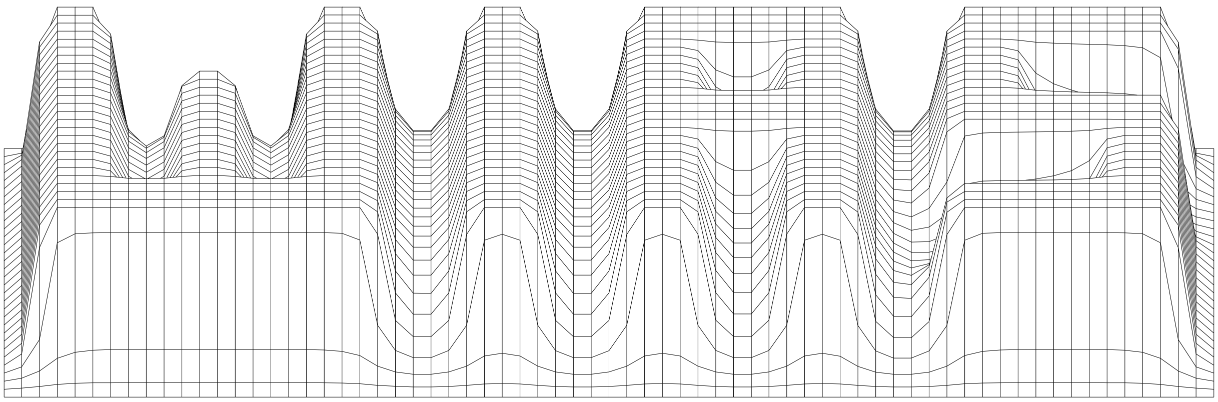
A Wavelet Algorithm for the Boundary Element Solution of a Geodetic Boundary Value Problem

Andreas Rathsfeld
Weierstraß-Institut
für Angewandte Analysis und Stochastik
Mohrenstr. 39
D-10117 Berlin
Germany
rathsfeld@wias-berlin.de

September 19, 1996



1991 Mathematics Subject Classification. 45L10, 65R20 ,65N38.

Keywords. singular integral equation, collocation, wavelet algorithm.

Abstract

In this paper we consider a piecewise bilinear collocation method for the solution of a singular integral equation over a part of the surface of the earth. This singular equation is the boundary integral equation corresponding to the oblique derivative boundary problem for Laplace's equation. We introduce special wavelet bases for the spaces of test and trial functions. Analogously to well-known results on wavelet algorithms, the stiffness matrices with respect to these bases can be reduced to sparse matrices such that the assembling of the matrices and the iterative solution of the matrix equations become fast. Though the theoretical results apply only to integral equations with "smooth" solutions over "smooth" manifolds, we present numerical tests for a geometry as difficult as the surface of the earth.

0 Introduction

It is a well-known fact that usual finite element discretisations of linear integral equations (e.g. of boundary integral equations) lead to systems of linear equations with fully populated matrices. Thus, even an iterative solution method requires a huge number of arithmetic operations and a large storage capacity. In order to improve these finite element approaches, several new algorithms have been developed. For a relatively wide class of boundary integral equations, Rokhlin and Greengard [30, 15] have introduced their methods of multipole expansion, Hackbusch and Nowak [16] (cf. also [31]) have considered panel clustering algorithms, and Brandt and Lubrecht [3] have set up multilevel schemes. Another approach for saving storage and computation time consists in employing wavelet bases of the finite element spaces. This idea goes back to Beylkin, Coifman, and Rokhlin [2], and has been thoroughly investigated by Dahmen, v.Petersdorff, Prößdorf, Schneider, and Schwab [10, 11, 9, 12, 26, 25, 24, 32] (cf. also the contributions by Alpert, Harten, Yad-Shalom, Dorobantu, Kleemann, and the author [1, 17, 14, 6, 7, 29]). Note that all the different algorithms from multipole expansion to wavelets seem to have a common multilevel background.

The subject of the present paper is to apply the wavelet technique from [2] to the solution of a two-dimensional singular integral equation which corresponds to the oblique derivative problem for Laplace's equation. For the computation of the gravity field of the earth in geodesy, this equation has to be considered over the surface of the earth or part of it (cf. Klees [19, 18] or cf. the similar equation for the Molodensky problem in Moritz [22], Section 43). If the underlying surface is smooth (piecewise continuously differentiable up to a certain order), then it is clear that the wavelet algorithms (cf. [9, 25]) admit high order compressions. In addition, if the surface is piecewise analytic, then the algorithms are fast (cf. [32, 25]). The surface of the earth, however, cannot supposed to be smooth. Thus, we will apply a wavelet algorithm for a complicated real-life geometry and report on numerical tests for a situation where no proofs are available yet. For proofs of compression and complexity results under low order smoothness assumptions, we refer to the paper [28], where a slightly modified algorithm over a closed manifold is treated.

Following Klees [19, 18], we will describe the problem of oblique derivative, the corresponding boundary integral equation, and its discretisation via piecewise bilinear collocation over uniform grids in Section 1. In Section 2 we will introduce the wavelet algorithm based on biorthogonal wavelets for the bilinear trial space (cf. Cohen, Daubechies, and

Feauveau [5] as well as their application in [26, 29]) and on wavelets defined by Harten and Yad-Shalom [17] for the space of test functionals (cf. also the earlier paper by Brandt and Lubrecht [3]). We will discuss a compression scheme similar to that of Dahmen, Prößdorf, Schneider, v.Petersdorff, and Schwab [9, 25]. For the number of degrees of freedom n, it admits a reduction from n^2 entries in the full matrix to $n^{4/3}$ entries in the compressed sparse matrix. Though this compression is not asymptotically optimal, we expect shorter computation times for $n \leq 10\,000$. Section 3 is devoted to the quadrature algorithm for the computation of the stiffness matrix implemented in the numerical tests. The rules are based on singularity subtraction and Duffy's transformation. We present further details of the algorithm and the results of the numerical tests in Section 4. For a $9\,025 \times 9\,025$ matrix, it turns out that, in order to keep the relative compression error less than 10^{-5}, it is sufficient to compute only 5.1% of the stiffness matrix. The computing time for a $2\,025 \times 2\,025$ system can be reduced from 669s (seconds) in case of the conventional finite element code to 125s for the wavelet algorithm. The saving of computing time is likely to be much higher for the $9\,025 \times 9\,025$ system. Unfortunately, the main memory of 96 MB was not enough to perform the test for conventional finite elements.

In summary, the wavelet algorithm presented here provides a good tool for an improvement in computing time and storage requirements even for complicated geometries. However, the question remains open which of the fast methods from multipole expansion to wavelets is the best. For an answer, a lot of further numerical test would be necessary. From the theoretical point of view, panel clustering seems to be more suitable for the test example in consideration since the kernel function of the integral operator is, roughly speaking, the restriction of an analytic function in space to the complicated surface (The kernel function is the sum of products of such analytic kernels with non-smooth functions depending on one variable only!). If a Galerkin discretisation is chosen, then even the sparse grid methods of Zenger [33] can be applied successfully. This approach is closely related to a wavelet method where the higher dimensional wavelets are replaced by tensor products of univariate wavelets.

1 The Boundary Element Method for the Solution of a Geodetic Boundary Value Problem

1.1. The Singular Integral Equation

In this section we recall the boundary element approach of Klees [19, 18] (cf. also [21], Section 23). The potential w of the gravity field in the external space Ω_a of the earth's body can be represented as a sum of a known reference potential w_0 and an unknown difference $\delta w = w - w_0$. If $g = |\nabla w|$ is given by measurements on the earth's surface Γ and if a small order term in the non-linear boundary condition $|\nabla(\delta w + w_0)| = g$ is neglected, then δw is the solution of the boundary value problem

$$\begin{aligned} \triangle \delta w &= 0, \text{ in } \Omega_a, && (1.1)\\ \langle \tilde{\tau}, \nabla \delta w \rangle &= \tilde{v}, \text{ on } \Gamma, && (1.2) \end{aligned}$$

where

$$\tilde{v} := \frac{g^2 - |\nabla w_0|^2}{2}, \quad \tilde{\tau} := \nabla w_0.$$

We seek a solution δw in form of a single layer potential

$$\delta w(z) := \frac{1}{4\pi} \int_\Gamma \frac{\tilde{u}(y)}{|y-z|} d_y \Gamma \tag{1.3}$$

with a yet unknown density function $\tilde{u}$ on Γ. Substituting (1.3) into the boundary condition (1.2) and applying the jump relations for the derivative of the single layer potential, we arrive at to the singular integral equation

$$-\frac{1}{2} \cos[\tilde{n}(x), \tilde{\tau}(x)] \tilde{u}(x) + \frac{1}{4\pi} p.v. \int_\Gamma \frac{\cos[\tilde{\tau}(x), y-x]}{|y-x|^2} \tilde{u}(y) d_y \Gamma = \tilde{v}(x), \quad x \in \Gamma, \tag{1.4}$$

where $\tilde{n}(x)$ denotes the unit normal of the surface Γ at x pointing into Ω_a, and the integral over Γ is to be understood in the principal value sense (cf. [20]). By $\cos[\tilde{n}(x), \tilde{\tau}(x)]$ we denote the cosine of the angle between the vectors $\tilde{n}(x)$ and $\tilde{\tau}(x)$ and $\cos[\tilde{\tau}(x), y-x]$ is defined analogously. Throughout this paper we will assume $\cos[\tilde{n}(x), \tilde{\tau}(x)] > 0$ for any $x \in \Gamma$. This guarantees that the operator on the left-hand side of (1.4) is strongly elliptic (cf. [19, 20]). Moreover, we will assume that the null space of the operator is trivial. This together with the strong ellipticity implies the operator on the left-hand side of (1.4) to be invertible in the space $L^2(\Gamma)$.

If we wanted to discretise the whole surface of the earth, any work station would be likely to collapse because of the enormous number of grid points we would have to take into account. Therefore, we restrict the domain to a rectangular part of Γ which we denote by the same letter. More precisely, Γ is the set of points with latitude between 48.60^0 and 56.65^0 and with longitude between 5.35^0 and 13.40^0. The surface is given by a C^1 parametrisation F over the square $S := [0,1] \times [0,1]$. This parametrisation is the tensor product Overhauser interpolation (cf. [23]) over a uniform rectangular grid. Using F, we arrive at the equivalent singular integral equation $Au = v$ of the form

$$-\frac{1}{2} \cos[n(P), \tau(P)] u(P) + \frac{1}{4\pi} p.v. \int_S \frac{\cos[\tau(P), F(Q) - F(P)]}{|F(Q) - F(P)|^2} F'(Q) u(Q) d_Q S = v(P), \tag{1.5}$$

where P runs over S, $n(P) := \tilde{n}(F(P))$, $\tau(P) := \tilde{\tau}(F(P))$, $u(P) := \tilde{u}(F(P))$, $v(P) := \tilde{v}(F(P))$, and

$$F'(Q) := |\partial_x F((x,y)) \times \partial_y F((x,y))| \tag{1.6}$$

for $Q = (x,y) \in S$. Equation (1.5) is uniquely solvable in $L^2(S)$, and we are going to solve it numerically by bilinear collocation. If u_h denotes the approximate solution of u, then, to get an approximate value $\delta w_h(z)$ to the difference potential $\delta w(z)$, we set $\tilde{u}_h(F(P)) := u_h(P)$ and

$$\delta w_h(z) := \frac{1}{4\pi} \int_\Gamma \frac{\tilde{u}_h(y)}{|y-z|} d_y \Gamma = \frac{1}{4\pi} \int_S \frac{u_h(P)}{|F(P) - z|} F'(P) d_P S. \tag{1.7}$$

1.2. The Collocation Method

In order to set up the collocation system, we introduce a basis of trial functions $\{\varphi_I^N\}$ and the collocation points $\{P_I^N\}$. We choose an integer N and set $h := \frac{1}{N-1}$, $x_i := ih$, $i = 0, 1, \ldots, N-1$ as well as $P_I := P_I^N := (x_i, x_j)$ for $I \in \mathcal{M} := \{(i,j) : i,j = 1, \ldots, N-2\}$. Over the uniform grid $\{x_i\}$ on $[0,1]$, we define the scaled hat functions

$$\varphi_i^N(x) := \frac{1}{\sqrt{h}} \begin{cases} \frac{x - x_{i-1}}{h} & \text{if } x_{i-1} < x < x_i \\ \frac{x_{i+1} - x}{h} & \text{if } x_i < x < x_{i+1} \\ 0 & \text{else} \end{cases} \tag{1.8}$$

for $i = 1, \ldots, N-2$. By taking tensor products, we obtain $\varphi_I^N(Q) := \varphi_i^N(x) \cdot \varphi_j^N(y)$ for $I \in \mathcal{M}$ and $Q = (x,y) \in S$. Note that the trial functions from the space $\mathcal{X}_h :=$span$\{\varphi_I^N : I \in \mathcal{M}\}$ vanish on the boundary of S.

