Numerical Solution of Partial Differential Equations Using Wavelet Approximation Space

(MATH 667 project - May 16, 2003)

Sami Hamid, Tzanio Kolev, Quoc Thong Le Gia, Wuxiang Wu

1 Introduction

Let Ω be a bounded domain in $\mathbb{R}^2$ with boundary Γ. Our objective in this paper is to solve differential equations of the form

$$-\nabla \cdot (a\nabla u) + \alpha\, u = f \quad \text{in } \Omega, \tag{1}$$

where $u : \mathbb{R}^2 \mapsto \mathbb{R}$ is the unknown solution and $f : \mathbb{R}^2 \mapsto \mathbb{R}$ is the right-hand side data. The functions $a : \mathbb{R}^2 \mapsto \mathbb{R}$ and $\alpha : \mathbb{R}^2 \mapsto \mathbb{R}$ have some physical interpretation in the different applications of (1). Boundary conditions should be imposed, which in general are:

$$u = g \quad \text{on } \Gamma_D\,, \quad (a\nabla u) \cdot n = \nu \quad \text{on } \Gamma_N, \tag{2}$$

where n stands for the unit outward normal at each point of Γ and Γ_D, Γ_N are two open subsets of Γ, such that

$$\Gamma_D \cap \Gamma_N = \emptyset \quad \text{and} \quad \Gamma = \overline{\Gamma_D} \cup \overline{\Gamma_N}. \tag{3}$$

The functions $g : \Gamma_D \mapsto \mathbb{R}$ and $\nu : \Gamma_N \mapsto \mathbb{R}$ represent Dirichlet and Neumann boundary data respectively.
Common techniques used for numerical solutions of physical problems like this fall into three main classes:

- *Finite difference methods*
  The different unknowns are defined by their values on discrete (finite) grid and differential operators are replaced by difference operators using neighboring points.

- *Finite element methods*
  The unknown solution is approximated by a linear combination of a set of linearly independent test functions, which are piecewise continuous and non-vanishing only on finite number of "elements" in the domain.

- *Spectral methods*
  Utilize basis functions which are infinitely differentiable and non-vanishing on the entire domain.

Wavelet applications to the solution of partial differential equations like (1)-(2) have been very limited. This is mostly due to the fact that wavelet-based numerical algorithms are still in their infancy. Nevertheless, some serious practical applications are already available (see e.g. [8]). The basic idea behind wavelet decomposition is to represent a function in terms

of building blocks called wavelets. They are localized in both location and scale, so one can expect that numerical methods based on wavelets attain both good spatial and spectral resolution. Therefore, a very promising idea is to combine the compressing approximation properties of wavelet space with the finite element approach.
One way to use the wavelet decomposition for numerical solution of partial differential equations, commonly known as the Wavelet Galerkin (WG) method, is to use a Galerkin projection in order to derive a system of algebraic equations for wavelet coefficients. The main problem in this method is that it is not very effective in the treatment of general boundary conditions. A variety of techniques have been devised in order to overcome these difficulties. One of them, called fictitious domain approach, is presented in the sequence of papers [4, 5, 6, 11, 12]. Other approaches based on construction of special wavelet spaces can be found in [7, 9, 13].
In §2 we present the framework for numerical solutions of equations of the form (1)-(2) from finite element point of view. In §3 we outline the basics of the multiresolution analysis and wavelet theory. The connection between these two subjects is made in §4 which contains the main idea of this paper. This is followed by the numerical experiments given in §5. Finally some of the Matlab and C code that has been implemented is collected in §6.
The following notation will be used: for any domain $\Omega \subset \mathbb{R}^d$, $d \in \mathbb{N}$ we denote $L^2(\Omega)$ to be the set of square integrable functions in Ω in Lebesgue sense. This is a Hilbert space with respect to the inner product

$$\langle f, g \rangle \equiv \langle f, g \rangle_{L^2(\Omega)} = \int_\Omega fg.$$

$L^\infty(\Omega)$ is defined by

$$L^\infty(\Omega) = \{f \ : \ \operatorname{ess\,sup}_{x\in\Omega} |f(x)| < \infty\}.$$

$H^k(\Omega)$, $k = 0, 1, \ldots$ will stand for the Sobolev space $W_2^k(\Omega)$ defined as:

$$H^k(\Omega) \equiv W_2^k(\Omega) = \{f \in L^2(\Omega) \ : \ \partial^\alpha f \in L^2(\Omega) \quad \forall \text{ multiindex } \alpha, \, |\alpha| \le k\},$$

where all the derivatives are taken in a weak sense. In particular $H^0 \equiv L^2$.
The norm and semi-norm in $H^k(\Omega)$ are given by:

$$\|u\|_k \equiv \|u\|_{H^k(\Omega)} = \left(\sum_{|\alpha|\le k} \int_\Omega |\partial^\alpha u|^2 \right)^{1/2}, \quad |u|_k = \left(\sum_{|\alpha|=k} \int_\Omega |\partial^\alpha u|^2 \right)^{1/2}.$$

2 Numerical solution of partial differential equations

In this section we outline the theory behind the methods for numerical solution of elliptic partial differential equations. For more details on this subject see [2, 3].

2.1 Weak formulation

Often it is not possible to find a classical solution of the system (1)-(2). In such a case we consider the so-called weak solution, which is the solution of the following reformulation of the original problem:

$$\text{Find } u \in \mathrm{V}_g : \quad \mathcal{A}(u,v) = \mathcal{F}(v) \quad \forall v \in \mathrm{V}, \tag{4}$$

$$\mathcal{A}(u,v) = \int_\Omega a\nabla u \nabla v + \alpha\, uv\,, \quad \mathcal{F}(v) = \int_\Omega fv + \int_{\Gamma_N} \nu v \tag{5}$$

$$\mathrm{V} = \{v \in H^1(\Omega) : v = 0 \text{ on } \Gamma_D\}\,, \quad \mathrm{V}_g = \{v \in H^1(\Omega) : v = g \text{ on } \Gamma_D\}. \tag{6}$$

In this paper we will assume that

$$\alpha(x) \in L^\infty(\Omega),\ \nu(x) \in L^2(\Gamma_N), \tag{7}$$

$$a(x) \in L^\infty(\Omega),\ 0 < a_0 \le a(x) \le a_1, \tag{8}$$

$$\exists u_g \in H^1(\Omega)\ :\ u_g = g \text{ on } \Gamma_D, \tag{9}$$

and will focus on the following special cases of the weak formulation:

DP Pure Dirichlet Boundary Conditions
$\Gamma \equiv \Gamma_D$, $g \equiv 0$,$\mathrm{V} = \mathrm{V}_g = H_0^1(\Omega)$, $\alpha \equiv 0$ in Ω.

NP Pure Neumann Boundary Conditions
$\Gamma \equiv \Gamma_N$, $\mathrm{V} = \mathrm{V}_g = H^1(\Omega)$, $0 < \alpha_0 \le \alpha(x)$ for any $x \in \Omega$.

Notice that in both cases $\mathrm{V} \equiv \mathrm{V}_g$.

Lemma 1 *Assume (7)-(9). If $u \in C^2(\overline{\Omega})$ satisfies (1)-(2) then it satisfies (4). Moreover if u is the solution of (4) and it is sufficiently smooth then it also satisfies (1)-(2).*

Proof: Let $u \in C^2(\overline{\Omega})$ satisfies (1)-(2). Clearly $u \in \mathrm{V}_g$. Multiply (1) by any $v \in \mathrm{V}$, integrate over Ω and apply the divergence theorem:

$$\int_\Omega a\nabla u \nabla v - \int_\Gamma (a\nabla u)\cdot n\, v + \int_\Omega \alpha\, u\, v = \int_\Omega fv.$$

Since $v = 0$ on Γ_D and $(a\nabla u)\cdot n = \nu$ on Γ_N, we have (4).
Now, assume (4) and let $v \in \mathrm{V}$. Because of the smoothness we can apply the divergence theorem again:

$$\int_\Omega \left[-\nabla\cdot(a\nabla u)\, v + \alpha\, uv\right] + \int_{\Gamma_N} (a\nabla u)\cdot n\, v = \int_\Omega fv + \int_{\Gamma_N} \nu\, v \quad \forall v \in \mathrm{V}. \tag{10}$$

Let $v \in C_0^\infty(\Omega) \subset \mathrm{V}$. The boundary integrals will vanish since v has compact support. Thus

$$\langle -\nabla\cdot(a\nabla u) + \alpha\, u, v\rangle_{L^2(\Omega)} = \langle f, v\rangle_{L^2(\Omega)} \quad \forall v \in C_0^\infty(\Omega),$$

which implies (1). Plugging this back in (10) gives

$$\langle (a\nabla u)\cdot n, v\rangle_{L^2(\Gamma_N)} = \langle \nu, v\rangle_{L^2(\Gamma_N)} \quad \forall v \in \mathrm{V}$$

which together with $u = g$ on Γ_D (since $u \in \mathrm{V}_g$) implies (2). ∎

Remark 1 *If the boundary of Ω is smooth enough one can prove that actually the solution of (1)-(2) is smooth, so the weak and the classical formulations are equivalent (see [2]).*

