matrix is minimal for db1 (Haar). Table 8 proves our considerations.

<table>
<thead>
<tr>
<th>$dbN$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>6</td>
<td>20</td>
<td>130</td>
<td>628</td>
<td>2760</td>
<td>11376</td>
<td>46220</td>
<td>178364</td>
<td>633162</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>16</td>
<td>61</td>
<td>238</td>
<td>975</td>
<td>3922</td>
<td>15847</td>
<td>63867</td>
<td>257844</td>
<td>1038642</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>14</td>
<td>64</td>
<td>256</td>
<td>1022</td>
<td>4095</td>
<td>16382</td>
<td>65534</td>
<td>262142</td>
<td>1048571</td>
</tr>
</tbody>
</table>

Table 8: Number of nonzero operator matrix elements for $x-y$ kernel function

We can notice that all these operator matrices are almost full. The difference between number of nonzero elements is not significant.

Let us plot the values of operator matrix of different kernels in wavelet basis generated by $db2$: figures 22 and 23 for $e^{\frac{|x-y|}{2}}$ kernel, 24, 25 for $\cos(x - y)$ kernel and 26, 27 for $x - y$ kernel.

Plots of values of the operator matrix of $e^{\frac{|x-y|}{2}}$ kernel for different $dbN$, $n = 3, 4, 5$ can be seen in figures 28, 29, 30, 31, 32, 33.

As we can see with increasing $N$, entries of the operator matrix is decreasing. We can also notice bigger coefficients on the first row and column of the matrix. That corresponds to high amplitude of wavelet coefficients on the boundary. Operator matrices have the same structure as was described in the previous section.
Figure 22: Plot of operator matrix of $e^{|x-y|}$ kernel for db2

Figure 23: Logarithmic plot of absolute values of operator matrix of $e^{|x-y|}$ kernel for db2
Figure 24: Plot of operator matrix of $\cos(x - y)$ kernel for db2

Figure 25: Logarithmic plot of absolute values of operator matrix of $\cos(x - y)$ kernel for db2
Figure 26: Plot of operator matrix of \( x - y \) kernel for db2

Figure 27: Logarithmic plot of absolute values of operator matrix of \( x - y \) kernel for db2
Figure 28: Plot of operator matrix of $e^{|x-y|}$ function for db3

Figure 29: Logarithmic plot of absolute values of operator matrix of $e^{|x-y|}$ function for db3
Figure 30: Plot of operator matrix of $e^{\|x-y\|}$ function for db4

Figure 31: Logarithmic plot of absolute values of operator matrix of $e^{\|x-y\|}$ function for db4
Figure 32: Plot of operator matrix of $e^{|x-y|}$ function for db5

Figure 33: Logarithmic plot of absolute values of operator matrix of $e^{|x-y|}$ function for db5
7 Conclusion

In this work we introduced the concept of wavelets and MRA and presented Haar and Daubechies wavelets. We provided the theory for construction of Daubechies wavelet functions as well as Daubechies scaling functions. We use wavelet transform decomposition and reconstruction methods and describe all the necessary algorithms used. Later, we solve an integral operator with the help of Haar and Daubechies wavelet bases. In case of the Daubechies wavelets, we extend the function periodically outside \([0, 1]\). However there are other wavelet constructions available on the interval - folded and boundary wavelets which have more vanishing moments than periodized wavelets \([12],[1]\). One can use these wavelets for solving an integral equation of the first kind.

In numerical results section, we analyze the condition number of the operator matrix arising from different wavelet representations. However, some questions are still open. One can provide a theory to explain why the condition numbers of the operator matrix with a kernel of the form \(k(x, y) = f(|x - y|)\) and \(k(x, y) = |f(x) - f(y)|\) are equal. It is unclear yet, why do we get the same approximation error for different dbN wavelets. Another question can be to predict \(N\) theoretically for which, the operator matrix will have the smallest condition number among different dbN wavelets.