Now with the collocation method one seeks an approximate solution $u_h = \sum_{I \in \mathcal{M}} \xi_I \varphi_I^N$, $\xi_I \in \mathbb{R}$, which fulfills Equation (1.5) at least at the collocation points P_I, $I \in \mathcal{M}$, i.e.,

$$Au_h(P_I) = v(P_I), \; I \in \mathcal{M}. \tag{1.9}$$

Obviously, (1.9) is equivalent to the matrix equation $A_h \xi = \eta$ with $\xi := \{\xi_I\}_{I \in \mathcal{M}}$, $\eta := \{\eta_I\} := \{hv(P_I)\}_{I \in \mathcal{M}}$ and $A_h := (h(A\varphi_I^N)(P_J))_{J,I \in \mathcal{M}}$. The matrix A_h can be identified with the operator in the space $\mathcal{X}_h$ whose matrix with respect to the basis $\{\varphi_I^N : I \in \mathcal{M}\}$ coincides with A_h. The space $\mathcal{X}_h$ is a subspace of $L^2(S)$. Now the collocation method is called stable if, for h sufficiently small, the approximate operators A_h are invertible and the norms $\|A_h^{-1}\|_{\mathcal{L}(\mathcal{X}_h)}$ are uniformly bounded. In other words, the collocation is stable if and only if the l^2 condition numbers of the matrices A_h are uniformly bounded with respect to h. The following theorem is due to Prößdorf and Schneider [27].

Theorem 1.1 *If the surface Γ is sufficiently smooth, then the piecewise bilinear collocation method is stable. Moreover, if the exact solution u belongs to the Sobolev space H^2 over S and vanishes at the boundary of S, then there exists a constant C such that $\|u - u_h\|_{L^2} \leq C\, h^2$ holds for the approximate solutions u_h of the collocation method. Furthermore, for a fixed $z \in \Omega_a$ and δw_h from (1.7), the estimate $|D^\beta \delta w(z) - D^\beta \delta w_h(z)| \leq C\, h^2$ is valid where D^β denotes the partial derivative of multiindex β and the constant C depends on β, z, and u but not on h.*

Remark 1.1 *Obviously, the error for the numerical approximations to the solution u of (1.5) and to the potential δw is affected by the restriction of the boundary integral equation to a part of the earth's surface and by employing the Overhauser parametrisation defined over the same grid as the trial functions. Though a rigorous analysis of the collocation method would require the investigation of these influences, we will ignore them throughout the present paper.*

2 The Wavelet Algorithm

2.1. Basis Transformations and the Wavelet Transform of the Stiffness Matrix

The wavelet algorithm for the bilinear collocation method relies on the introduction of two new wavelet bases which will be defined in the multilevel settings of the next subsections. The first is the basis $\{\psi_I,\, I \in \mathcal{N}\}$ of wavelet basis functions in the space $\mathcal{X}_h$. The set $\mathcal{N}$ is a new index set with the same cardinality as $\mathcal{M}$. This basis $\{\psi_I,\, I \in \mathcal{N}\}$ is chosen such that the functions ψ_I have a possibly small support and such that certain moment conditions are satisfied, i.e., the ψ_I are orthogonal to polynomials of an order less than a certain prescribed number. Similarly, we chose a wavelet basis $\{\vartheta_I,\, I \in \mathcal{N}\}$ in the space of test functionals spanned by the functionals of function evaluation at the collocation points. In fact, the ϑ_I will be linear combinations of a small number of function evaluations. Analogously to the moment conditions for the ψ_I, we require that these functionals vanish at polynomials of low order. We arrive at

$$\begin{aligned} \mathcal{X}_h = \quad \operatorname{span}\{\varphi_I,\, I \in \mathcal{M}\} &= \operatorname{span}\{\psi_I,\, I \in \mathcal{N}\}, &(2.1)\\ \operatorname{span}\{h\delta_{P_I},\, I \in \mathcal{M}\} &= \operatorname{span}\{\vartheta_I,\, I \in \mathcal{N}\}. &(2.2)\end{aligned}$$

In view of these new bases, the collocation equation (1.9) is equivalent to

$$\langle Au_h, \vartheta_J\rangle = \langle v, \vartheta_J\rangle,\ J \in \mathcal{N}, \qquad u_h = \sum_{I\in\mathcal{N}} \mu_I \psi_I. \tag{2.3}$$

The matrix equation $A_h\xi = \eta$ can be replaced by $B_h\mu = \nu$, where $\nu := \{\nu_J\}_{J\in\mathcal{N}} := \{\langle v, \vartheta_J\rangle\}_{J\in\mathcal{N}}$ and $\mu := \{\mu_J\}_{J\in\mathcal{N}}$. The matrix $B_h := (\langle A\psi_I, \vartheta_J\rangle)_{J,I\in\mathcal{N}}$ is called the wavelet transform of A_h. We define the basis transform E_h by $E_h\{\xi_I\}_{I\in\mathcal{M}} = \{\mu_I\}_{I\in\mathcal{N}}$ for $\sum_{I\in\mathcal{M}} \xi_I\varphi_I = \sum_{J\in\mathcal{N}} \mu_J\psi_J$ and the basis transform R_h by $R_h\{\nu_I\}_{I\in\mathcal{N}} = \{\eta_I\}_{I\in\mathcal{M}}$, where $\eta_I := hv_h(P_I)$ and v_h is the unique function in $\mathcal{X}_h$ with $\langle v_h, \vartheta_J\rangle = \nu_J$. Then we get $A_h = R_hB_hE_h$.

Now the wavelet algorithm looks as follows. We solve the equation $A_h\xi = \eta$ iteratively (e.g. by classical iteration, Jacobi's iteration or by GMRes). The main part of the computation is spent for the multiplication of iterative solutions $\tilde{\xi}$ or residual vectors $\tilde{\xi}$ by the matrix A_h. In the wavelet algorithm, this step is done by first multiplying $\tilde{\xi}$ by E_h, then by B_h, and finally by R_h. As we will see in the next subsections, the basis transforms $\tilde{\xi} \mapsto E_h\tilde{\xi}$ and $[B_hE_h\tilde{\xi}] \mapsto R_h[B_hE_h\tilde{\xi}]$ can be realised via fast pyramid type algorithms. For the multiplication by B_h, we will prove that, due to the moment conditions and the smallness of the supports of the bases $\{\vartheta_I,\, I \in \mathcal{N}\}$ and $\{\psi_I,\, I \in \mathcal{N}\}$, the majority of entries in B_h is very small (cf. Lemma 2.3). Thus, setting these entries equal to zero, we end up with a compressed matrix C_h and the multiplication by B_h can be replaced by the multiplication with C_h. The additional error due to the compression will be less than the discretisation error of the collocation (cf. Theorem 2.1). Since the matrix C_h is sparse, the multiplication by C_h is fast.

Remark 2.1 *In some situations it is possible to solve $B_h\mu = \nu$ directly. For instance, if the transforms E_h, R_h are uniformly bounded in suitable Sobolev spaces and if the* *norm of a trial function u_h can easily be expressed in terms of its wavelet coefficients, then B_h admits a diagonal preconditioning and $B_h\mu = \nu$ can be solved with a number of iterations independent of the mesh size h. For details we refer to the papers by Dahmen, Kunoth, Prößdorf, and Schneider [8, 11]. In the situation considered in the present paper, however, R_h will be unbounded and no diagonal preconditioner is known. In general, if the condition number of the original matrix A_h is uniformly bounded, then the actual value of the condition number of the wavelet transform B_h is often much worse even if it is uniformly bounded. For this case, the iterative solution of the original matrix equation is faster.*

In any case, the main part of the computing time for boundary element methods is spent for the calculation of the stiffness matrix. For the wavelet algorithm, we do not need the whole matrices A_h or B_h but only the compressed matrix C_h which saves a lot of computing time. However, this reduction in computing time is not so easy to achieve as it might seem at first glance. In fact, a sophisticated algorithm of quadrature is needed to guarantee small quadrature errors and to reduce the amount of work. We will discuss this issue in Section 3.

2.2. The Bilinear Wavelets for the Trial Space

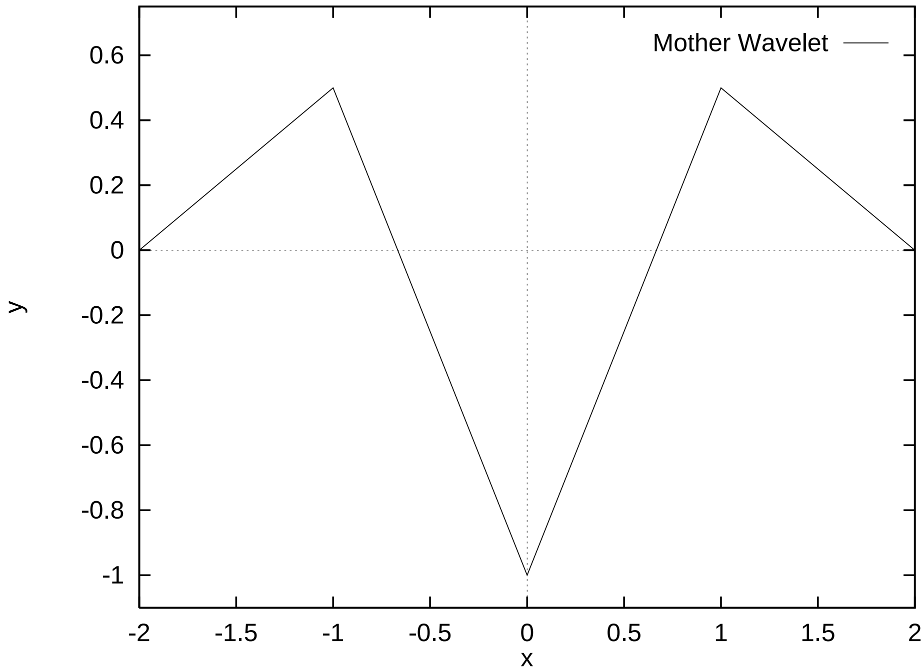


Figure 1: Mother wavelet ψ.