Theorem 1 *Suppose that $\mathcal{A}(\cdot,\cdot)$ is a symmetric bilinear form and $\mathcal{F}(\cdot)$ is a linear functional in the Hilbert space $\mathcal{H}$. If*

- *$\mathcal{A}(\cdot,\cdot)$ is continuous i.e. $\exists\, A_1 :\ |\mathcal{A}(u,v)| \leq A_1 \|u\|_{\mathcal{H}} \|v\|_{\mathcal{H}}$*
- *$\mathcal{A}(\cdot,\cdot)$ is coercive i.e. $\exists\, A_0 > 0 :\ A_0 \|u\|^2_{\mathcal{H}} \leq \mathcal{A}(u,u)$*
- *$\mathcal{F}(\cdot)$ is continuous i.e. $\exists\, F_1 :\ |\mathcal{F}(v)| \leq F_1 \|v\|_{\mathcal{H}}$*

then the following problem:

$$\text{Find } u \in \mathcal{H} :\quad \mathcal{A}(u,v) = \mathcal{F}(v) \quad \forall v \in \mathcal{H} \tag{11}$$

has unique solution.

Proof: First note that because of the coercivity, $\mathcal{A}(\cdot,\cdot)$ defines an inner product, so

$$\|u\|_{\mathcal{A}} = \sqrt{\mathcal{A}(u,u)} \tag{12}$$

is a new norm in $\mathcal{H}$ equivalent to $\|\cdot\|_{\mathcal{H}}$. Now the existence and uniqueness of u follow from the Reisz representation theorem for bounded linear functionals in Hilbert space. ∎

Remark 2 *A more general result, known as Lax-Milgram theorem, where $\mathcal{A}(\cdot,\cdot)$ does not need to be symmetric can be proved (see e.g. [2]). Note however that in this case $\mathcal{A}(\cdot,\cdot)$ is no longer an inner product in $\mathcal{H}$.*

Corollary 1 *Assume (7)-(9) then the problems* **DP** *and* **NP** *satisfy the conditions of Theorem 1, and therefore admit unique solution.*

Proof: In both cases $\|\cdot\|_{\mathcal{H}} = \|\cdot\|_{H^1(\Omega)}$. The only nontrivial part is the coercivity for **DP**, which follows from the classical Poincare inequality:

$$\exists C :\ \|u\|_{H^1(\Omega)} \leq C\, |u|_{H^1(\Omega)} \quad \forall u \in H^1_0(\Omega).$$

∎

2.2 Galerkin approximation

In practice we use computers to obtain numerical approximations to the solution of (4). Therefore we should replace this problem with one of finite dimension. Probably the most natural way to do this is to take some finite dimensional subspace $\mathrm{V}_h \subset \mathrm{V}$ and replace the solution u by its approximation in V_h. One method to obtain such an approximation is to use the fact that $\mathcal{A}(\cdot,\cdot)$ defines an inner product in V, and use u_h - the orthogonal projection of u on V_h with respect to this inner product. This gives rise to the so-called Galerkin approximation:

$$\text{Find } u_h \in \mathrm{V}_h :\quad \mathcal{A}(u_h, v_h) = \mathcal{F}(v_h) \quad \forall v_h \in \mathrm{V}_h. \tag{13}$$

The quality of this approximation depends on how "rich" V_h is i.e. on the approximation properties of this space. Note that V_h is finite dimensional, therefore it is closed, which implies that it is a Hilbert space. Thus, (13) will have unique solution by Lax-Milgram.

This also can be seen directly. Suppose that $\dim(\mathrm{V}_h) = n$ and $\{\phi_k\}_{k=1}^n$ is a basis for V_h. Define

$$u = \sum_{k=1}^{n} \mathrm{x}_k \, \phi_k \,, \quad \mathrm{A}_{ij} = \mathcal{A}(\phi_j, \phi_i) \,, \quad \mathrm{b}_i = \mathcal{F}(\phi_i), \tag{14}$$

then (13) is equivalent to the following linear system

$$\mathrm{A}\,\mathrm{x} = \mathrm{b}. \tag{15}$$

The matrix of this system (often called stiffness matrix) is symmetric and positive definite (SPD), so (15) admits unique solution. In practice an exact solution of this system is difficult to be found because of the roundoff errors. Unstability of the system is measured by the condition number of A which in the case of SPD matrix is given by

$$\kappa(A) = \frac{\lambda_n}{\lambda_1} \,, \quad \sigma(\mathrm{A}) = \{\lambda_1 \leq \cdots \leq \lambda_n\}. \tag{16}$$

To improve the condition number we can introduce a preconditioner - SPD matrix $\mathcal{C}$ which approximates (in some sense) A, but can be easily inverted. Then one uses an iterative method such as preconditioned conjugate gradient (PCG) to solve the system $\mathcal{C}^{-1}\mathrm{A}\,\mathrm{x} = \mathcal{C}^{-1}\mathrm{b}$. Here we will give just an outline of this algorithm, and refer to [10] for more details.

Algorithm (PCG)
(solves $\mathbf{Ax} = \mathbf{b}$ with relative error ε and preconditioner $\mathcal{C}$)

initial approximation	$\mathbf{x} = \mathbf{x}_0 = \mathcal{C}^{-1} b$
initial residual	$\mathbf{r} = \mathbf{r}_0 = b - \mathrm{A}\,\mathbf{x}$
initial quasiresidual	$\tilde{\mathbf{r}} = \mathcal{C}^{-1}\mathbf{r}_0$
initial search direction	$\mathbf{d} = \mathbf{d}_0 = \tilde{\mathbf{r}}$
iterate	$\curvearrowright$
compute	$\alpha = (\tilde{\mathbf{r}}^t \cdot \mathbf{r})/(\mathbf{d}^t \cdot \mathrm{A} \cdot \mathbf{d})$
next approximation	$\mathbf{x} = \mathbf{x} + \alpha\mathbf{d}$
next residual	$\mathbf{r} = \mathbf{r} - \alpha\mathrm{A}\,\mathbf{d}$
next quasiresidual	$\tilde{\mathbf{r}} = \mathcal{C}^{-1}\mathbf{r}$
stop if	$\tilde{\mathbf{r}}^t \cdot \mathbf{r} \leq \varepsilon^2\, \tilde{\mathbf{r}}_0^t \cdot \mathbf{r}_0$
compute	$\beta = (\tilde{\mathbf{r}}^t \cdot \mathbf{r})/\left((\tilde{\mathbf{r}}^t \cdot \mathbf{r})_{\mathrm{prev}}\right)$
next search direction	$\mathbf{d} = \tilde{\mathbf{r}} + \beta\mathbf{d}$
next iteration	$\circlearrowleft$

Remark 3 *The above algorithm will converge after no more than*

$$\frac{1}{2}\kappa(\mathcal{C}^{-1}A)\log\frac{2}{\epsilon}$$

iterations.

2.3 Finite element method

It is clear that one of the crucial steps in our approach is the choice of the approximation space V_h. Naturally we want to approximate V (which is $H_0^1(\Omega)$ or $H^1(\Omega)$) with a family of finite dimensional subspaces $\{\mathrm{V}_h\}_{h\in\mathbb{R}^+}$ such that

$$\lim_{h\to 0^+} \mathrm{V}_h = \mathrm{V}.$$

One classical example of such sequence is given by the finite element method (FEM). In this method we suppose that Ω can be represented as a union of some elements τ_h e.g. triangles of size h:

$$\overline{\Omega} = \cup\{\overline{\tau_i} \,:\, \tau_i \in \mathcal{T}_h\}. \tag{17}$$

V_h is defined to be the space of piecewise polynomials of order k in each triangle:

$$\mathrm{V}_h = \{v \in C(\Omega) \,:\, v|_{\tau_i} \in \mathcal{P}_k \quad \forall \tau_i \in \mathcal{T}_h\} \tag{18}$$

The basis functions in V_h for $k = 1$ look like the one in Figure 1. The approximation power

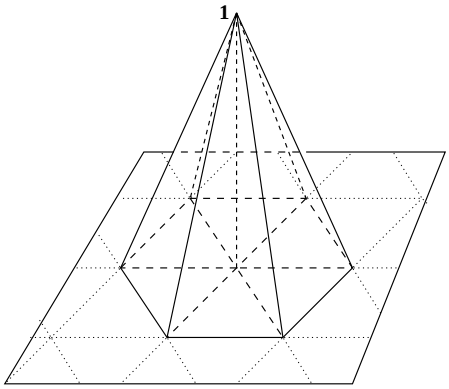

Figure 1: Basis functions for piecewise linear finite element in 2D.

of the FEM space V_h is given by the following theorem (see [2]).