In this work we were able to solve integral equations only if its operator matrix has low condition number. As integral equations of the first kind are ill-posed problems, some methods are necessary for solving such equations. One can use regularization methods such as Philips-Tikhonov or iterative methods (Landweber iterative methods) and solve an integral equation with an operator matrix having a big condition number. The same can be done using the approximate inverse method.
References


### A Tables of $h_k$ coefficients

<table>
<thead>
<tr>
<th>$N$</th>
<th>$k$</th>
<th>$h_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0.482962913145</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.836516303738</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.224143868042</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>-0.129409522551</td>
</tr>
<tr>
<td>$N=2$</td>
<td>0</td>
<td>0.332670552950</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.806891509311</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.459877502118</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>-0.135011020010</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.085441273882</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.035226291882</td>
</tr>
<tr>
<td>$N=3$</td>
<td>0</td>
<td>0.230377813309</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.714846570553</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.630880767930</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>-0.027983769417</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>-0.187034811719</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.030841381836</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.032883011667</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>-0.010597401785</td>
</tr>
<tr>
<td>$N=4$</td>
<td>0</td>
<td>0.160102397974</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.603829269797</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.724308528438</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.138428145901</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>-0.242294887066</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>-0.032244869585</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>-0.006241490213</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.077571493840</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>-0.012580751999</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>0.003335725285</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$N$</th>
<th>$k$</th>
<th>$h_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0.111540743350</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.494623890398</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.751133908021</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.315250351709</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>-0.12976867567</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.09750160558</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.02752865530</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.000553842201</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>0.00477257511</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>-0.22626493965</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>-0.031582039317</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>-0.001077301085</td>
</tr>
<tr>
<td>$N=6$</td>
<td>0</td>
<td>0.077852054085</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.39659319482</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.729132090846</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.46972287405</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.143906003929</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>-0.224036184994</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.071309219267</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.080612609151</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>-0.038029936935</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>0.000165745163</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0.01255098556</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>0.00042577973</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>-0.001801640704</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>0.000353713800</td>
</tr>
<tr>
<td>N</td>
<td>k</td>
<td>$h_k$</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0.054415842243</td>
</tr>
<tr>
<td>1</td>
<td>0.284015542962</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.5854683654</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.128747426620</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>-0.017369301002</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-0.000472484574</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.000472484574</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.013981027917</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.0007544940</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.000117476784</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>N=8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>N=9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
<tr>
<td>17</td>
</tr>
<tr>
<td>18</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>N=10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
<tr>
<td>17</td>
</tr>
</tbody>
</table>
## B Table of $C^n_k$ coefficients

<table>
<thead>
<tr>
<th>n</th>
<th>$C^n_0$</th>
<th>$C^n_1$</th>
<th>$C^n_2$</th>
<th>$C^n_3$</th>
<th>$C^n_4$</th>
<th>$C^n_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\frac{1}{2}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>$\frac{1}{6}$</td>
<td>$\frac{4}{6}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>$\frac{1}{8}$</td>
<td>$\frac{3}{8}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>$\frac{7}{90}$</td>
<td>$\frac{32}{90}$</td>
<td>$\frac{12}{90}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>$\frac{19}{288}$</td>
<td>$\frac{75}{288}$</td>
<td>$\frac{50}{288}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>$\frac{41}{840}$</td>
<td>$\frac{216}{840}$</td>
<td>$\frac{27}{840}$</td>
<td>$\frac{272}{840}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>$\frac{751}{17280}$</td>
<td>$\frac{3577}{17280}$</td>
<td>$\frac{928}{17280}$</td>
<td>$\frac{10496}{17280}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>$\frac{989}{28350}$</td>
<td>$\frac{5888}{28350}$</td>
<td>$\frac{1323}{28350}$</td>
<td>$\frac{2989}{28350}$</td>
<td>$\frac{4540}{28350}$</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>$\frac{2857}{89600}$</td>
<td>$\frac{315541}{89600}$</td>
<td>$\frac{1080}{89600}$</td>
<td>$\frac{19344}{89600}$</td>
<td>$\frac{5778}{89600}$</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>$\frac{16067}{87091200}$</td>
<td>$\frac{106300}{87091200}$</td>
<td>$\frac{48525}{87091200}$</td>
<td>$\frac{272400}{87091200}$</td>
<td>$\frac{260550}{87091200}$</td>
<td>$\frac{427638}{87091200}$</td>
</tr>
</tbody>
</table>