Set $N_l = 3 \cdot 2^l + 1,\ l = 0, 1, \ldots$ and suppose the number N in Section 1.2 is equal to N_L with $L \geq 1$. Recall that $h = \frac{1}{N_L - 1}$. Similarly to the functions φ_i^N and $\varphi_{(i,j)}^N$ of Section 1.2, we define $\varphi_i^{N_l}$ and $\varphi_{(i,j)}^{N_l}$, respectively. We begin with univariate wavelets and introduce the

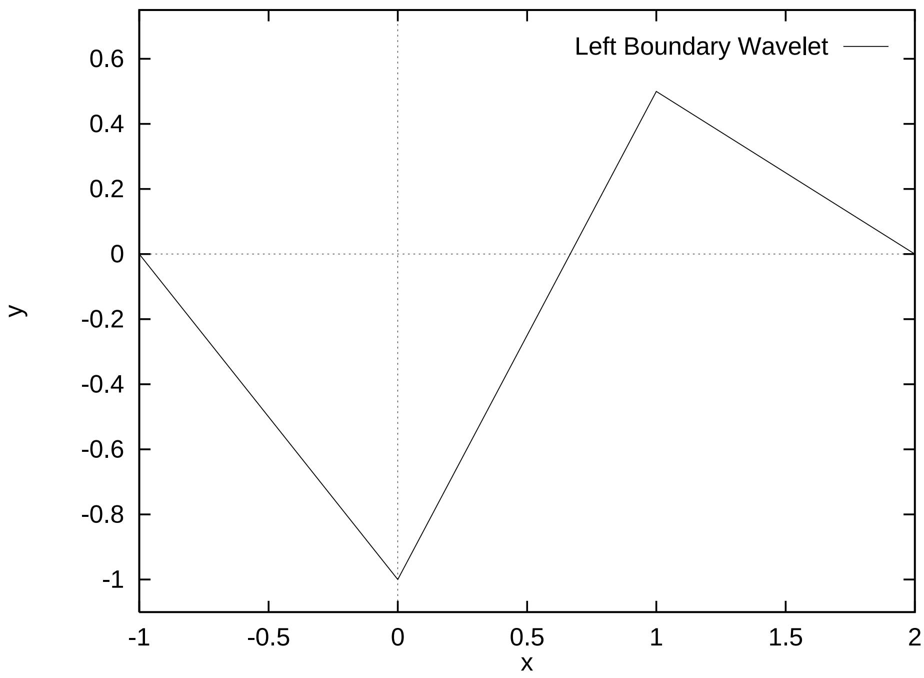


Figure 2: Boundary wavelet ψ_a.

shape function ψ together with its boundary modifications ψ_a and ψ_b by (cf. the Figures 1 and 2)

$$\psi(x) \;:=\; \begin{cases} \frac{3}{2}|x|-1 & \text{if } |x|\le 1 \\ 1-\frac{1}{2}|x| & \text{if } 1<|x|\le 2 \\ 0 & \text{else }, \end{cases} \tag{2.4}$$

$$\psi_a(x) \;:=\; \begin{cases} \psi(x) & \text{if } 0\le x \\ -1-x & \text{if } -1<x\le 0 \\ 0 & \text{else }, \end{cases} \tag{2.5}$$

$$\psi_b(x) \;:=\; \psi_a(-x). \tag{2.6}$$

Then a multilevel basis of $\mathrm{span}\{\varphi_j^{N_L},\ j=1,\dots N_L-2\}$ is given as follows. We define basis functions on each level $l=0,\dots,L$. For $l=0$, we easily take the hat functions $\{\varphi_i^{N_0}\ :\ i=1,2\}$ on the coarsest grid. For $l>0$, we take the dilated shape function $x\mapsto\psi([3\cdot 2^l]x)$ and shift it to each point of the difference grid $\{\frac{i}{N_l-1}\ :\ i=1,\dots,N_l-2\}\setminus\{\frac{i}{N_{l-1}-1}: i=1,\dots,N_{l-1}-2\}$. In other words, we set

$$\begin{aligned}
\psi_{0,i}(x) &:= \varphi_i^{N_0}(x),\ \ i=1,2=N_0-2, && (2.7)\\
\psi_{l,1}(x) &:= \psi_a\left([3\cdot 2^l]\cdot x-1\right)\sqrt{3\cdot 2^l},\ \ l=1,\dots L,\\
\psi_{l,j}(x) &:= \psi\left([3\cdot 2^l]\cdot x-(2j-1)\right)\sqrt{3\cdot 2^l},\ \ j=2,\dots,N_{l-1}-2,\ l=1,\dots L,\\
\psi_{l,N_{l-1}-1}(x) &:= \psi_b\left([3\cdot 2^l]\cdot x-(2[N_{l-1}-1]-1)\right)\sqrt{3\cdot 2^l},\ \ l=1,\dots L. && (2.8)
\end{aligned}$$

Now we set

$$\mathcal{N} \;:=\; \{(l,t,i,j):\ l=0,\ t=1,\ i,j=1,2, \tag{2.9}$$

$$\begin{aligned} & l=1,\dots,L,\ t=1,\ i=1,\dots,N_{l-1}-1,\ j=1,\dots,N_{l-1}-2, \\ & l=1,\dots,L,\ t=2,\ i=1,\dots,N_{l-1}-2,\ j=1,\dots,N_{l-1}-1, \\ & l=1,\dots,L,\ t=3,\ i,j=1,\dots,N_{l-1}-1\} \end{aligned}$$

and introduce the <u>two-dimensional wavelets</u> as follows. We define basis functions for each level $l \in \{0,1,\dots,L\}$. For $l=0$, we take the tensor product hat functions $(x,y) \mapsto \varphi_i^{N_0}(x)\varphi_j^{N_0}(y)$ with $i,j=1,2$ on the coarsest grid. Now let φ stand for the univariate hat function

$$\varphi(x) := \begin{cases} 1-\frac{|x|}{2} & \text{if } |x| \le 2 \\ 0 & \text{else .} \end{cases} \tag{2.10}$$

Then, instead of the univariate shape function ψ in the definition of univariate wavelets, we consider the three shape functions $(x,y) \mapsto \psi(x)\varphi(y)$, $(x,y) \mapsto \varphi(x)\psi(y)$, and $(x,y) \mapsto \psi(x)\psi(y)$. For each level $l>0$, we take the dilated shape functions and shift them to the points of the difference grid $\{(\frac{i}{N_l-1},\frac{j}{N_l-1}) : i,j=1,\dots,N_l-2\} \setminus \{(\frac{i}{N_{l-1}-1},\frac{j}{N_{l-1}-1}) : i,j=1,\dots,N_{l-1}-2\}$. More exactly, we shift $(x,y) \mapsto \psi([3\cdot 2^l]x)\ \varphi([3\cdot 2^l]y)$ to the points $\left\{\left(\frac{2i-1}{N_l-1},\frac{2j}{N_l-1}\right)\right\}$, the function $(x,y) \mapsto \varphi([3\cdot 2^l]x)\psi([3\cdot 2^l]y)$ to $\left\{\left(\frac{2i}{N_l-1},\frac{2j-1}{N_l-1}\right)\right\}$, and $(x,y) \mapsto \psi([3\cdot 2^l]x)\psi([3\cdot 2^l]y)$ to the points $\left\{\left(\frac{2i-1}{N_l-1},\frac{2j-1}{N_l-1}\right)\right\}$. In other words, we set (cf. the Figures 3 and 4)

$$\begin{aligned} \psi_{(0,1,i,j)} &:= \varphi_i^{N_0}(x)\varphi_j^{N_0}(y), && (2.11) \\ \psi_{(l,1,i,j)} &:= \psi_{l,i}(x)\varphi_j^{N_{l-1}}(y), && (2.12) \\ \psi_{(l,2,i,j)} &:= \varphi_i^{N_{l-1}}(x)\psi_{l,j}(y), && (2.13) \\ \psi_{(l,3,i,j)} &:= \psi_{l,i}(x)\psi_{l,j}(y). && (2.14) \end{aligned}$$

It is easy to see that the functions $\psi_{l,j}$, $l \ge 1$, $j=2,\dots,N_{l-1}-2$ are orthogonal to linear functions. Hence, the functions $\psi_I = \psi_{(l,t,i,j)}$ are orthogonal to bilinear functions if $l \ge 1$, $t=1$ and $i \ne 1, N_{l-1}-1$ or if $l \ge 1$, $t=2$ and $j \ne 1, N_{l-1}-1$ or if $l \ge 1$, $t=3$ and $\{i,j\} \not\subseteq \{1,N_{l-1}-1\}$. The functions $\{\psi_I,\ I \in \mathcal{N}\}$ belong to the class of biorthogonal wavelets in the sense of [5]. More exactly, the functions without the boundary modification including ψ_a,ψ_b belong to this class. However, the boundary modification at the end points 0 and 1 of the interval is well known, too. Namely, the functions from $\mathrm{span}\{\varphi_j^{N_L},\ j=1,\dots,N_L-2\}$ can be extended to odd functions over $[-1,1]$ and further to 2-periodic functions over the real axis $\mathbb{R}$. The boundary function $x \mapsto \psi_a(3\cdot 2^l\cdot x-1)$ is the difference of the basis functions $x \mapsto \psi(3\cdot 2^l\cdot x-1)$ and $x \mapsto \psi(3\cdot 2^l\cdot x+1)$ over $\mathbb{R}$ and, thus, the restriction of a natural basis function from the space of extended functions. This fact guarantees the boundedness of the wavelet basis transform also for the case of the interval. Note that the univariate function $\psi_{l,j}$ have been used also in [26, 29]. All properties of the functions $\{\psi_I,\ I \in \mathcal{N}\}$ mentioned in this subsection are well known from the theory of wavelets (cf. [13, 4, 5]).

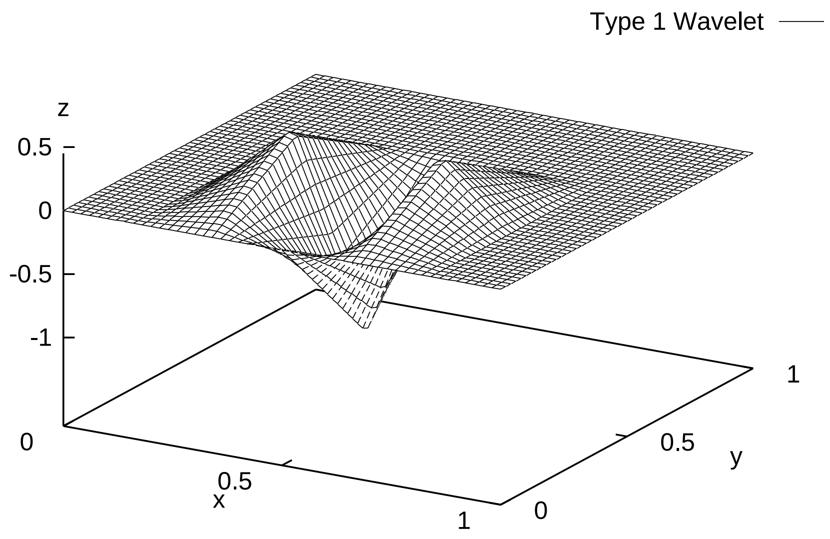


Figure 3: Wavelet $\psi_{(1,1,2,1)}$.

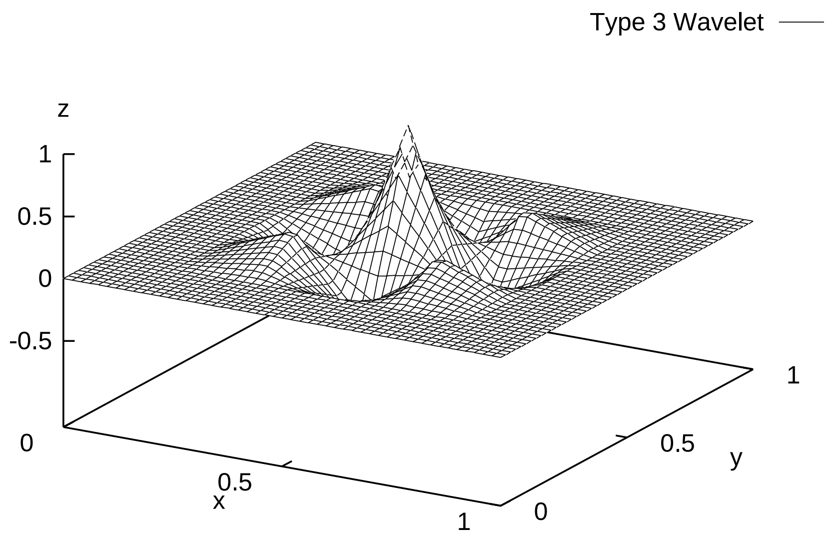


Figure 4: Wavelet $\psi_{(1,3,2,2)}$.