Theorem 2 (Approximation) *If the angles of the triangles in the triangulation $\mathcal{T}_h$ stay always larger than a fixed constant, then given $u \in H^{k+1}(\Omega)$ there exists $v_{u,h} \in V_h$ such that*

$$\|u - v_{u,h}\|_1 \leq Ch^k \|u\|_{k+1}, \tag{19}$$

$$\|u - v_{u,h}\|_0 \leq Ch^{k+1} \|u\|_{k+1}. \tag{20}$$

Remark 4 *One choice for $v_{u,h}$ is the nodal interpolant of u:*

$$\Pi_h u\,(x) = \sum_{i=1}^{N} u(d_i)\phi_i(x)\,, \quad x \in \Omega\,,$$

where $\{d_i\}_{i=1}^N$ are the degrees of freedom of $\mathcal{T}_h$ with corresponding basis functions $\{\phi_i\}_{i=1}^N$.

Having the approximation power and using that the Galerkin approximation is just an orthogonal projection with respect to $\mathcal{A}(\cdot,\cdot)$, we can derive an error estimate.

Theorem 3 (Error estimate) *Under the above assumptions, the error in the FEM is*

$$\|u - u_h\|_1 \leq Ch^k \|u\|_{k+1}. \tag{21}$$

Proof: By coercivity

$$\|u - u_h\|_1^2 \leq A_0^{-1} \mathcal{A}(u - u_h, u - u_h).$$

Since u_h is the $\mathcal{A}(\cdot,\cdot)$ orthogonal projection of u to V_h, we have $\mathcal{A}(u - u_h, v_h) = 0$ for all $v_h \in \mathrm{V}_h$. Thus,

$$\mathcal{A}(u - u_h, u - u_h) = \mathcal{A}(u - u_h, u - v_h) \leq \|u - u_h\|_{\mathcal{A}} \|u - v_h\|_{\mathcal{A}} \leq A_1 \|u - u_h\|_1 \|u - v_h\|_1$$

by Schwarz inequality and continuity. Choosing v_h to be $v_{u,h}$ from Theorem 2 we get

$$\|u - u_h\|_1 \leq A_0^{-1} A_1 \|u - v_{u,h}\|_1 \leq Ch^k \|u\|_{k+1}.$$

■

Remark 5 *Using a standard technique called Nitsche's trick we can derive similar estimate in $L^2(\Omega)$:*

$$\|u - u_h\|_0 \leq Ch^{k+1} \|u\|_{k+1}. \tag{22}$$

3 Multiresolution analysis and Daubechies wavelets

3.1 Multiresolution analysis of $\mathbb{R}$

In this section we outline some of the key results concerning wavelets theory that we are going to use later. For proofs and details see [1].

Definition 1 (Multiresolution Analysis) *A sequence of spaces $\{\mathcal{V}_j\}_{j\in\mathbb{Z}}$ is called Multiresolution Analysis (MRA) with scaling function ϕ if:*

1. $\mathcal{V}_j \subseteq \mathcal{V}_{j+1} \subseteq L^2(\mathbb{R})$
2. $\overline{\bigcup_{j\in\mathbb{Z}} \mathcal{V}_j} = L^2(\mathbb{R})$
3. $\bigcap_{j\in\mathbb{Z}} \mathcal{V}_j = \{0\}$
4. $f(\cdot) \in \mathcal{V}_j \iff f(2^{-j}\cdot) \in \mathcal{V}_0$
5. $\{\phi(\cdot - k)\}_{k\in\mathbb{Z}}$ *is an orthonormal basis for* $\mathcal{V}_0$.

Remark 6 *Condition 2 implies that any $f \in L^2(\mathbb{R})$ can be approximated with arbitrary precision with elements of $\mathcal{V}_j$.*

Remark 7 *In general in condition 5, we need only that $\{\phi(\cdot - k)\}_{k\in\mathbb{Z}}$ forms a frame for $\mathcal{V}_0$. Given this, an orthonormal scaling function can be constructed.*

Corollary 2 *Let $\{\mathcal{V}_j\}_{j\in\mathbb{Z}}$ be MRA with scaling function ϕ, and for any $j \in \mathbb{Z}$, $k \in \mathbb{Z}$ define*

$$\phi_{jk}(x) = 2^{j/2} \phi(2^j x - k). \tag{23}$$

Then $\{\phi_{jk}(x)\}_{k\in\mathbb{Z}}$ is an orthonormal basis for $\mathcal{V}_j$.

Corollary 3 *Let $\{\mathcal{V}_j\}_{j\in\mathbb{Z}}$ be MRA with scaling function ϕ. There exist coefficients $\{p_k\}_{k\in\mathbb{Z}}$ such that*

$$\phi(x) = \sum_{k\in\mathbb{Z}} p_k\, \phi(2x-k). \tag{24}$$

Moreover

$$\sum_{k\in\mathbb{Z}} p_{k-2l}\, \overline{p_k} = 2\delta_{l0} \quad \textit{and} \quad \sum_{k\in\mathbb{Z}} p_{2k} = \sum_{k\in\mathbb{Z}} p_{2k+1} = 1. \tag{25}$$

Theorem 4 *Let $\mathcal{V}_{j+1} = \mathcal{V}_j \bigoplus \mathcal{W}_j$ and define*

$$\psi(x) = \sum_{k\in\mathbb{Z}} (-1)^k\, \overline{p_{1-k}}\, \phi(2x-k). \tag{26}$$

Then $\{\psi_{jk}(x) = 2^{j/2}\psi(2^j x - k)\}_{k\in\mathbb{Z}}$ is an orthonormal basis for $\mathcal{W}_j$. Consequently

$$L^2(\mathbb{R}) = \bigoplus_{j\in\mathbb{Z}} \mathcal{W}_j \tag{27}$$

i.e. $\{2^{j/2}\psi(2^j\cdot - k)\}_{j\in\mathbb{Z},\, k\in\mathbb{Z}}$ is an orthonormal basis for $L^2(\mathbb{R})$.

Theorem 5 *Let ϕ be continuous function with compact support that satisfies:*

- *$\{\phi(\cdot - k)\}_{k\in\mathbb{Z}}$ is an orthonormal system*
- *$\int_{\mathbb{R}} \phi(x)\, dx = 1$*
- *Only finite number of the coefficients p_k in (24) are nonzero.*

Then ϕ is a scaling function (i.e. ϕ can be used in construction of MRA).

Theorem 6 *Suppose that the polynomial*

$$P(z) = \frac{1}{2} \sum_{k\in\mathbb{Z}} p_k\, z^k \tag{28}$$

satisfies

- *$P(1) = 1$*
- *$|P(z)| \neq 0$ for any z with $|z| = 1$*
- *$|P(z)|^2 + |P(-z)|^2 = 1$ for any z with $|z| = 1$.*

Then the iteration

$$\phi_0 = \chi_{[0,1]}, \quad \phi_n(x) = \sum_{k\in\mathbb{Z}} p_k\, \phi_{n-1}(2x-k) \tag{29}$$

converges pointwise and in $L^2(\mathbb{R})$ to a scaling function ϕ.

Remark 8 *If only $p_0, \cdots, p_N$ are nonzero, then $supp(\phi) \subseteq [0, N]$.*

Example 1 (Haar)

$$p_0 = p_1 = 1 \tag{30}$$

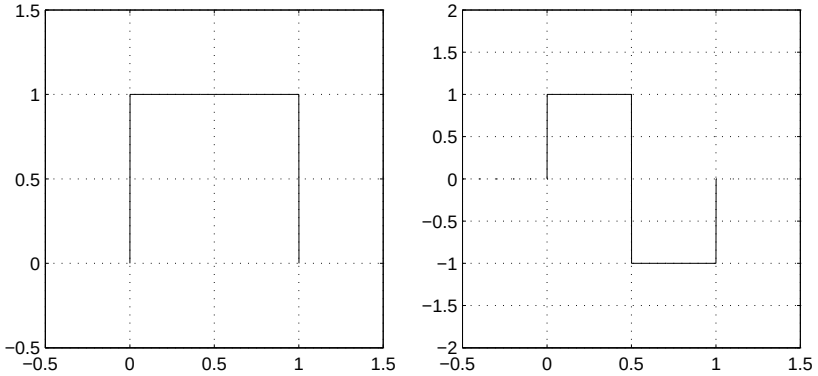

Figure 2: Haar scaling function ϕ (left) and wavelet ψ (right).