Table 9: Catesian numbers for $k \leq n/2$

## C Gauss-Legandre coefficients $w_i$ and $x_i$
<table>
<thead>
<tr>
<th>n</th>
<th>nodes ±x_k</th>
<th>Weights w_k</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.577350269189626</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0.888888888888889</td>
</tr>
<tr>
<td></td>
<td>0.774596669241483</td>
<td>0.555555555555556</td>
</tr>
<tr>
<td>4</td>
<td>0.339981043584856</td>
<td>0.652145154862546</td>
</tr>
<tr>
<td></td>
<td>0.861136311594053</td>
<td>0.347854845137454</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0.568888888888889</td>
</tr>
<tr>
<td></td>
<td>0.538469310105683</td>
<td>0.478628670499366</td>
</tr>
<tr>
<td></td>
<td>0.906179845938664</td>
<td>0.236926885056189</td>
</tr>
<tr>
<td>6</td>
<td>0.238619186083197</td>
<td>0.467913934572691</td>
</tr>
<tr>
<td></td>
<td>0.661209386466265</td>
<td>0.360761573048139</td>
</tr>
<tr>
<td></td>
<td>0.932469514203152</td>
<td>0.932469514203152</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0.417959183673469</td>
</tr>
<tr>
<td></td>
<td>0.404845151377397</td>
<td>0.381830050505119</td>
</tr>
<tr>
<td></td>
<td>0.741531185500394</td>
<td>0.279705391489277</td>
</tr>
<tr>
<td></td>
<td>0.949107912342759</td>
<td>0.129484966168870</td>
</tr>
<tr>
<td>8</td>
<td>0.183434642495650</td>
<td>0.36268378378362</td>
</tr>
<tr>
<td></td>
<td>0.525532409916329</td>
<td>0.313706645877887</td>
</tr>
<tr>
<td></td>
<td>0.79666477413627</td>
<td>0.222381034453374</td>
</tr>
<tr>
<td></td>
<td>0.960289856497536</td>
<td>0.101228536290376</td>
</tr>
</tbody>
</table>

Table 10: Gauss-Legendre Quadrature.
<table>
<thead>
<tr>
<th>$n$</th>
<th>nodes ±$x_k$</th>
<th>Weights $w_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.330239355001260</td>
<td>0.312347077040003</td>
</tr>
<tr>
<td>9</td>
<td>0.613371432700590</td>
<td>0.260610696402935</td>
</tr>
<tr>
<td></td>
<td>0.836031107326636</td>
<td>0.180648160694857</td>
</tr>
<tr>
<td></td>
<td>0.968160239507626</td>
<td>0.081274388361574</td>
</tr>
<tr>
<td>10</td>
<td>0.148874338981631</td>
<td>0.295524224714753</td>
</tr>
<tr>
<td></td>
<td>0.433895394129247</td>
<td>0.269266719309996</td>
</tr>
<tr>
<td></td>
<td>0.679409568299024</td>
<td>0.219086362515982</td>
</tr>
<tr>
<td></td>
<td>0.865063366688985</td>
<td>0.149451349150581</td>
</tr>
<tr>
<td></td>
<td>0.973906528517172</td>
<td>0.066671344308688</td>
</tr>
<tr>
<td>16</td>
<td>0.095012509837637</td>
<td>0.189450610455068</td>
</tr>
<tr>
<td></td>
<td>0.281603550779259</td>
<td>0.182603415044924</td>
</tr>
<tr>
<td></td>
<td>0.458016777657227</td>
<td>0.169156519395003</td>
</tr>
<tr>
<td></td>
<td>0.617876244402644</td>
<td>0.149595988816577</td>
</tr>
<tr>
<td></td>
<td>0.755404408355003</td>
<td>0.124628971255534</td>
</tr>
<tr>
<td></td>
<td>0.865631202387832</td>
<td>0.095158511682493</td>
</tr>
<tr>
<td></td>
<td>0.944575023073233</td>
<td>0.062253523938648</td>
</tr>
</tbody>
</table>

Table 11: Gauss-Legendre Quadrature.
Eidesstattliche Erklärung

Ich, Mikalai Zhudro, erkläre an Eides statt, dass ich die vorliegende Masterarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Author: ________________
Mikalai Zhudro