Lemma 2.1 *a) The wavelet transform E_h introduced in Subsection 2.1 and its inverse E_h^{-1} are bounded in the l^2 operator norm and the bounds are uniform with respect to L (recall that $h = \frac{1}{N-1}$, $N = 3 \cdot 2^L + 1$).*
b) The basis $\{\psi_I,\ I \in \mathcal{N}\}$ represents a hierarchical basis of $\mathcal{X}_h$, i.e., for $l = 1, \ldots, L$ one has

$$\begin{aligned} \operatorname{span}\left\{\varphi_{i,j}^{N_l}:\ i,j = 1,\ldots,N_l - 2\right\} &= \operatorname{span}\left\{\varphi_{i,j}^{N_{l-1}}:\ i,j = 1,\ldots,N_{l-1} - 2\right\} \\ &\bigoplus \operatorname{span}\left\{\psi_I:\ I = (l',t,i,j) \in \mathcal{N}, l' = l\right\}. \end{aligned} \tag{2.15}$$

This is a consequence of so called two-scale relations including the relations

$$\psi_{l,j} = \frac{1}{2}\varphi_{2j-2}^{N_l} - \varphi_{2j-1}^{N_l} + \frac{1}{2}\varphi_{2j}^{N_l}, \qquad j = 1,\ldots,N_{l-1} - 1, \tag{2.16}$$

$$\varphi_j^{N_{l-1}} = \sqrt{2}\left\{\frac{1}{2}\varphi_{2j-1}^{N_l} + \varphi_{2j}^{N_l} + \frac{1}{2}\varphi_{2j+1}^{N_l}\right\}, \quad j = 1,\ldots,N_{l-1} - 2, \tag{2.17}$$

where we have to set $\varphi_0^{N_l} := 0$ and $\varphi_{N_l-1}^{N_l} := 0$ in (2.16) for $j = 1$ or $j = N_{l-1} - 1$. As a consequence of Lemma 2.1, b) we obtain the following fast pyramid algorithm for the application of E_h. Suppose we are given the vector of coefficients $\{\xi_{(i,j)}^L,\ i,j = 1,\ldots N_L - 2\}$ of the function $u^L = \sum \xi_{(i,j)}^L \varphi_{(i,j)}^{N_L}$. In view of (2.15) with $l = L$, we split $u^L = u^{L-1} + v^L$ such that $u^{L-1} = \sum \xi_{(i,j)}^{L-1} \varphi_{(i,j)}^{N_{L-1}}$ and $v^L = \sum \eta_{(L,t,i,j)} \psi_{(L,t,i,j)}$. We have obtained the first wavelet coefficients $\eta_{(l,t,i,j)}$ for $l = L$. In the next step we use (2.15) with $l = L-1$ and split $u^{L-1} = u^{L-2} + v^{L-1}$ such that $u^{L-2} = \sum \xi_{(i,j)}^{L-2} \varphi_{(i,j)}^{N_{L-2}}$ and $v^{L-1} = \sum \eta_{(L-1,t,i,j)} \psi_{(L-1,t,i,j)}$. This gives us the values $\eta_{(l,t,i,j)}$ for $l = L-1$. Proceeding in the same fashion for $l = L-2,\ldots,1$, we finally get all the coefficients $\eta_{(l,t,i,j)}$ for $l = L,\ldots,1$ and $\eta_{(0,1,i,j)} = \xi_{(i,j)}^0$.

It remains to show that the splitting $u^l = u^{l-1} + v^l$ with $u^l = \sum \xi_{(i,j)}^l \varphi_{(i,j)}^{N_l}$, $u^{l-1} = \sum \xi_{(i,j)}^{l-1} \varphi_{(i,j)}^{N_{l-1}}$, and $v_l = \sum \eta_{(l,t,i,j)} \psi_{(l,t,i,j)}$ can be done by a fast algorithm. To this end, we observe that the bases $\{\varphi_{(i,j)}^{N_l}\}$, $\{\varphi_{(i,j)}^{N_{l-1}}\}$, and $\{\psi_{(l,t,i,j)}\}$ are tensor products of the one-dimensional bases $\{\varphi_i^{N_l}\}$, $\{\varphi_j^{N_{l-1}}\}$, and $\{\psi_{l,i}\}$, respectively. Consequently, we only need to apply one-dimensional splittings. Indeed, for every i, we split $\sum_j \xi_{(i,j)}^l \varphi_j^{N_l}(y)$ into $\sum_j \alpha_{(i,j)}^l \psi_{l,j}(y) + \sum_j \beta_{(i,j)}^l \varphi_j^{N_{l-1}}(y)$, i.e.,

$$\sum_{i,j} \xi_{(i,j)}^l \varphi_i^{N_l}(x)\varphi_j^{N_l}(y) = \sum_{i,j} \alpha_{(i,j)}^l \varphi_i^{N_l}(x)\psi_{l,j}(y) + \sum_{i,j} \beta_{(i,j)}^l \varphi_i^{N_l}(x)\varphi_j^{N_{l-1}}(y).$$

Then, for any j, we split $\sum_i \alpha_{(i,j)}^l \varphi_i^{N_l}(x)$ and $\sum_i \beta_{(i,j)}^l \varphi_i^{N_l}(x)$ into the sums $\sum_i \gamma_{(i,j)}^l \psi_{l,i}(x) + \sum_i \delta_{(i,j)}^l \varphi_i^{N_{l-1}}(x)$ and $\sum_i \epsilon_{(i,j)}^l \psi_{l,i}(x) + \sum_i \zeta_{(i,j)}^l \varphi_i^{N_{l-1}}(x)$, respectively. In other words

$$\begin{aligned} \sum_{i,j} \xi_{(i,j)}^l \varphi_i^{N_l}(x)\varphi_j^{N_l}(y) &= \sum_{i,j} \gamma_{(i,j)}^l \psi_{l,i}(x)\psi_{l,j}(y) + \sum_{i,j} \delta_{(i,j)}^l \varphi_i^{N_{l-1}}(x)\psi_{l,j}(y) \\ &+ \sum_{i,j} \epsilon_{(i,j)}^l \psi_{l,i}(x)\varphi_j^{N_{l-1}}(y) + \sum_{i,j} \zeta_{(i,j)}^l \varphi_i^{N_{l-1}}(x)\varphi_j^{N_{l-1}}(y). \end{aligned}$$

Setting $\xi_{(i,j)}^{l-1} = \zeta_{(i,j)}^l$, $\eta_{(l,1,i,j)} = \epsilon_{(i,j)}^l$, $\eta_{(l,2,i,j)} = \delta_{(i,j)}^l$, and $\eta_{(l,3,i,j)} = \gamma_{(i,j)}^l$, we arrive at the two-dimensional splitting.

For the one-dimensional splitting, we have to determine $\{\alpha_i\}$, $\{\beta_i\}$ from $\{\xi_i\}$ such that

$$\sum \xi_i \varphi_i^{N_l} = \sum \alpha_i \psi_{l,i} + \sum \beta_i \varphi_i^{N_{l-1}}. \tag{2.18}$$

If we substitute (2.16), (2.17) into the right-hand side of (2.18) and compare the coefficients of the $\varphi_i^{N_l}$, we arrive at a tridiagonal linear system for the unknowns α_i, β_i. This, however, can be solved with $O(N_l)$ arithmetic operations. The whole algorithm for the multiplication by E_h requires no more than $O(2^{2L}) = O((N_L - 2)^2)$ operations, where $(N_L - 2)^2$ is the dimension of the trial space.

Remark 2.2 *For the numerical tests, we even modify the boundary wavelets ψ_a and ψ_b. We set*

$$\psi_a(x) := \begin{cases} 1 - \frac{1}{2}x & \text{if } 1 < x < 2 \\ x - \frac{1}{2} & \text{if } 0 < x < 1 \\ -\frac{1}{2}(x+1) & \text{if } -1 < x < 0 \\ 0 & \text{else} \end{cases} \tag{2.19}$$

and $\psi_b(x) := \psi_a(-x)$. Though we do not know whether Lemma 2.1 a) holds for this modification, we have not observed any instability in the computations. The advantage of the modification is that we obtain the moment condition of order zero, i.e., the new boundary wavelets are orthogonal to constant functions.

2.3. The Wavelets for the Test Space

Retain the notation of $\mathcal{N}$ and N_l from the previous subsection and, in accordance with Section 1.2, set $P_{(i,j)}^{N_l} := (\frac{i}{N_l-1}, \frac{j}{N_l-1})$. We denote the functionals of function evaluation at $P_{(i,j)}^{N_l}$ by $\delta_{P_{(i,j)}^{N_l}}$, i.e., we set $\langle f, \delta_{P_{(i,j)}^{N_l}} \rangle = f(P_{(i,j)}^{N_l})$. Now we introduce the <u>wavelet functionals</u> as follows. We define the basis functionals for each level $l \in \{0, 1, \ldots, L\}$. If $l = 0$, then we choose the $\delta_{P_{(i,j)}^{N_0}}$ with $i, j = 1, 2$. For $l > 0$, we define the basis functionals as differences $\delta_P - \delta_Q$, where P is a point of the difference grid $\{P_{(i,j)}^{N_l} : i, j = 1, \ldots, N_l - 2\} \setminus \{P_{(i,j)}^{N_{l-1}} : i, j = 1, \ldots, N_{l-1} - 2\}$ and Q is a neighbor point of P on the grid $\{P_{(i,j)}^{N_l} : i, j = 1, \ldots, N_l - 2\}$. If $P = \left(\frac{2i-1}{N_l-1}, \frac{2j}{N_l-1}\right)$ with $2i < N_l - 1$ then we choose the right neighbor $Q := \left(\frac{2i}{N_l-1}, \frac{2j}{N_l-1}\right)$. For $P = \left(\frac{2i-1}{N_l-1}, \frac{2j}{N_l-1}\right)$ with $2i = N_l - 1$, we take the left neighbor $Q := \left(\frac{2i-2}{N_l-1}, \frac{2j}{N_l-1}\right)$ since $\left(\frac{2i}{N_l-1}, \frac{2j}{N_l-1}\right)$ is no more in the grid $\{P_{(i,j)}^{N_l} : i, j = 1, \ldots, N_l - 2\}$. Similarly, we choose the upper neighbor $Q := \left(\frac{i}{N_l-1}, \frac{2j}{N_l-1}\right)$ for $P = \left(\frac{i}{N_l-1}, \frac{2j-1}{N_l-1}\right)$ with $2j < N_l - 1$ as well as the lower neighbor $Q := \left(\frac{i}{N_l-1}, \frac{2j-2}{N_l-1}\right)$ for $P = \left(\frac{i}{N_l-1}, \frac{2j-1}{N_l-1}\right)$ with $2j = N_l - 1$. In other words, for $J = (l, t, i, j) \in \mathcal{N}$, we introduce the wavelet functionals

$$
\vartheta_J := [3\cdot 2^l]^{-1}\begin{cases}
\delta_{P^{N_0}_{(i,j)}} & \text{if } l=0,\ t=1,\ i,j=1,2\\
\delta_{P^{N_l}_{(2i-1,2j)}} - \delta_{P^{N_l}_{(2i,2j)}} & \text{if } l\ge 1,\ t=1,\ i,j=1,\dots,N_{l-1}-2\\
\delta_{P^{N_l}_{(2i-1,2j)}} - \delta_{P^{N_l}_{(2i-2,2j)}} & \text{if } l\ge 1,\ t=1,\\
 & i=N_{l-1}-1,\ j=1,\dots,N_{l-1}-2\\
\delta_{P^{N_l}_{(2i,2j-1)}} - \delta_{P^{N_l}_{(2i,2j)}} & \text{if } l\ge 1,\ t=2,\\
 & i,j=1,\dots,N_{l-1}-2\\
\delta_{P^{N_l}_{(2i,2j-1)}} - \delta_{P^{N_l}_{(2i,2j-2)}} & \text{if } l\ge 1,\ t=2,\\
 & i=1,\dots,N_{l-1}-2,\ j=N_{l-1}-1\\
\delta_{P^{N_l}_{(2i-1,2j-1)}} - \delta_{P^{N_l}_{(2i-1,2j)}} & \text{if } l\ge 1,\ t=3,\\
 & i=1,\dots,N_{l-1}-1,\ j=1,\dots,N_{l-1}-2\\
\delta_{P^{N_l}_{(2i-1,2j-1)}} - \delta_{P^{N_l}_{(2i-1,2j-2)}} & \text{if } l\ge 1,\ t=3,\\
 & i=1,\dots,N_{l-1}-1,\ j=N_{l-1}-1.
\end{cases} \tag{2.20}
$$

It is easy to see that $\text{span}\{\vartheta_J,\ J\in\mathcal{N}\} = \text{span}\{h\delta_{P^N_I},\ I\in\mathcal{M}\}$.