Example 2 (Daubechies 2)

$$p_0 = \frac{1+\sqrt{3}}{4},\ p_1 = \frac{3+\sqrt{3}}{4},\ p_2 = \frac{3-\sqrt{3}}{4},\ p_3 = \frac{1-\sqrt{3}}{4}. \tag{31}$$

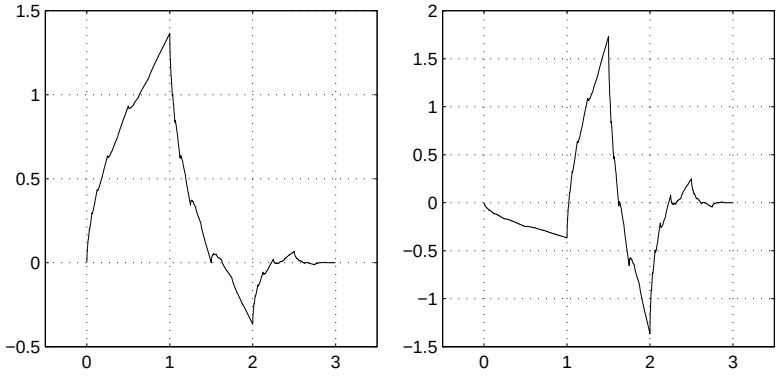

Figure 3: Daubechies 2 scaling function ϕ (left) and wavelet ψ (right).

Theorem 7 *For any $N \in \mathbb{N}$ there exists a Daubechies MRA with functions ϕ and ψ that have compact support of length $2N-1$. Moreover the Daubechies wavelet has N vanishing moments i.e.*

$$\langle x^k, \psi \rangle_{L^2(\mathbb{R})} = 0 \quad , k = \overline{0, N-1}. \tag{32}$$

Remark 9 *In practice if we know the values of ϕ in set of points Z then the scaling relation (29) produces the values of ϕ in $Z \cup \{p/2 : p \in Z\}$. Because of the continuity, it will be sufficient to know the values of ϕ only in the set $\{0, 1, \cdots, 2N-1\}$ to construct it. In order to get these values we can iterate the scaling relation, or we can use the fact that they form the eigenvector corresponding to eigenvalue 1 for the matrix $A := (a_{ij} = p_{2i-j+1})_{i,j=1}^{2N-1}$ (see Function 1 in §6).*

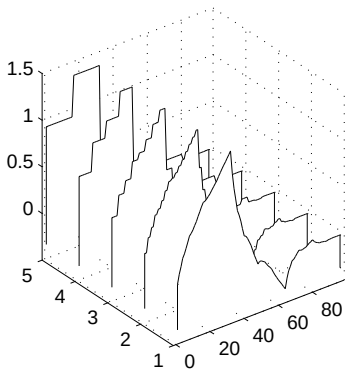

Figure 4: Daubechies 2 scaling function iterations.

Definition 2 (Multiresolution Analysis in 2D) *Let $\{\mathcal{V}_j\}_{j\in\mathbb{Z}}$ be an MRA with scaling function ϕ. Then an MRA for $L^2(\mathbb{R}^2)$ can be defined using the spaces*

$$\mathbb{V}_j = \mathcal{V}_j \otimes \mathcal{V}_j \quad \textit{and} \quad \mathbb{W}_j = (\mathcal{V}_j \otimes \mathcal{W}_j) \oplus (\mathcal{W}_j \times \mathcal{W}_j) \oplus (\mathcal{W}_j \otimes \mathcal{V}_j), \tag{33}$$

where

$$F \otimes G := \{f(x)\, g(y) : f \in F, g \in G\}.$$

An example of basis function in $\mathbb{V}_j$ is given in Figure 5 (see also Function 2 in §6).

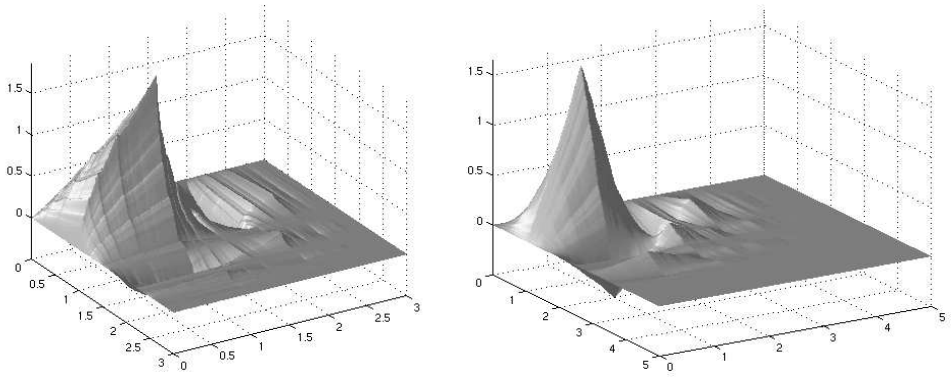

Figure 5: Basis functions in $\mathbb{V}_j$ corresponding to $\phi\otimes\phi$ using Daubechies MRA for $N = 2, 3$.

3.2 Approximation properties of $\mathbb{V}_j$

Let ϕ be the Daubechies scaling function of order N as discussed in §3.1. The scaling function ϕ has compact support $[0, 2N-1]$ and

$$\int_{\mathbb{R}} x\phi(x)dx = \mathcal{F}[x\phi(x)](0) = i\frac{d}{d\omega}\mathcal{F}[\phi(x)](0) = \frac{1}{2}\sum_{k=0}^{2N-1} kp_k.$$

Let us denote by c the first moment of the function ϕ, i.e.

$$c := \frac{1}{2}\sum_{k=0}^{2N-1} kp_k. \tag{34}$$

Theorem 8 *Assume that $f \in C^2(\overline{\Omega})$, let*

$$f^j(x,y) = 2^{-j} \sum_{(p,q)\in\Lambda_j} f\left(\frac{p+c}{2^j}, \frac{q+c}{2^j}\right) \phi_{jp}(x)\phi_{jq}(y), \quad (x,y) \in \Omega, \tag{35}$$

where the index set

$$\Lambda_j = \{(p,q) \in \mathbb{Z}^2 : supp(\phi_{jp}(x)\phi_{jq}(y)) \cap \Omega \neq \emptyset\}. \tag{36}$$

Then

$$\|f - f^j\|_1 \leq C\, 2^{-j},$$

and

$$\|f - f^j\|_0 \leq C\, 2^{-2j},$$

where C is a constant depending only on the diameter of Ω, N, and the maximum modulus of the first and second order derivatives of f on $\overline{\Omega}$.

The proof of the above theorem is covered in great details in [12]. Here we just sketch the idea of the proof of the second estimate.

For each $(p,q) \in \Lambda_j$, using Taylor expansion for f at a point $(x,y) \in \Omega$, we have

$$\begin{aligned} f\left(\frac{p+c}{2^j}, \frac{q+c}{2^j}\right) &= f(x,y) + \frac{\partial f}{\partial x}(x,y)\left(\frac{p+c}{2^j} - x\right) + \frac{\partial f}{\partial y}(x,y)\left(\frac{q+c}{2^j} - y\right) + \\ &\quad \frac{1}{2}R(x,y), \end{aligned} \tag{37}$$

where

$$\begin{aligned} R(x,y) &= \frac{\partial^2 f}{\partial x^2}(\xi_p,\xi_q)\left(\frac{p+c}{2^j} - x\right)^2 + 2\frac{\partial^2 f}{\partial x \partial y}(\xi_p,\xi_q)\left(\frac{p+c}{2^j} - x\right)\left(\frac{q+c}{2^j} - y\right) + \\ &\quad \frac{\partial^2 f}{\partial y^2}(\xi_p,\xi_q)\left(\frac{q+c}{2^j} - y\right)^2 \end{aligned}$$

for some (ξ_p, ξ_q) on the segment connecting $\left(\frac{p+c}{2^j}, \frac{q+c}{2^j}\right)$ and (x,y). Substituting (37) into (35) and using the relation $\sum_{k\in\mathbb{Z}} \phi_{jk}(x) = 2^{j/2}$, we have

$$\begin{aligned} 2^j(f - f^j)(x,y) &= \sum_{(p,q)\in\Lambda_j} \frac{\partial f}{\partial x}(x,y)\left(\frac{p+c}{2^j} - x\right)\phi_{jp}(x)\phi_{jq}(y) + \\ &\quad \sum_{(p,q)\in\Lambda_j} \frac{\partial f}{\partial y}(x,y)\left(\frac{q+c}{2^j} - y\right)\phi_{jp}(x)\phi_{jq}(y) + \\ &\quad \sum_{(p,q)\in\Lambda_j} R(x,y)\phi_{jp}(x)\phi_{jq}(y). \end{aligned} \tag{38}$$