In order to analyse the wavelet transform R_h, we introduce the bidual wavelet system. First, for each level $l\in\{0,1,\dots,L\}$, we consider the uniform partition $(0,1]\times(0,1] = \cup_{i,j=1}^{N_l-1}(\frac{i-1}{N_l-1},\frac{i}{N_l-1}]\times(\frac{j-1}{N_l-1},\frac{j}{N_l-1}]$. The space of piecewise constant functions over this partition is $(N_l-1)^2$ dimensional. To get an $(N_l-2)^2$ dimensional space we enlarge the subdomains intersecting the right and the upper part of the boundary ∂S. Thus, we consider the partition

$$
\begin{aligned}
(0,1)\times(0,1) &= \cup_{i,j=1}^{N_l-2} S^{N_l}_{i,j}, && (2.21)\\
S^{N_l}_{i,j} &:= \begin{cases}
(\frac{i-1}{N_l-1},\frac{i}{N_l-1}]\times(\frac{j-1}{N_l-1},\frac{j}{N_l-1}] & \text{if } i,j=1,\dots,N_l-3\\
(\frac{i-1}{N_l-1},\frac{i+1}{N_l-1}]\times(\frac{j-1}{N_l-1},\frac{j}{N_l-1}] & \text{if } i=N_l-2,\ j=1,\dots,N_l-3\\
(\frac{i-1}{N_l-1},\frac{i}{N_l-1}]\times(\frac{j-1}{N_l-1},\frac{j+1}{N_l-1}] & \text{if } i=1,\dots,N_l-3,\ j=N_l-2\\
(\frac{i-1}{N_l-1},\frac{i+1}{N_l-1}]\times(\frac{j-1}{N_l-1},\frac{j+1}{N_l-1}] & \text{if } i=j=N_l-2.
\end{cases}
\end{aligned}
$$

The set of piecewise constant functions over this partition of level l is spanned by the functions $\chi^{N_l}_{(i,j)},\ i,j,=1,\dots,N_l-2$, where

$$
\chi^{N_l}_{(i,j)}(x) := \begin{cases} [3\cdot 2^l] & \text{if } x\in S^{N_l}_{i,j}\\ 0 & \text{else }. \end{cases} \tag{2.22}
$$

Now we introduce a simple hierarchical basis for the space $\mathcal{Y}_h := \text{span}\{\chi^{N_L}_I : I\in\mathcal{M}\}$ of piecewise constant functions over the finest level. For the coarsest level $l=0$, we take the $\chi^{N_0}_{(i,j)}$ with $i,j=1,2$. For $l>0$, we observe that the piecewise constant functions of level l over the square $S^{N_{l-1}}_{i,j}$ are spanned by the level $(l-1)$ function $\chi^{N_{l-1}}_{i,j}$ and the three level l functions $\chi^{N_l}_{2i-1,2j}+\chi^{N_l}_{2i-1,2j-1}$, $\chi^{N_l}_{2i,2j-1}$, and $\chi^{N_l}_{2i-1,2j-1}$. Thus, for each i,j, we take these three level l functions as basis functions. In other words, the hierarchical basis in $\mathcal{Y}_h$ is given by

$$
\omega_{(0,1,i,j)} := \chi^{N_0}_{(i,j)},\ i,j=1,2 \tag{2.23}
$$

$$\begin{aligned}
\omega_{(l,1,i,j)} &:= \chi^{N_l}_{(2i-1,2j)} + \chi^{N_l}_{(2i-1,2j-1)},\ i=1,\ldots,N_{l-1}-1,\ j=1,\ldots,N_{l-1}-2 \\
\omega_{(l,2,i,j)} &:= \chi^{N_l}_{(2i,2j-1)},\ i=1,\ldots,N_{l-1}-2,\ j=1,\ldots,N_{l-1}-1 \\
\omega_{(l,3,i,j)} &:= \chi^{N_l}_{(2i-1,2j-1)},\ i,j=1,\ldots,N_{l-1}-1, \\
& \quad\ l=1,\ldots,L.
\end{aligned}$$

For this basis it is not hard to verify that, for any $I, J \in \mathcal{N}$, there holds

$$\langle \omega_I, \vartheta_J \rangle = \delta_{I,J} \tag{2.24}$$

and the projection P_h onto the space $\mathcal{Y}_h$ interpolating at the points $\{P_I,\ I \in \mathcal{M}\}$ can be represented as

$$P_h f = \sum_{I\in\mathcal{M}} h f(P_I)\chi_I^{N_L} = \sum_{J\in\mathcal{N}} \langle f, \vartheta_J\rangle \omega_J. \tag{2.25}$$

Now the matrix R_h of Subsection 2.1 is nothing else than the basis transform from $\{\omega_J\}$ to $\{\chi_I^{N_L}\}$, i.e., $R_h\nu = \mu$ if $\sum_{J\in\mathcal{N}} \nu_J\omega_J = \sum_{I\in\mathcal{M}} \mu_I\chi_I^{N_L}$. In view of the uniform norm equivalence

$$\|\sum_{I\in\mathcal{M}} \mu_I\chi_I^{N_L}\|_{L^2(S)} \sim \sqrt{\sum_{I\in\mathcal{M}} |\mu_I|^2}, \tag{2.26}$$

it is natural to supply the image space of R_h with the l^2 norm. Unfortunately, the analogous norm equivalence for the system $\{\omega_J\}$ is not valid. We get (compare [28])

Lemma 2.2 *a) For the basis transform $R_h \in \mathcal{L}(l^2(\mathcal{N}), l^2(\mathcal{M}))$, there exists a constant C independent of L (recall that $h = \frac{1}{N-1}$, $N = 3\cdot 2^L + 1$) such that*

$$C^{-1}\sqrt{L} \le \|R_h\| \le C\sqrt{L}, \tag{2.27}$$

$$C^{-1}2^L \le \|R_h^{-1}\| \le C 2^L. \tag{2.28}$$

b) The basis $\{\omega_J,\ J\in\mathcal{N}\}$ represents a hierarchical basis of $\mathcal{Y}_h$, i.e., for $l=1,\ldots,L$, we get

$$\begin{aligned}
\operatorname{span}\left\{\chi^{N_l}_{i,j}:\ i,j=1,\ldots,N_l-2\right\} &= \operatorname{span}\left\{\chi^{N_{l-1}}_{i,j}:\ i,j=1,\ldots,N_{l-1}-2\right\} \\
&\quad \bigoplus \operatorname{span}\left\{\omega_I:\ I=(l',t,i,j)\in\mathcal{N}, l'=l\right\}.
\end{aligned} \tag{2.29}$$

This fact is a consequence of the two-scale relations which include the relations (2.23) and

$$\chi^{N_{l-1}}_{(i,j)} = 2\left\{\chi^{N_l}_{(2i-1,2j-1)} + \chi^{N_l}_{(2i-1,2j)} + \chi^{N_l}_{(2i,2j-1)} + \chi^{N_l}_{(2i,2j)}\right\}. \tag{2.30}$$

Similarly to E_h, the basis transform R_h can be performed with the help of a fast pyramid scheme. Indeed, suppose we are given the vector of wavelet coefficients $\nu = \{\nu_J\}_{J\in\mathcal{N}}$ of a function $v^L = \sum_{J\in\mathcal{N}} \nu_J\omega_J$. In view of $\omega_{(0,1,i,j)} = \chi^{N_0}_{(i,j)}$ and the relation (2.29) with $l=1$, we compute the coefficients $\mu^1_{(i,j)}$ of the representation $v^1 = \sum_{i,j=1,\ldots,N_1-2} \mu^1_{(i,j)}\chi^{N_1}_{(i,j)}$ for the function $v^1 := \sum_{l=0}^1 \sum_{t,i,j} \nu_{(l,t,i,j)}\omega_{(l,t,i,j)}$. Next, in view of (2.29) with $l=2$, we compute the coefficients $\mu^2_{(i,j)}$ of the representation $v^2 = \sum_{i,j=1,\dots,N_2-2} \mu^2_{(i,j)} \chi^{N_2}_{(i,j)}$ for the function $v^2 := \sum_{i,j=1,\dots,N_1-2} \mu^1_{(i,j)} \chi^{N_1}_{(i,j)} + \sum_{t,i,j} \nu_{(2,t,i,j)} \omega_{(2,t,i,j)}$. Proceeding in this manner, we finally determine the coefficients $\mu_{(i,j)} = \mu^L_{(i,j)}$ of the representation $\sum_{i,j=1,\dots,N_L-2} \mu^L_{(i,j)} \chi^{N_L}_{(i,j)}$ for the function $v^L := \sum_{i,j=1,\dots,N_{L-1}-2} \mu^{L-1}_{(i,j)} \chi^{N_{L-1}}_{(i,j)} + \sum_{t,i,j} \nu_{(L,t,i,j)} \omega_{(L,t,i,j)}$. Remark that, for each level l, the computation of the $\mu^l_{(i,j)}$ from the $\mu^{l-1}_{(i,j)}$ and the $\nu_{(l,t,i,j)}$ can be realised by substituting the two scale relations (2.23) and (2.30) into $\sum_{i,j} \mu^l_{(i,j)} \chi^{N_l}_{(i,j)} = \sum_{i,j} \mu^{l-1}_{(i,j)} \chi^{N_{l-1}}_{(i,j)} + \sum_{t,i,j} \nu_{(l,t,i,j)} \omega_{(l,t,i,j)}$ and by comparing the coefficients of the functions $\chi^{N_l}_{(i,j)}$. In other words, on each level l, the vectors $\{\mu^{l-1}_{(i,j)}\}_{(i,j)}$ and $\{\nu_{(l,t,i,j)}\}_{(t,i,j)}$ are to be multiplied by a matrix containing only a small number of non-zero entries in each row. By this way, the number of all arithmetic operations for the multiplication by R_h is less than $O(2^{2L}) = O((N_L-2)^2)$, where $(N_L-2)^2$ is the dimension of the test space.

2.4. The Compression Scheme

In this section we describe the compression algorithm. The results and proofs are completely analogous to those given by Dahmen, Prößdorf, Schneider, v.Petersdorff, and Schwab [11, 25] (cf. also [28] for the treatment of wavelets on the boundary of S). Hence, we present only the results without proofs. We begin with an estimate for the entries of B_h.

Lemma 2.3 *Suppose the parametrisation F of the surface Γ is smooth. Then, for $J = (l_J, t_J, i_J, j_J)$ and $I = (l_I, t_I, i_I, j_I)$ from $\mathcal{N}$ such that ψ_I is orthogonal to bilinear functions, the entry $b_{J,I} := \langle A\psi_I, \vartheta_J \rangle$ of the wavelet transform B_h can be estimated as*

$$|b_{J,I}| \le 2^{-3l_I - 2l_J} \left[\mathrm{dist}\,(\mathrm{supp}\,\psi_I, \mathrm{supp}\,\vartheta_J)\right]^{-5}, \tag{2.31}$$

where $\mathrm{supp}\,\psi_I$ and $\mathrm{supp}\,\vartheta_J$ denote the supports of the function ψ_I and the functional ϑ_J, respectively. By $\mathrm{dist}(\mathrm{supp}\,\psi_I, \mathrm{supp}\,\vartheta_J)$ we have denoted the distance between the sets $\mathrm{supp}\,\psi_I$ and $\mathrm{supp}\,\vartheta_J$.

Estimate (2.31) tells us where the small entries are. Neglecting these entries, we arrive at the following compressed matrix C_h.

Theorem 2.1 *Suppose that the parametrisation F of the surface Γ is smooth and that the solution u and the right-hand side v of (1.5) belong to the Sobolev space H^2 and vanish on the boundary of the domain S. Choose β, γ, and α such that $\frac{1}{3} \le \beta \le 1$, $1 \le \gamma < \frac{4}{3}$, and $\alpha = \frac{1}{3} + \beta$. Let the compressed matrix $C_h = (c_{J,I})_{J,I\in\mathcal{N}}$ be defined by*

$$c_{J,I} := \begin{cases} b_{J,I} & \textit{if}\ \mathrm{dist}\,(\mathrm{supp}\,\psi_I, \mathrm{supp}\,\vartheta_J) \le \max\{2^{-l_I},\, 2^{-l_J},\, (a2^{\alpha L}) 2^{-\beta l_J - \gamma l_I}\} \\ 0 & \textit{else} \end{cases} \tag{2.32}$$

with a suitable constant a. If a is large enough, then there is an $h_0 > 0$ such that, for any $h = \frac{1}{N_L - 1} < h_0$, the operator $\tilde{A}_h := [R_h C_h E_h] \in \mathcal{L}(\mathcal{X}_h)$ is invertible and its inverse $\tilde{A}_h^{-1}$ is uniformly bounded. Additionally, if u_h denotes the solution of $\tilde{A}_h u_h = v_h$ with $v_h = \sum_{I\in\mathcal{M}} h v_h(P_I) \varphi_I$, then

$$\|u - u_h\|_{L^2} \le C a^{-3} h^2 \begin{cases} \{\log\, h^{-1}\}^{3/2} & \textit{if}\ \beta = \frac{1}{3} \\ \sqrt{\log\, h^{-1}} & \textit{else} \end{cases} \tag{2.33}$$

and the number of non-zero entries in the $(N_L-2)^2 \times (N_L-2)^2$ *matrix* C_h *is less than* $Ca^2[(N_L-2)^2]^{4/3}[\log(N_L-2)]^b = Ca^2[h^{-2}]^{4/3}[\log\, h^{-1}]^b$*, where*

$$b := \begin{cases} 2 & \text{if } \beta = \gamma = 1 \\ 1 & \text{if } \beta = 1 \neq \gamma \text{ or if } \gamma = 1 \neq \beta \\ 0 & \text{else} . \end{cases} \tag{2.34}$$

In other words, if the matrix C_h *is given and if the equation* $\tilde{A}_h u_h = v_h$ *is solved by an optimal iterative algorithm, then an approximate solution with an error less than* $Ch^2\sqrt{\log\, h^{-1}}$ *can be computed with no more than* $C[h^{-2}]^{4/3}[\log\, h^{-1}]^b$ *arithmetic operations.*

Remark 2.3 *An optimal compression* C_h *of* B_h *to a number of non-zero entries less than* $C(N_L-2)^2$ *like for the Galerkin scheme in [32] (cf. also the almost optimal compression in [11, 26]) seems to be impossible since the moment conditions for the wavelets are not strong enough. However, we have chosen the test wavelets not to achieve optimal asymptotic orders of complexity but to develop a fast algorithm for the case* $(N_L-2)^2 \leq 10\,000$*. In certain parts (cf. Algorithm 3 in Subsection 4.1) of the matrix assembling the number of points in the test functionals appears as a factor for the computing time. This is the reason why we have chosen the particular wavelets presented in Subsections 2.2 and 2.3.*

Remark 2.4 *From the Lemmata 2.2 and 2.1 we get* $\|C_h\| = \|R_h^{-1}\tilde{A}_h E_h^{-1}\| \sim 2^L$ *and* $\|R_h\| \sim L$*. Thus, the multiplication of a certain vector* $\tilde{\xi}_h$ *by* $R_h C_h E_h$ *can lead to an additional error of* $C2^L L$ *times the numerical error of* $\tilde{\xi}_h$*. This fact, however, seems not to disturb the numerical behaviour of the wavelet algorithm.*

In the numerical computations we have tested several choices of β and γ. Choosing $\beta = \frac{1}{3}$ and $\gamma = \frac{4}{3}$ has turned out to be a little bit worse than the others. However, the compression rates (compression rate = number of all entries divided by the number of non-zero entries in C_h) have not differed too much. The numerical results presented in Section 4 are obtained with $\beta = 1$ and $\gamma = 1$. The constant a has been determined by tests such that the additional error caused by the compression is less than a prescribed error. We will call the just defined compression (2.32) an a priori compression since the pattern of the compressed matrix is given a priori.

Of course, this scheme (2.32) leads to $O((N_L-2)^4)$ arithmetic operations for the assembling of the $(N_L-2)^2 \times (N_L-2)^2$ matrix C_h if we check for each entry whether it is to be computed or not. For a fast algorithm, we will use the tree structure of the hierarchical wavelet basis. Namely, to any wavelet ψ_I with $I = (l,t,i,j)$ on level l with $1 \leq l \leq L$, there correspond four sons ψ_{I_1}, ψ_{I_2}, ψ_{I_3}, and ψ_{I_4} on level $l+1$, where $I_1 = (l+1,t,2i-1,2j-1)$, $I_2 = (l+1,t,2i,2j-1)$, $I_3 = (l+1,t,2i-1,2j)$, and $I_4 = (l+1,t,2i,2j)$. We call ψ_I the father of ψ_{I_k}, $k = 1,\ldots,4$. For the scheme (2.32), we observe that, if $b_{J,I}$ is neglected, then also the entries b_{J,I_k}, $k = 1,\ldots,4$ are neglected, i.e., $c_{J,I} = 0$ implies $c_{J,I_k} = 0$, $k = 1,\ldots,4$. Consequently, if an entry is neglected, then all the entries corresponding to the sons, the sons of the sons and etc. can be neglected without any test whether the compression criterion is fulfilled or not.

3 The Quadrature Algorithm for the Computation of the Stiffness Matrix

In the present section, we describe the quadrature rules which we have used for the numerical tests. A different quadrature algorithm has been introduced and analysed by Schneider, v.Petersdorff, and Schwab [32, 25]. This algorithm for the Galerkin method should apply also to collocation. However, it relies on Gauß rules with variable degrees and requires the analyticity of the kernel and the underlying geometry. The quadrature rules we have implemented in the numerical tests of the present paper (cf. also the similar quadrature algorithm for one-dimensional double layer equations in [29]) are easier than those of [32, 25]. Instead of composite Gauß rules of variable order we apply composite two point rules.

3.1. Quadrature Rules for the Collocation without Wavelets

We have to define quadrature rules for the integral

$$Int := \int_S K(P,Q) u_h(Q) d_Q S, \tag{3.1}$$

where P is a collocation point, u_h is a bilinear function of $\mathcal{X}_h$, and the kernel (cf. (1.5))

$$K(P,Q) := \frac{1}{4\pi} \frac{\cos[\tau(P), F(Q) - F(P)]}{|F(Q) - F(P)|^2} F'(Q) \tag{3.2}$$

is singular for $Q = P$. We denote the Fréchet derivative of the parametrisation F by DF and introduce the <u>main part</u> $H(P,Q)$ of the kernel $K(P,Q)$ by

$$H(P,Q) := \frac{1}{4\pi} \frac{\cos[\tau(P), DF(P) \cdot (Q - P)]}{|DF(P) \cdot (Q - P)|^2} F'(P). \tag{3.3}$$

Substituting $K = H + (K - H)$ into (3.1) and applying <u>singularity subtraction</u> to the integral with the main part H, we arrive at

$$\begin{aligned} Int &= \int_S H(P,Q) d_Q S u_h(P) + \int_S H(P,Q)[u_h(Q) - u_h(P)] d_Q S \\ &\quad + \int_S [K(P,Q) - H(P,Q)] u_h(Q) d_Q S. \end{aligned} \tag{3.4}$$

In the last two integrals we employ a quadrature rule of the form

$$\int_S f(Q) d_Q S \sim \sum_{I \in \mathcal{O}} f(Q_I) w_I \tag{3.5}$$

to get

$$\begin{aligned} Int &\sim \int_S H(P,Q) d_Q S u_h(P) + \sum_{K \in \mathcal{O}} H(P,Q_K)[u_h(Q_K) - u_h(P)] w_K \\ &\quad + \sum_{K \in \mathcal{O}} [K(P,Q_K) - H(P,Q_K)] u_h(Q_K) w_K, \end{aligned} \tag{3.6}$$

$$Int \sim \left\{ \int_S H(P,Q) d_Q S - \sum_{K \in \mathcal{O}} H(P,Q_K) w_K \right\} u_h(P) + \sum_{K \in \mathcal{O}} K(P,Q_K) u_h(Q_K) w_K, \tag{3.7}$$

where the integral $\int_S H(P,Q)d_Q S$ can be computed with the help of the formula

$$\int_{a-1}^{a}\int_{b-1}^{b}\frac{cx+dy}{\{ex^2+fxy+gy^2\}^{3/2}}dydx = \tag{3.8}$$
$$-2\frac{2gc-fd}{\sqrt{g}[4eg+f^2]}\log\left\{\frac{[2gb+fa+2\sqrt{g}\sqrt{ea^2+fab+gb^2}](1-a)^2}{[2gb-f(1-a)+2\sqrt{g}\sqrt{e(1-a)^2-f(1-a)b+gb^2}]a^2}\right.$$
$$\left.\cdot\frac{2g(1-b)-fa+2\sqrt{g}\sqrt{ea^2-fa(1-b)+g(1-b)^2}}{2g(1-b)+f(1-a)+2\sqrt{g}\sqrt{e(1-a)^2+f(1-a)(1-b)+g(1-b)^2}}\right\}$$
$$-2\frac{2ed-fc}{\sqrt{e}[4eg+f^2]}\log\left\{\frac{[2ea+fb+2\sqrt{e}\sqrt{ea^2+fab+gb^2}](1-b)^2}{[2ea-f(1-b)+2\sqrt{e}\sqrt{ea^2-fa(1-b)+g(1-b)^2}]b^2}\right.$$
$$\left.\cdot\frac{2e(1-a)-fb+2\sqrt{e}\sqrt{e(1-a)^2-fb(1-a)+gb^2}}{2e(1-a)+f(1-b)+2\sqrt{e}\sqrt{e(1-a)^2+f(1-a)(1-b)+g(1-b)^2}}\right\}$$

The integrands $Q\mapsto H(P,Q)[u_h(Q)-u_h(P)]$ and $Q\mapsto[K(P,Q)-H(P,Q)]u_h(Q)$ are weakly singular at the point $Q=P$. Note that

$$\begin{aligned}|H(P,Q)[u_h(Q)-u_h(P)]| &\le C|Q-P|^{-1},\\ |[K(P,Q)-H(P,Q)]u_h(Q)| &\le C|Q-P|^{-1}.\end{aligned}\tag{3.9}$$

Therefore, we have to choose a rule (3.5) which depends on the singularity point P and is accurate for integrals of weakly singular functions.

To set up (3.5), we will give a rectangular partition of S which is adapted to the singular behaviour of the integrand and to the uniform grid which has been used for the definition of the trial space. From this partition, we obtain (3.5) by taking the tensor product of the two point Gauß rule over each subdomain. Thus, to define the underlying partition in dependence on the collocation point, we fix $P=(x_P,y_P)=(i_Ph,j_Ph)$. We consider the uniform partition

$$\begin{aligned}S &\subseteq \bigcup_{i=-i_P}^{N_L-2-i_P}\ \bigcup_{j=-j_P}^{N_L-2-j_P} S_{i,j},\\ S_{i,j} &:= \{(x_Q,y_Q): ih<x_Q-x_P<(i+1)h,\ jh<y_Q-y_P<(j+1)h\}\end{aligned}\tag{3.10}$$

which is the underlying partition for the definition of the trial space. Since the derivatives of the trial functions have jump discontinuities along the lines of this uniform rectangular partition, the quadrature partition should be a refinement of this grid. For $S_{i,j}$ with $P\notin S_{i,j}$ a further subdivision of $S_{i,j}$ is even unnecessary. Thus the rule (3.5) over the $S_{i,j}$ with $P\notin S_{i,j}$ is simply the tensor product of the two point Gauß rule taken over $S_{i,j}$.

Now we consider the case $P\in S_{i,j}$. Without loss of generality we suppose $S_{i,j}=S_{0,0}$. In the subdomain $S':=S_{0,0}\cap\{(x,y): x_P\le x,\ 0\le y-y_P\le x-x_P\}$ the first term of the asymptotics at P for the singular integrands in (3.4) takes the form

$$\Phi\left(\frac{y_Q-y_P}{x_Q-x_P}\right)\frac{1}{x_Q-x_P}. \tag{3.11}$$

Here $Q = (x_Q, y_Q) \in S'$ and Φ stands for a continuous function. For a singular function f of this kind, we apply Duffy's transformation to get

$$\begin{aligned}
&\int_{x_P}^{x_P+h} \int_{y_P}^{y_P+h} f(x,y)dydx \\
= &\int_0^h \int_0^x f(x_P+x, y_P+y)dydx + \int_0^h \int_0^y f(x_P+x, y_P+y)dxdy \\
= &\int_0^h \int_0^1 f(x_P+x, y_P+zx)xdzdx + \int_0^h \int_0^1 f(x_P+zy, y_P+y)ydzdy \\
= &\sum_{i=1}^4 \int_0^h \int_{(i-1)/4}^{i/4} f(x_P+x, y_P+zx)xdzdx + \sum_{i=1}^4 \int_0^h \int_{(i-1)/4}^{i/4} f(x_P+zy, y_P+y)ydzdy.
\end{aligned} \tag{3.12}$$

Note that, for the function f equal to (3.11), the integrand $f(x_P + x, y_P + zx)x$ is equal to $\Phi(z)$ and smooth. The quadrature rule (3.5) over the subdomain $S_{0,0}$ is now defined as the tensor product two point Gauß rule applied to each of the integrals in (3.12).