We first consider the wavelet expansion of the function x

$$x = \sum_{k\in\mathbb{Z}} 2^{-3j/2}(k+c)\phi_{jk}(x),$$

where c is the first moment of ϕ as in (34). Using the relation $\sum_{k\in\mathbb{Z}} \phi_{jk} = 2^{j/2}$ we conclude that

$$\sum_{k\in\mathbb{Z}} (c + k - 2^j x)\phi_{jk} = 0.$$

This implies that the first two summations in (38) vanish, and we obtain

$$2^j (f^j - f)(x, y) = \sum_{(p,q)\in\Lambda_j} R(x, y)\phi_{jp}(x)\phi_{jq}(y).$$

But

$$|R(x, y)| \le C \left(\left| \frac{p + c}{2^j} - x \right|^2 + \left| \frac{p + c}{2^j} - x \right| \left| \frac{q + c}{2^j} - y \right| + \left| \frac{q + c}{2^j} - y \right|^2 \right)$$

where the constant C depends only on the maximum of the second order derivatives of f. Therefore

$$2^{2j}|f^j - f|^2 \le C \left(\sum_{(p,q)\in\Lambda_j} \left(\left| \frac{p + c}{2^j} - x \right|^2 + \left| \frac{p + c}{2^j} - x \right| \left| \frac{q + c}{2^j} - y \right| + \left| \frac{q + c}{2^j} - y \right|^2 \right) \phi_{jp}(x)\phi_{jq}(y) \right)^2$$

Hence by the fact that $\|\phi_{jk}\|_0 = 1$ and Hölder inequality, we have

$$2^{2j}\|f - f^j\|^2_{L^2(\Omega)} \le C \sum_{(p,q)\in\Lambda_j} (1/2^j)^4 \le C(1/2^j)^4 card(\Lambda_j) \le C(1/2^{2j}).$$

3.3 Multiresolution analysis of $[0, 1]$

In this section we outline the construction of an MRA of $H_0^1[0, 1]$. The MRA concept cannot be directly extended to $H_0^1[0, 1]$ for the reason that there is no coarser scale than the unit interval itself.

The simplest solution is to use periodic wavelets. They can be adapted to non-periodic conditions by the Chebyshev transform $x = \frac{1}{\pi} \arccos y$ which leads to "Chebychev" wavelets. In 1989, Jaffard and Meyer proposed a construction of wavelet bases on an open set in $\mathbb{R}^n$ starting from spline functions. However, their construction was theoretical and has not yet been implemented numerically. More recently, other constructions of wavelet bases on the interval $[0, 1]$ have been proposed, they are all based on Daubechies compactly supported wavelets in $L^2(\mathbb{R})$. The first construction was done by Meyer. It was rather theoretical and had some drawbacks, the most important being its numerical instability. Another construction avoiding these problems was then proposed and extended in dimension 2. Nevertheless, these wavelets take arbitrary values at the boundaries and are not well adapted to the resolution of boundary value problems.

A typical construction of MRA for $H_0^1[0, 1]$ consists of several steps. Since most of them are too technical, we will give just a rough outline for the scheme used in [9]. First, MRA on $[0, +\infty)$ without boundary conditions, can be constructed using so-called edge scaling functions which are obtained as a linear combination of the original scaling functions. The coefficients in this expansion are the values at the integers of some special polynomial. Second, combining some of these functions with some of the original scaling functions we obtain MRA on $[0, +\infty)$ with boundary conditions. Next, using a simple change of variables $x \mapsto 1 - x$ we get MRA on $(-\infty, 1]$ with boundary conditions. Finally, we merge the last two to construct MRA for the intersection $[0, 1]$.

4 Wavelet Galerkin methods

4.1 Application in rectangular domains

Let us first consider the case where Ω is rectangle. For convenience we assume $\Omega = [0,1]^2$. In this simple situation we will show how to replace finite element approximation space V_h with finite dimensional space $\mathbb{V}_j$, $h = 2^{-j}$ constructed using MRA. This is so-called Wavelet Galerkin approximation.

4.1.1 Periodic boundary conditions

Define

$$H^1_p(\Omega) = \{v \in H^1(\Omega) \ : \ v(0,y) = v(1,y),\ v(x,0) = v(x,1) \quad \forall x,y \in [0,1]\}. \tag{39}$$

The problem (1) with periodic boundary conditions can be reformulated in a weak form similar to **DP**, **NP**. The only difference is that in this case $\mathrm{V} = H^1_p(\Omega)$ (the boundary integral vanishes since Ω is rectangle). We consider this type of boundary conditions partly because it is very easy to demonstrate how one can obtain wavelet approximation space. Indeed, let

$$\mathcal{V}_{j,p} = \{v(x) = \sum_{k\in\mathbb{Z}} c_k \phi_{jk}(x) \ \ x \in (0,1), \quad \text{with } c_k = c_{k+2^j}\}. \tag{40}$$

It can be shown that

$$\lim_{j\to\infty} \mathcal{V}_{j,p} = H^1_p[0,1].$$

Therefore we can define

$$\mathbb{V}_j = \mathcal{V}_{j,p} \otimes \mathcal{V}_{j,p}. \tag{41}$$

Error estimate similar to (22) is given by:

$$\|u - u_{j,p}\|_0 \le \frac{C}{2^{2j}} \|u\|_2. \tag{42}$$

See [12] for more information.

4.1.2 Pure Dirichlet boundary conditions

Consider the problem **DP**. Take any of the MRA of $H^1_0[0,1]$ discussed in §3.3. Define

$$\mathbb{V}_j = \mathcal{V}_j \otimes \mathcal{V}_j. \tag{43}$$

Let us note that in this case the matrix A in (15) will be diagonal.

4.1.3 Pure Neumann boundary conditions

Consider the problem **NP**. Define (see (36))

$$\mathbb{V}_j = \text{span}\,\{\phi_{jp}(x)\phi_{jq}(y) \ : \ (p,q) \in \Lambda_j\}. \tag{44}$$

Clearly $\mathbb{V}_j \subset H^1(\Omega)$ is finite dimensional, since Ω is bounded.

Theorem 9 *Let u_j be the solution of the Galerkin approximation problem using $\mathbb{V}_j$. Then*

$$\|u - u_j\|_1 \le \frac{C}{2^j} \|u\|_2. \tag{45}$$

Proof: Using Theorem 8 in the same manner as the proof of Theorem 3. ∎

Corollary 4

$$\lim_{j\to\infty} \mathbb{V}_j = H^1(\Omega).$$

Remark 10 *Using Nitsche's argument it can be shown that the error estimate in* $L^2(\Omega)$ *is*

$$\|u - u_j\|_0 \leq \frac{C}{2^{2j}} \|u\|_2. \tag{46}$$

Let us note that it will be difficult to compute the entries of the stiffness matrix A for general domain Ω.

4.2 Application in arbitrary domains. Fictitious domain approach

Usually, solving the pure Neumann problem **NP** directly on Ω using finite element method raise a number of computational difficulties. In order to solve **NP** we embed Ω in a larger rectangular domain Ω_R in $\mathbb{R}^2$ (see Figure 6). Now let V be a closed subspace of $H^1(\Omega_R)$ such that

$$\{v : v = \tilde{v}|_\Omega, \tilde{v} \in V\} = H^1(\Omega).$$

Typical choices for V are $H_0^1(\Omega_R)$ and the space $H_p^1(\Omega)$ (see (39)).
Now we introduce the following subspaces of V

$$\begin{aligned} U &= \{v \in V : \mathcal{A}(v, w) = \mathcal{F}(w), \forall w \in V\}, \\ U_0 &= \{v \in V : \mathcal{A}(v, w) = 0, \forall w \in V\}. \end{aligned}$$

Assume that $\mathcal{B}$ is a continuous coercive bilinear form on $V \times V$ and $\mathcal{L}$ is a continuous linear functional on V. We can, for example, define $\mathcal{B}$ by

$$\mathcal{B}(v, w) = \int_{\Omega_R} b\nabla v \cdot \nabla w + \beta\, vw\,,$$

where b and β satisfy conditions similar to (7)-(9). We consider the following problem

$$\begin{cases} \tilde{u} \in U, \\ \mathcal{B}(\tilde{u}, v - \tilde{u}) \geq \mathcal{L}(v - \tilde{u}), \quad \forall v \in U. \end{cases} \tag{47}$$

It can be shown that (47) has a unique solution and the solution is necessarily the solution of

$$\begin{cases} \tilde{u} \in U, \\ \mathcal{B}(\tilde{u}, v) = \mathcal{L}(v), \quad \forall v \in U_0. \end{cases} \tag{48}$$

We consider then the following linear variational problem

$$\begin{cases} u_\epsilon \in V, \\ \mathcal{A}(u_\epsilon, v) + \epsilon\mathcal{B}(u_\epsilon, v) = \mathcal{F}(v) + \epsilon\mathcal{L}(v), \quad \forall v \in V, \end{cases} \tag{49}$$

where ϵ is a positive parameter. This is so-called penalty formulation.