Remark 3.1 *This quadrature rule (3.5) is the result of a numerical test. We have started with a complicated rule over a graded mesh including variable Gauß orders. However, without any essential effect on the error of the collocation method, we could reduce the rule to the simple quadrature presented here. To get a quadrature which is theoretically optimal, we have to enlarge the orders of the Gauß quadrature rule in the neighbourhood of the singularity point slightly. Then the asymptotic quadrature error can be shown to be less than the discretisation error of the collocation method.*

3.2. Quadrature Rules for the Wavelet Algorithm

The quadrature rule for the wavelet algorithm is very similar to that of Subsection 3.1. Now the quadrature rules are to be applied to the last two integrals in (3.4), where $u_h \in \mathcal{X}_h$ is spanned by some but not all of the wavelet basis functions ψ_I. Since we will use a composite rule of tensor product two point Gauß rules, we only have to define the partition.

Consider a fixed collocation point P. This point belongs to the support of at most $2L+3$ test functionals ϑ_J. Let us denote them by $\vartheta_{J_1}, \vartheta_{J_2}, \ldots, \vartheta_{J_m}$. In view of the compression, we have to compute (3.4) for u_h in the span $\mathcal{Z}_h$ of all ψ_I, $I \in \mathcal{N}$ such that there exists a non-zero entry $c_{J_i,I}$, $1 \le i \le m$ of C_h. Thus, analogously to the previous subsection where the quadrature partition is a refinement of (3.10), the underlying partition of the quadrature should now be a refinement of the coarsest partition $Part_L$ for which the functions of $\mathcal{Z}_h$ are smooth on each subdomain. More precisely, the partition $Part_L$ is defined recursively in the following way: For $L = 1$, we set $Part_1 = \{T_{i,j} : i,j = 1,\ldots,N_1-1\}$ with $T_{i,j} := \{(x,y) : \frac{i-1}{N_1-1} < x < \frac{i}{N_1-1}, \frac{j-1}{N_1-1} < y < \frac{j}{N_1-1}\}$. If $Part_{l-1}$ is given, then $Part_l$ is the refinement of $Part_{l-1}$, where only those subdomains are divided into four equal squares for which there exists a non-differentiable function in $\mathcal{Z}_h$ or which are adjacent to P.

To get the refined partition of the quadrature, we leave the squares of $Part_L$ not containing the singularity point P unchanged and divide the squares containing P by the same manner as in Subsection 3.1. Finally, we define the composite quadrature rule over this partition by taking the tensor product two point Gauß rule over each subdomain. Since we have started from the partition $Part_L$ which is much coarser than the uniform partition (3.10), the number of quadrature knots will be smaller than that for the rule in Subsection 3.1. This is the essential point which leads to a reduction in computing time for the assembling of the stiffness matrix.

Remark 3.2 *The numbers of arithmetic operations to compute the compressed stiffness matrix C_h ($\sim$ sum of number of quadrature points for all collocation points P) by this quadrature algorithm is $C[h^{-2}]^{4/3}[\log h^{-1}]^b$, i.e., is asymptotically equal to the number of non-zero entries (cf. Theorem 2.1). In the numerical tests we have compared this algorithm with the discretised collocation algorithm without wavelets. These tests have revealed that the wavelet quadrature algorithm is fast and sufficiently accurate. The theoretical proof of the accuracy, however, seems to be hard. Therefore, we have introduced a modification in [28] such that the quadrature partition depends not only on the collocation point but also on the test functional including a point evaluation at the collocation point. For this quadrature algorithm we have proved that the asymptotic quadrature error is of the same order as the discretisation error. Moreover, numerical tests have shown that the modified quadrature algorithm requires only slightly more computing time than the one presented in this section.*

4 The Numerical Tests

4.1. The Algorithm for the Assembling of the Stiffness Matrix

Since the time for the set up of the matrix is much longer than that for the iterative solution of the matrix equation, we consider the assembling of the matrix in more detail. We will compare the algorithm without wavelets with the wavelet algorithm. However, instead of the scaled stiffness matrix $A_h = (hA\varphi_J(P_I))_{I,J\in\mathcal{M}}$ we compute $A_h^u := (A\varphi_J(P_I))_{I,J\in\mathcal{M}}$, and instead of the compressed matrix C_h of the scaled wavelet transform $B_h = (\langle\vartheta_I, A\psi_J\rangle)_{I,J\in\mathcal{N}}$ we set up the compressed matrix C_h^u of the matrix $B_h^u := (\langle[3\cdot 2^{l_I}]\vartheta_I, A\psi_J\rangle)_{I,J\in\mathcal{N}}$. In other words, we have to compute the entries $a_{I,J}^u \approx A\varphi_J(P_I)$ with $I,J\in\mathcal{M}$ of the collocation matrix without wavelets by (cf. (3.7))

$$\begin{aligned} a_{I,J}^u \;=\; & \sum_{K\in\mathcal{O}} K(P_I,Q_K)\varphi_J(Q_K)w_K + \\ & \left[-\frac{1}{2}\cos[n(P_I),\tau(P_I)] + \int_S H(P_I,Q)d_QS - \sum_{K\in\mathcal{O}} H(P_I,Q_K)w_K\right]\delta_{I,J} \end{aligned} \tag{4.1}$$

and the non-zero entries $c_{I,J}^u \approx \langle[3\cdot 2^{l_I}]\vartheta_I, A\psi_J\rangle$, $I,J\in\mathcal{N}$ of the wavelet algorithm by

$$\begin{aligned} c_{I,J}^u \;=\; & \sum_{K\in\mathcal{O}} K(P_I^1,Q_K)\psi_J(Q_K)w_K - \varrho\sum_{K\in\mathcal{O}} K(P_I^2,Q_K)\psi_J(Q_K)w_K \\ & + \left[-\frac{1}{2}\cos[n(P_I^1),\tau(P_I^1)] + \int_S H(P_I^1,Q)d_QS - \sum_{K\in\mathcal{O}} H(P_I^1,Q_K)w_K\right]\psi_J(P_I^1) \\ & -\varrho\left[-\frac{1}{2}\cos[n(P_I^2),\tau(P_I^2)] + \int_S H(P_I^2,Q)d_QS - \sum_{K\in\mathcal{O}} H(P_I^2,Q_K)w_K\right]\psi_J(P_I^2), \end{aligned} \tag{4.2}$$

where P_I^1, P_I^2 are defined such that $\langle [3 \cdot 2^{l_I}]\vartheta_I, f\rangle = f(P_I^1) - \varrho f(P_I^2)$ with $\varrho = 0$ for $I = (0,1,i,j)$ and with $\varrho = 1$ else (cf. (2.20)).

The most time consuming elements of the algorithm are the evaluations of the kernel function which include several Overhauser interpolations and transitions from spherical coordinates to the parameter coordinates over the square S. One computation of a kernel function requires more than twenty times the time of a single evaluation of a wavelet function. Therefore, we have to arrange the do loops in such an order that the number of kernel evaluations is minimised. For the algorithm without wavelets, this leads to the scheme of Algorithm 1. Proceeding analogously, the wavelet algorithm looks as in Algorithm 2. The essential parts in the Algorithms 1 and 2 are the do loops over the quadrature knots Q_k. Comparing these, we observe that the wavelet algorithm contains an additional do loop over the functionals $\vartheta_{\tilde{I}}$ with $P_I \in \operatorname{supp} \vartheta_{\tilde{I}}$. In view of this loop, we have chosen the test wavelets $\vartheta_{\tilde{I}}$ as linear combinations of only two Dirac delta functionals. Hence, for this choice, the loop over the $\vartheta_{\tilde{I}}$ is in the average over two items. On the other hand, to the do loop over two functions φ_J with $\varphi_J(Q_k) \neq 0$ in Algorithm 1, there corresponds a do loop over more than two functions ψ_J with $\psi_J(Q_k) \neq 0$ in Algorithm 2. The greater the number of vanishing moments, the larger is the support of the wavelets and the larger is the number of ψ_J not vanishing at Q_k. Though the higher compression for wavelets with more vanishing moments reduces the number of ψ_J with $\psi_J(Q_k) \neq 0$ and $J \in \mathcal{N}_U(P_I)$, we believe that the number of vanishing moments should not be too large, and we have chosen two vanishing moments in Subsection 2.2. Nevertheless, the additional do loop over the $\vartheta_{\tilde{I}}$ and the

do for any collocation point P_I, $I \in \mathcal{M}$
    do for any φ_J with $J \in \mathcal{M}$
        Set the initial value for the matrix entry $a^u_{I,J}$ equal to zero.
    end do
    Compute $d_I := -\frac{1}{2}\cos[n(P_I), \tau(P_I)] + \int_S H(P_I, Q) d_Q S$ by (3.8).
    Determine the knots Q_K and the weights w_K, $K \in \mathcal{O}$ of the quadrature
    rule (3.5) in dependence on P_I.
* *
    do for any quadrature knot Q_K with $K \in \mathcal{O}$
        Compute $[H(P_I, Q_K) w_K]$ and $[K(P_I, Q_K) w_K]$.
        Subtract $[H(P_I, Q_K) w_K]$ from d_I.
        Determine the set $\mathcal{M}(Q_K)$ of all indices $J \in \mathcal{M}$ with $\varphi_J(Q_K) \neq 0$.
        do for any φ_J with $J \in \mathcal{M}(Q_K)$
            Compute $\varphi_J(Q_K)$ and $[K(P_I, Q_K) w_K] \varphi_J(Q_K)$.
            Add $[K(P_I, Q_K) w_K] \varphi_J(Q_K)$ to $a^u_{I,J}$.
        end do
    end do
* *
    Add d_I to $a^u_{I,I}$.
end do

Algorithm 1: Matrix computation for the collocation.

Compute the matrix pattern according to (2.32).
do for any collocation point P_I, $I \in \mathcal{M}$
  Determine the set $\mathcal{N}(P_I)$ of $\tilde{I} \in \mathcal{N}$ with $P_I \in \operatorname{supp} \vartheta_{\tilde{I}}$.
  do for any $\vartheta_{\tilde{I}}$ with $\tilde{I} \in \mathcal{N}(P_I)$
    Determine the set $\mathcal{N}(P_I, \tilde{I})$ of $J \in \mathcal{N}$ such that, according to the just determined matrix pattern, there exists a non-zero matrix entry $c^u_{\tilde{I},J}$ with $J \in \mathcal{N}$.
    do for any ψ_J with $J \in \mathcal{N}(P_I, \tilde{I})$
      Set the initial value for the matrix entry $c^u_{\tilde{I},J}$ equal to zero.
    end do
  end do
  Determine the union $\mathcal{N}_U(P_I)$ of the sets $\mathcal{N}(P_I, \tilde{I})$.
  Compute $d_I := -\frac{1}{2}\cos[n(P_I), \tau(P_I)] + \int_S H(P_I, Q) d_Q S$ by (3.8).
  Determine the knots Q_K and weights w_K, $K \in \mathcal{O}$ of the quadrature rule (3.5) in dependence on P_I and $\mathcal{N}_U(P_I)$.
* *
  do for any quadrature knot Q_K with $K \in \mathcal{O}$
    Compute $[H(P_I, Q_K)w_K]$ and $[K(P_I, Q_K)w_K]$.
    Subtract $[H(P_I, Q_K)w_K]$ from d_I.
    Determine the set $\mathcal{N}_Q(Q_K)$ of all indices $J \in \mathcal{N}_U(P_I)$ with $\psi_J(Q_K) \neq 0$.
    do for any ψ_J with $J \in \mathcal{N}_Q(Q_K)$
      Compute $\psi_J(Q_K)$ and $[K(P_I, Q_K)w_K]\psi_J(Q_K)$.
      do for any $\vartheta_{\tilde{I}}$ with $\tilde{I} \in \mathcal{N}(P_I)$
        if $J \in \mathcal{N}(P_I, \tilde{I})$ *then*
          Add $\pm[K(P_I, Q_K)w_K]\psi_J(Q_K)$ to $c^u_{\tilde{I},J}$.
        end if
      end do
    end do
  end do
* *
  Determine the set $\mathcal{N}_A(P_I)$ of $J \in \mathcal{N}$ with $\psi_J(P_I) \neq 0$.
  do for any ψ_J with $J \in \mathcal{N}_A(P_I)$
    Compute $[d_I \psi_J(P_I)]$.
    do for any $\vartheta_{\tilde{I}}$ with $\tilde{I} \in \mathcal{N}(P_I)$
      Add $\pm[d_I \psi_J(P_I)]$ to $c^u_{\tilde{I},J}$.
    end do
  end do
end do

Algorithm 2: Matrix computation for the wavelet approach.

longer do loop over the ψ_J suggest longer computing times for the wavelet algorithm. The only way for Algorithm 2 to be faster than Algorithm 1 is that the number of quadrature knots is less for Algorithm 2. Indeed, for high compression rates, the number of knots will be essentially reduced for the wavelet algorithm and, hence, a fewer number of time consuming kernel evaluations will reduce the overall computing time.