Theorem 10 *Suppose that Ω is a $C^{0,1}$ domain, suppose also that the hypotheses on $\mathcal{A}, \mathcal{B}, \mathcal{F}, \mathcal{L}$ hold. We have then*

$$\begin{aligned} \lim_{\epsilon \to 0} \|u_\epsilon - \tilde{u}\|_{H^1(\Omega_R)} &= 0, \\ \lim_{\epsilon \to 0} \epsilon^{-1/2} \|u_\epsilon - u\|_{H^1(\Omega)} &= 0, \end{aligned}$$

where $u, \tilde{u}$ and u_ϵ are the solutions of problems **NP**, *(48),(49). respectively.*

Now we would like to formulate a Galerkin approximation of the regularized Neumann problem as posed in **NP** where a and b are identity functions and $\alpha > 0$, $\beta > 0$.
Let V_h be a finite dimensional subspace of V. We have to find u_h^ϵ in V_h such that

$$\mathcal{A}(u_h^\epsilon, v_h) + \epsilon \mathcal{B}(u_h^\epsilon, v_h) = \mathcal{F}(v_h) + \epsilon \mathcal{L}(v_h), \quad v_h \in V_h. \tag{50}$$

Problem (50) has a unique solution in V_h by Lax-Milgram theorem. Let $\mathcal{T}_h$ be a regular triangulation of the polygonal domain Ω_R and take V_h to be the FEM approximation space as defined in (18). The following estimates are results of Theorem 4.4, 4.5 in [6]

Theorem 11 *Let Ω be a bounded domain in $\mathbb{R}^2$ with a $C^{k,1}$ boundary for integer $k > 0$, strictly contained in the polygonal open set Ω_R. If the solution of (4) verifies $u \in H^{k+1}(\Omega)$, then there exists a constant C independent of h such that*

$$\|u - u_h^\epsilon\|_{H^1(\Omega)} \leq C(h^k \|u\|_{H^{k+1}(\Omega)} + \sqrt{\epsilon} \|u\|_{H^{k+1}(\Omega)} + \sqrt{\epsilon} \|\tilde{f}\|_{L^2(\Omega_R)}),$$

where u_h^ϵ is the solution of problem (50) and $\tilde{f}$ is an extension of f from Ω to Ω_R.

Theorem 12 *Assume that the conditions in Theorem 11 hold. Then there exists a constant C independent of h such that*

$$\|u - u_h^\epsilon\|_{L^2(\Omega)} \leq C(h^{k+1} \|u\|_{H^{k+1}(\Omega)} + \sqrt{\epsilon} \|u\|_{H^{k+1}(\Omega)} + \sqrt{\epsilon} \|\tilde{f}\|_{L^2(\Omega_R)}).$$

Let ϕ be the Daubechies scaling function of N ($N \geq 3$). Define $\phi_{jk} = 2^{j/2} \phi(2^j x - k)$ for $k, j \in \mathbb{Z}$. We assume that $\Omega_R = (-s, s) \times (-s, s)$ where s is a positive integer. Set

$$\mathbb{V}_j = \{v \in L^2(\Omega_R) : v(x, y) = \sum_{k,l \in \mathbb{Z}} c_{k,l} \phi_{jk}(x) \phi_{jl}(y), \quad (x, y) \in \Omega_R\}$$

The space $\mathbb{V}_j$ (respectively u_j) corresponds to V_h (respectively u_h) with $h = 2^{-j}$. Under the assumption of Theorems 11 and 12, the analysis of section 3 gives

$$\|u - u_j^\epsilon\|_{H^1(\Omega)} = O(2^{-j} + \sqrt{\epsilon}),$$

and

$$\|u - u_j^\epsilon\|_{L^2(\Omega)} = O(2^{-2j} + \sqrt{\epsilon}).$$

5 Numerical experiments

In this section we apply the fictitious domain technique combined with approximation space of Daubechies wavelets to a simple 2D problem (see Figure 6). Specifically $\Omega = [1,2] \times [1,2]$

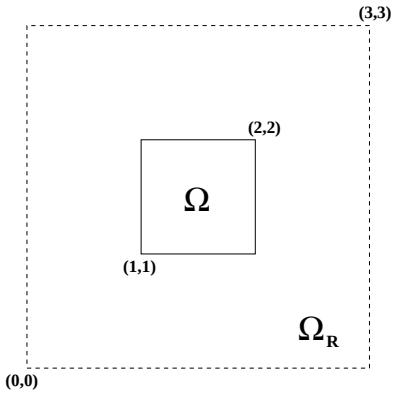

Figure 6: Geometry of the computational domain Ω and the fictitious domain Ω_R

and $\Omega_R = [0,3] \times [0,3]$. The differential equation is **DP** with $a \equiv 1$:

$$-\Delta u = f \ \text{ on } \Omega\,, \quad u = 0 \ \text{ on } \Gamma. \tag{51}$$

The order of the wavelets is $N = 3$. The penalty formulation in this case is:

$$\int_{\Omega_R} \nabla u_\epsilon \nabla v + \frac{1}{\epsilon} \int_{\partial\Omega} u_\epsilon v = \int_{\Omega_R} \tilde{f} v, \tag{52}$$

where $\tilde{f}$ is the actual extension of f to Ω_R. The corresponding linear system reads:

$$\mathrm{A}x + \frac{1}{\epsilon}\mathrm{M}x = \mathrm{b}, \tag{53}$$

where A is a circulant matrix and M corresponds to the boundary integral. We use discretization of level 4, that is $h = 2^{-4}$. We run two different tests with right-hand sides equal to:

$$f(x,y) = \frac{1}{2} \sin \pi x \sin \pi y \tag{54}$$

$$f(x,y) = \begin{cases} 1 & \text{if } |x - 1/2| < 10^{-1} \text{ and } |y - 1/2| < 10^{-1}, \\ 0 & \text{otherwise.} \end{cases} \tag{55}$$

The implementation uses C and Matlab codes, which should be executed as follows:

- Start matlab, run **get_scale(level)** with the chosen level.
- Edit **Wave_C.c** to choose the right-hand side f and set the level of approximation (the same as in the previous step).
- Start **wavelet_plot**.
- The approximate solution is returned in the environment variable H.

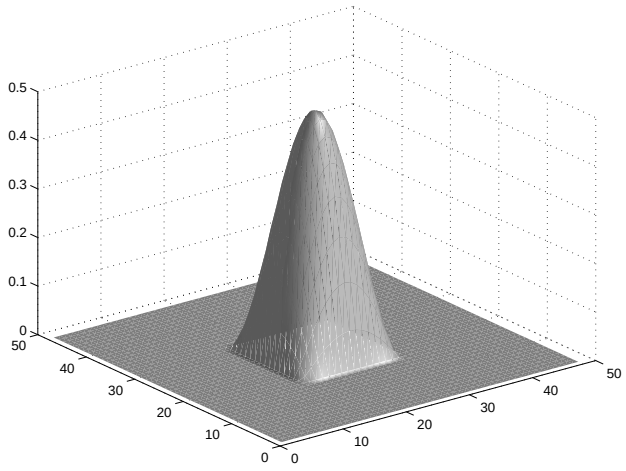

Figure 7: The approximate solution corresponding to (54).

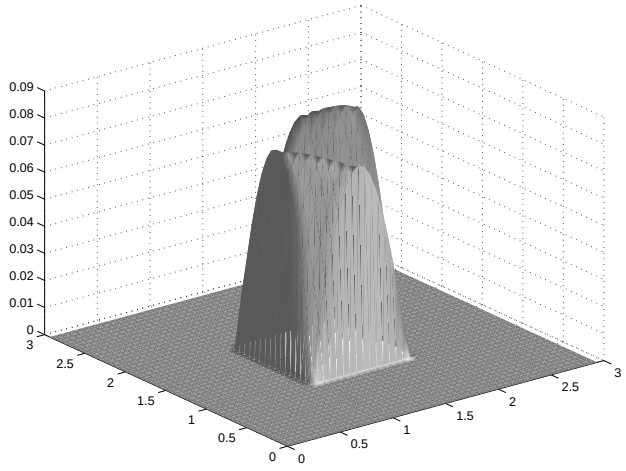

Figure 8: The error of the approximate solution corresponding to (54).

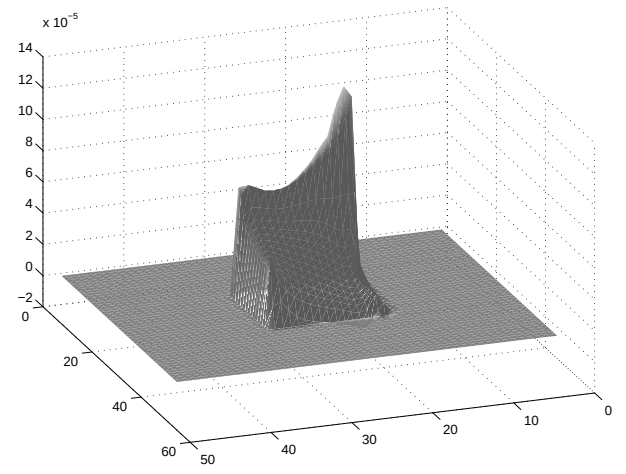

Figure 9: The approximate solution corresponding to (55).

6 MATLAB code

Function 1 (dwav.m) *Plots Daubechies scaling function and wavelet*

```
function dwav(n,p,init)
%DWAV(N,P,INIT) This function plots Daubechies scaling function and wavelet
%
% N - order of \phi (the support is [0,2*N-1])
% P - perform 2^P iterations
% INIT - values of \phi in {0,...,2N-1} are obtained by
%        1 -> solving eigenvalue problem
%        2 -> iteration of scaling relation

% Daubechies filter coefficients /sqrt(2) for N=0...5
coef = ...
[ 1.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.;
  0.707106781187,0.707106781187,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.;
  0.482962913145,0.836516303738,0.224143868042,-0.129409522551, ...
  0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.;
  0.33267055295,0.806891509311,0.459877502118,-0.13501102001, ...
  -0.085441273882,0.035226291882,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.;
  0.230377813309,0.714846570553,0.63088076793,-0.027983769417, ...
  -0.187034811719,0.030841381836,0.032883011667,-0.010597401785,0.,...
  0.,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.;
  0.160102397974,0.603829269797,0.724308528438,0.138428145901, ...
  -0.242294887066,-0.032244869585,0.07757149384,-0.006241490213, ...
  -0.012580751999,0.003335725285,0.,0.,0.,0.,0.,0.,0.,0.,0.,0.];

m=2^p;
if (nargin < 3)
  init=1;
end

% Scaling Function Initialization
% Version1 - using eigenvector of T-{0}
if init==1
  mat=zeros(2*n,2*n);
  for i=1:2*n
    for j=1:2*n
      if ((1 <= 2*i-j) & (2*i-j <= 2*n))
        mat(i,j)=coef(n+1,2*i-j);
      end
    end
  end
  [V,D]=eig(mat*sqrt(2));
  for i=1:2*n
    if (abs(D(i,i)-1.0)< 0.001)
      if n==1
```

```
        eigmat=V(:,i)';
      else
        eigmat=sign(V(2,i))*V(:,i)';
      end
      break;
    end
  end
end

% Scaling Function Initialization
% Version2 - using 1/(2*n) and iterations at integers
if init==2
  eigmat=ones(1,2*n)/(2*n);
  for i=1:100
    for x=0:2*n-1
      s=0;
      for k=1:2*n
        if (0<=2*x-k+1 & 2*x-k+1<=2*n-1)
          s=s+coef(n+1,k)*eigmat(2*x-k+1+1);
        end
      end
      eigmat(x+1)=sqrt(2)*s;
    end
  end
end

% Plot Scaling Function
t=[1:(m*(2*n-1)+1)]/m;
for i=1:length(t)
  y(i)=phi(coef,eigmat,m,n,t(i));
end
figure
plot(t,y);
title(['Scaling function {\phi}(x) for N=',num2str(n)]);
axis normal; grid on;

% Plot Wavelet
for i=1:length(t)
  y(i)=psi(coef,eigmat,m,n,t(i));
end
figure
plot(t,y);
title(['Wavelet function {\psi}(x) for N=',num2str(n)]);
axis normal; grid on;

% Scaling Relation
function y=phi(coef,eigmat,m,n,x)
if (x<0 | x>2*n-1)
```

```
  y=0;
elseif (x-floor(x) < 0.0001)
  y=eigmat(x+1);
elseif (m*x-floor(m*x) < 0.0001)
  s=0;
  for k=1:2*n
    s=s+coef(n+1,k)*phi(coef,eigmat,m,n,2*x-k+1);
  end
  y=sqrt(2)*s;
else
  y=0;
end

% Wavelet Relation
function y=psi(coef,eigmat,m,n,x)
if (x<0 | x>2*n-1)
  y=0;
elseif (m*x-floor(m*x) < 0.0001)
  s=0;
  for k=1:2*n
    s=s+(-1)^(k+1)*coef(n+1,2*n-k+1)*phi(coef,eigmat,m,n,2*x-k+1);
  end
  y=sqrt(2)*s;
else
  y=0;
end
```

Function 2 (w2d.m) *Plots the tensor square of specified scaling function. Uses Wavelet Toolbox Function wavefun (change 'db3' to any wavelet name in the toolbox).*

```
[phi,psi,x] = wavefun('db3',7);
Z=phi'*phi;
surfl(x,x,Z);
shading interp;
colormap(gray);
axis equal tight;
view(58.50,24);
```

Function 3 (get_scale.m) *Computes the values of the Daubechies scaling function corresponding to arguments stored in 'scale_x.txt' and saves them in 'scale_value.txt'.*

```
function get_scale(x)
stream1=fopen('scale_value.txt','w');
stream2=fopen('scale_x.txt','w');
[sca,wav,x]=wavefun('db3',x);
fprintf(stream1,'%10.10f\n',sca);
fprintf(stream2,'%10.10f\n',x);
```

Function 4 (WaveC.c) *Computes the right-hand side of (53).*

```
/* to compile: cc Wave_C.c -lm */

#include<stdio.h>
#include<math.h>
#include<time.h>

#define  lev 4
#define  SIZE  (5*2*2*2*2+1)
#define  PI2   (2*asin(1))*(2*asin(1))
#define  PI    2*asin(1)

double f(double x,double y)
{
  /* return PI2*(sin(PI*x))*sin(PI*y); */
  return sin(x*x+y*y);
}

int pow2(int x,int y)
{
int i, t=1;
for(i=0;i<y;i++)
t=t*x;
return t;
}

double integral_right_f1(int m1,int m2,double *sca_x,double *x,
double*sca_y,double *y,double A[][SIZE])
{
double c; int M,N,Nj,Ni,j,i,G; M=pow2(2,lev);

N=5*M;
c=0;
Nj=N;
Ni=N;
G=3*M;

if (G-m2<5)
Nj=(G-m2)*M;

if (G-m1<5)
Ni=(G-m1)*M;

for(j=0;j<Nj-1;j++)
for(i=0;i<Nj-1;i++)
c=c+(A[j][i]+A[j][i+1]+A[j+1][i]+A[j+1][i+1])*
f((x[i+1]+m1+x[i]+m1)/(2*M),(y[j+1]+m2+y[j]+m2)/(2*M));
```

```
c=c/pow2(M,3)/4;

return c;
}

void main()
{
clock_t begin, end;
double *sca_x,*sca_y,*x,*y,*A, B[SIZE][SIZE];
FILE *input_file1,*input_file2,*output_file3;
int M,N,j,i,s,r,n;

begin=clock();

M=pow2(2,lev);
N=5*M;
sca_x=(double *)malloc((N+1)*sizeof(double));
sca_y=(double *)malloc((N+1)*sizeof(double));
x=(double *)malloc((N+1)*sizeof(double));
y=(double *)malloc((N+1)*sizeof(double));

input_file1=fopen("scale_value.txt","r");
input_file2=fopen("scale_x.txt","r");
for(i=0;i<N+1;i++)
{
fscanf(input_file1,"%lf",&sca_x[i]);
sca_y[i]=sca_x[i];
fscanf(input_file2,"%lf",&x[i]);
y[i]=x[i];
}

for(j=0;j<N+1;j++)
for(i=0;i<N+1;i++)
B[j][i]=sca_y[j]*sca_x[i];

remove("vector_value.txt");
output_file3=fopen("vector_value.txt","w");

n=3*M;
A=(double *)malloc((n*n)*sizeof(double));
for(s=0;s<n;s++)
for(r=0;r<n;r++)
{
A[r+s*n]=integral_right_f1(r,s,sca_x,x,sca_y,y,B);
fprintf(output_file3,"%10.10f\n",A[r+s*n]);
}
```

```
fclose(input_file1);
fclose(input_file2);
fclose(output_file3);

end=clock();
input_file1=fopen("time.txt","w");
fprintf(input_file1,"Time used is %10.10f ",
(double)(end-begin)/CLOCKS_PER_SEC/60);
fclose(input_file1);
}
```

Function 5 (stiff_first.m) *Computes the matrix A in (53).*

```
function A=stiff_first

T(1)=5.267857142857143;
T(2)=-3.390476190476190;
T(3)=0.8761904761904762;
T(4)=-0.11428571428571429;
T(5)=-0.005357142857142857;

global lev;
M=2^lev;
N=3*M;
A=sparse(N,N);
%*******************************************************ONE
for s=0:N-1,
for r=0:N-1,
%for q=0:N-1,
for p=0:N-1,
H=abs(p-r);

if H<5,
m1=T(H+1);
elseif H>N-5,
m1=T(N-H+1);
else,
m1=0;
end

A(r+1+s*N,p+1+s*N)=m1;
end
end
end
%*****************************************************SECOND
for s=0:N-1,
for r=0:N-1,
for q=0:N-1,
```

```
W=abs(q-s);
if W<5,
m2=T(W+1);
elseif W>N-5,
m2=T(N-W+1);
else,
m2=0;
end

A(r+1+s*N,r+1+q*N)=A(r+1+s*N,r+1+q*N)+m2;
end
end
end
%*********************************************
A=A.*(M^2);
```

Function 6 (stiff_second.m) *Computes the matrix M in (53).*

```
function B=stiff_second

[sca,wav,x]=wavefun('db3',10);
H=2^10;

global lev;
M=2^lev;
M2=M*2;
N=3*M;
size=N*N;

B1=sparse(size,size);
%********************************************************** ONE
for s=M-4:M-1,
for r=M-4:M-1,
for q=M-4:M-1,
for p=M-4:M-1,
B1(r+1+s*N,p+1+q*N)=integral(p,r)*sca(H*(M-q))*sca(H*(M-s));
end
end
end
end

for s=M-4:M-1,
for r=M:M2-5,
for q=M-4:M-1,
B1(r+1+s*N,r+1+q*N)=B1(r+1+s*N,r+1+q*N)+sca(H*(M-q))*sca(H*(M-s));
end
end
```

```
end

for s=M-4:M-1,
for r=M2-4:M2-1,
for q=M-4:M-1,
for p=M2-4:M2-1,
B1(r+1+s*N,p+1+q*N)=B1(r+1+s*N,p+1+q*N)+integral(p,r)*sca(H*(M-q))*sca(H*(M-s));
end
end
end
end

%*********************************************************SECOND
for s=M-4:M-1,
for r=M2-4:M2-1,
for q=M-4:M-1,
for p=M2-4:M2-1,
B1(r+1+s*N,p+1+q*N)=B1(r+1+s*N,p+1+q*N)+integral(q,s)*sca(H*(M2-p))*sca(H*(M2-r));
end
end
end
end

for s=M:M2-5,
for r=M2-4:M2-1,
for p=M2-4:M2-1,
B1(r+1+s*N,p+1+s*N)=B1(r+1+s*N,p+1+s*N)+sca(H*(M2-p))*sca(H*(M2-r));
end
end
end

for s=M2-4:M2-1,
for r=M2-4:M2-1,
for q=M2-4:M2-1,
for p=M2-4:M2-1,
B1(r+1+s*N,p+1+q*N)=B1(r+1+s*N,p+1+q*N)+integral(q,s)*sca(H*(M2-p))*sca(H*(M2-r));
end
end
end
end
%*********************************************************************THREE
for s=M2-4:M2-1,
for r=M-4:M-1,
for q=M2-4:M2-1,
for p=M-4:M-1,
B1(r+1+s*N,p+1+q*N)=B1(r+1+s*N,p+1+q*N)-integral(p,r)*sca(H*(M2-q))*sca(H*(M2-s));
end
end
```

```
end
end

for s=M2-4:M2-1,
for r=M:M2-5,
for q=M2-4:M2-1,
B1(r+1+s*N,r+1+q*N)=B1(r+1+s*N,p+1+q*N)-sca(H*(M2-q))*sca(H*(M2-s));
end
end
end

for s=M2-4:M2-1,
for r=M2-4:M2-1,
for q=M2-4:M2-1,
for p=M2-4:M2-1,
B1(r+1+s*N,p+1+q*N)=B1(r+1+s*N,p+1+q*N)-integral(p,r)*sca(H*(M2-q))*sca(H*(M2-s));
end
end
end
end
%*********************************************************************FOUR
for s=M-4:M-1,
for r=M-4:M-1,
for q=M-4:M-1,
for p=M-4:M-1,
B1(r+1+s*N,p+1+q*N)=B1(r+1+s*N,p+1+q*N)-integral(q,s)*sca(H*(M-p))*sca(H*(M-r));
end
end
end
end

for s=M:M2-5,
for r=M-4:M-1,
for p=M-4:M-1,
B1(r+1+s*N,p+1+s*N)=B1(r+1+s*N,p+1+s*N)-sca(H*(M-p))*sca(H*(M-r));
end
end
end

for s=M2-4:M2-1,
for r=M-4:M-1,
for q=M2-4:M2-1,
for p=M-4:M-1,
B1(r+1+s*N,p+1+q*N)=B1(r+1+s*N,p+1+q*N)-integral(q,s)*sca(H*(M-p))*sca(H*(M-r));
end
end
end
end
```

```
%*********************************************************************
B=B1.*M;
```

Function 7 (wavelet_plot.m) *Computes the approximate solution and plots it.*

```
global lev;
lev=4;
e=1/(2^(2*lev));
M=(3*(2^lev))^2;
C=sparse(M,M);

%************************************
A1=stiff_first
A2=stiff_second
C=A1+A2./e;
%************************************
clear stiff_first stiff_second;
%*************************************************
stream=fopen('vector_value.txt','r');
RHS=fscanf(stream,'%f');
U=C\RHS;
RHS
%*************************************************
U=U.*2^lev;
%*****************************************
Z=zeros(3*(2^lev),3*(2^lev));
for i=0:3*2^lev-1,
Z(:,i+1)=U(i*3*2^lev+1:(i+1)*3*2^lev);
end
H=zeros(3*(2^lev),3*(2^lev));
H(2^lev+1:2*2^lev-1,2^lev+1:2*2^lev-1)=Z(2^lev+1:2*2^lev-1,2^lev+1:2*2^lev-1);
x=0:1/2^lev:3-1/2^lev;
y=x;
[X,Y]=meshgrid(x,y);
mesh(H);
```

References

[1] A. Boggess, F. Narcowich, *Fourier Analysis and Wavelets*, Manuscript, **2000**.

[2] S. Brenner, L. Scot, *The Mathematical Theory of Finite Element Methods*, Springer-Verlag, New York, **1994**.

[3] P. Ciarlet, *The Finite Element Method for Elliptic Problems*, Elsevier North-Holland, New York, **1978**.

[4] R. Glowinski, A. Rieder, R. Wells Jr., X. Zhou, *A Preconditioned CG-Method for Wavelet Galerkin Discretizations of Elliptic Problems*, Technical Report CML TR94-14, Computational Mathematics Laboratory - Rice University, **1994**.

[5] R. Glowinski, A. Rieder, R. Wells Jr., X. Zhou, *A Wavelet Multigrid Preconditioner for Dirichlet Boundary Value Problems in General Domains*, RAIRO Modél. Math. Anal. Numér., 30, p.711–729, **1996**.

[6] R. Glowinski, T. Pan, R. Wells Jr., X. Zhou, *Wavelet and Finite Element Solutions for the Neumann Problem using Fictitious Domains*, J. Comput. Phys., 126, p.40–51, **1996**.

[7] A. Iontcheva, *Numerical Solution of Differential Equations Using Wavelet Integrals*, Recent Adv. in Numer. Methods and Appl., World scientific, Singapore, **1999**.

[8] F. Koster, M. Griebel, N. Kevlahan, M. Farge, K. Schneider, *Towards an Adaptive Wavelet-Based 3D Navier-Stokes Solver*, Notes Numer. Fluid Mech., 66, Vieweg, Braunschweig, **1998**.

[9] P. Monasse, V. Perrier, *Orthonormal Wavelet Bases Adapted for Partial Differential Equations with Boundary Conditions*, SIAM J. Math. Anal., 29, p.1040-1065, **1998**.

[10] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing, New York, **1996**.

[11] R. Wells Jr., X. Zhou, *Wavelet Solutions for the Dirichlet Problem*, Numer. Math., 70, p.379–396, **1995**.

[12] R. Wells Jr., X. Zhou, *Wavelet Interpolation and Approximate Solutions of Elliptic Partial Differential Equations*, NATO Adv. Sci. Inst. Ser. C Math. Phys. Sci., 429, Kluwer Acad. Publ., Dordrecht, **1994**.

[13] J. Xu, W. Shann, *Galerkin-wavelet Methods for Two-Point Boundary Value Problems*, Numer. Math., 63, p.123–144, **1992**.