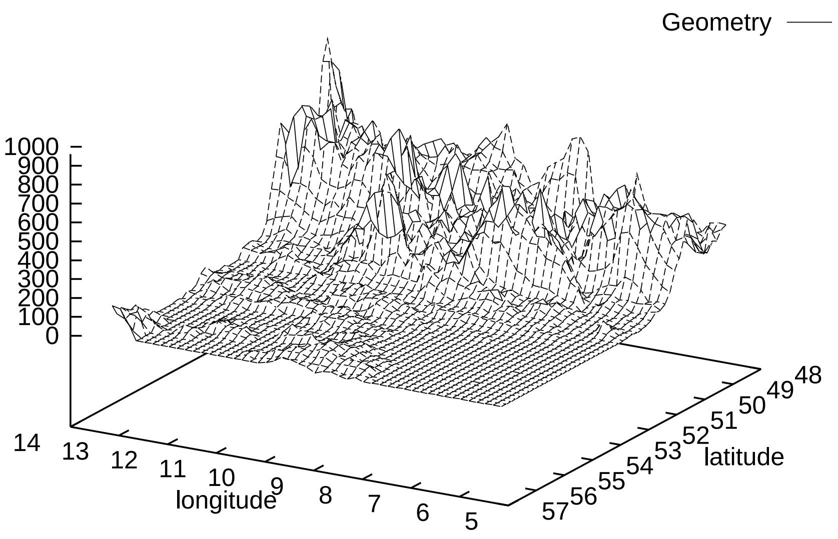


Figure 5: Height depending on latitude and longitude.

Now we make a remark on how to store the matrices. For a computation on a work station, it is important to take into account the number of I/O operations between CPU and main memory. In case of Algorithm 1, the fully populated matrix A_h can be stored as a two dimensional array. For Algorithm 2, the non-zero entries of C_h are stored in a one-dimensional array and their indices in extra arrays. Doing so, we frequently have to access to entries from C_h stored at different places of the one-dimensional array and, consequently, the ratio of I/O time per time for arithmetic operations is much worse than for Algorithm 1. To improve this ratio, we recommend to introduce an additional two-dimensional array which, for a fixed collocation point P_I, stores the values for the entries of C_h depending on $\tilde{I} \in \mathcal{N}(P_I)$ and on $J \in \mathcal{N}$ (cf. Algorithm 2 and recall that $\mathcal{N}(P_I)$ contains no more than $2L+3$ indices.). In the do loop over the quadrature knots, we sum up the values of the entries and put them into the two-dimensional array. After the loop, we write the values into the one-dimensional array and store the indices in extra arrays.

Finally, let us note that, to reduce the computing time for the evaluation of wavelet functions, the scaling factors $\sqrt{3 \cdot 2^l}$ (cf. (2.7)-(2.8)) should be dropped and the recursive structure of the function system should be applied. Namely, knowing the values of the two univariate scaling functions $\varphi_j^{N_l}$ and $\varphi_{j+1}^{N_l}$ and of the two univariate wavelet functions $\psi_{l,j'}$ and $\psi_{l,j'+1}$ of level l not vanishing at a certain point Q_k, it takes only a few number of operations to compute the values of the level $l-1$ scaling and wavelet functions not vanishing at Q_k.

Remark 4.1 *In case that the quadrature depends on the test functional (cf. Remark 3.2), then Algorithm 2 can be applied if, for a fixed collocation point P_I, all the quadrature rules depending on the test functionals $\vartheta_{\tilde{I}}$, $\tilde{I} \in \mathcal{N}(P_I)$ are stored in an array of quadrature knots, in an array of weights, and in an additional array of logical values depending on the index of the quadrature knot and the index $\tilde{I} \in \mathcal{N}(P_I)$. The last tells us whether the quadrature point is a knot corresponding to $\vartheta_{\tilde{I}}$ or not.*

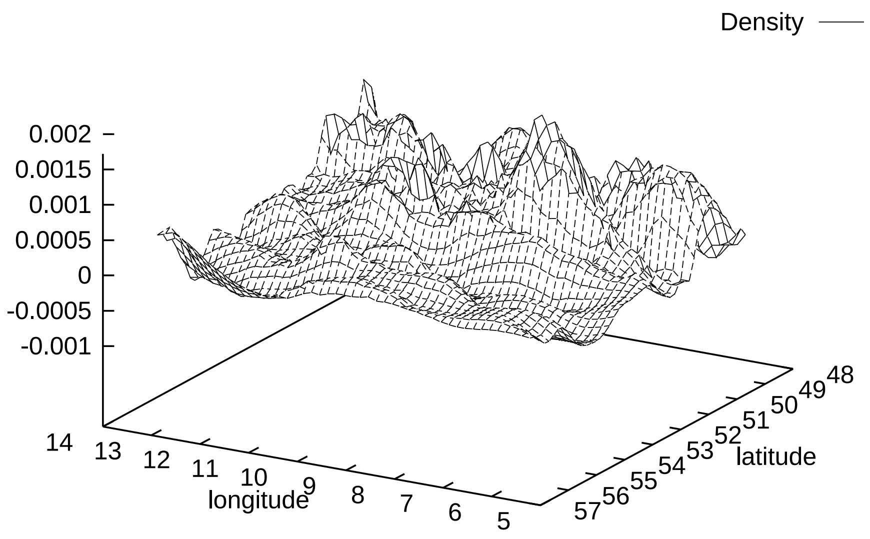


Figure 6: Density depending on latitude and longitude.

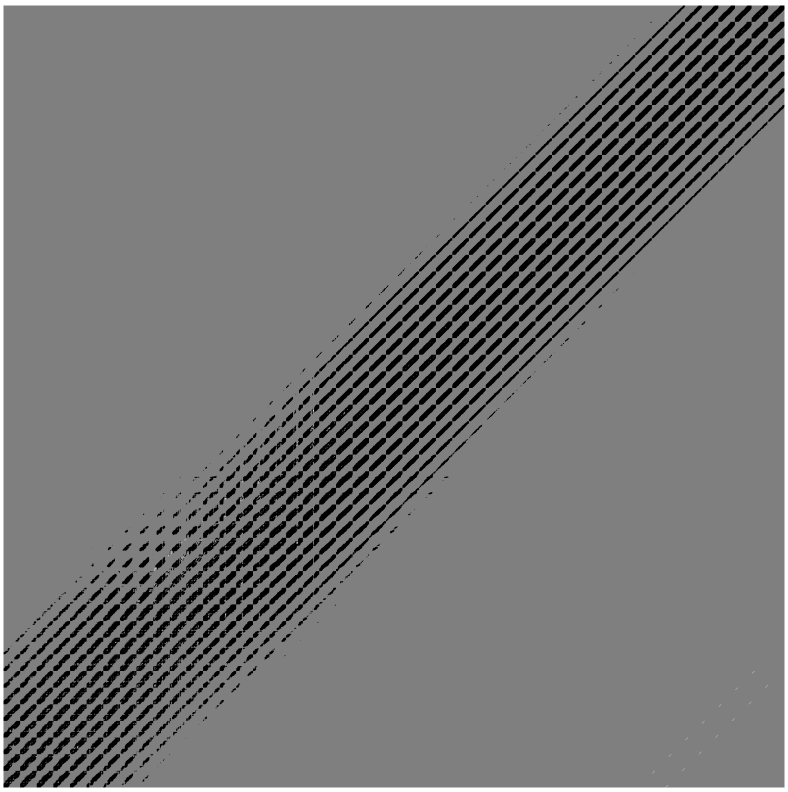


Figure 7: Matrix pattern of the stiffness matrix A_h for the collocation without wavelets.

L	1	2	3	4
err_Q	$8.8 \cdot 10^{-8}$	$1.2 \cdot 10^{-8}$	$4.0 \cdot 10^{-9}$	$1.4 \cdot 10^{-9}$

Table 1: Quadrature error for the collocation without wavelets.

4.2. Numerical Results of the Tests

For the numerical tests, we have taken a rectangular part Γ of the earth's surface containing Northern Germany (cf. Figure 5). The reference potential including the centrifugal potential is computed in a preprocessing step from an expansion into spherical harmonics up to degree/order 10. We have solved equation (1.5) by collocation with $N = N_L = 3 \cdot 2^L + 1$ and $L = 0, \ldots, 5$. The approximate density function $\tilde{u}_h$ (cf. (1.7)) is given in Figure 6. In order to check the quality of the quadrature rules from Subsection 3.1, we have computed another solution u_h^+ by halving the size of the underlying quadrature mesh and by refining the subdomains $S_{i,j},\ i,j \geq 0$ in x direction with additional grid points of a polynomially graded mesh (cf. Subsection 5.1). We have obtained the estimates for the quadrature errors $err_Q := \sup |u_h(P_I) - u_h^+(P_I)|$ given in Table 1. The supremum in the definition of err_Q is taken over four different points, two fixed in the interior of S and two as close as possible to corners of ∂S. Note that we are dealing with errors for values with a size of about $5 \cdot 10^{-4}$. These estimates show that the quadrature rules of Subsection 3.1 are satisfactory or even too good. The quadrature rules of Subsection 3.2 for the wavelet algorithm are also quite accurate since replacing them by those of Subsection 3.1 has not led to a reduction of the quadrature error.

Unfortunately, we have no good estimate for the discretisation errors of the collocation method (1.9). We believe them to be much larger than the quadrature errors. Since discretisation errors play the role of thresholds for the compression errors, we somehow have to choose an estimate. In view of a comment in [19] on the Galerkin discretisation error over a different domain, we assume a relative discretisation error in the supremum norm of $4^{5-L} \cdot 10^{-5}$ for the collocation solution u_h with $h = \frac{1}{N_L - 1}$, $N_L = 3 \cdot 2^L + 1$. In other words, we suppose an error 10^{-5} for $L = 5$ and determine the errors for the lower levels by assuming a second order convergence rate. Now let us have a look at the size of the entries in the stiffness matrices A_h^u and B_h^u (cf. Subsection 4.1), respectively. In the Figures 7 and 8 we have marked the entries greater than 10^{-4} by black pixels and those less than 10^{-4} but greater than 10^{-5} by grey ones. These pictures obtained for $N = N_4$ clearly demonstrate that the wavelet transform B_h^u contains much more small entries than A_h^u. In other words, B_h^u seems to be appropriate for a compression.

We have implemented the a priori compression (2.32), and the constant a has been chosen by experiment such that the additional error due to the compression is less than the estimate for the discretisation error. However, instead of the supremum norm we have taken the maximum over the four points used for the computation of err_Q. The values for the optimal constants a and the compression rates rat (rat = number of all entries divided by the number of non-zero entries in C_h) are presented in Table 2 (cf. Figure 9). By $knot_C$ we have denoted the sum $\sum_P \#\mathcal{O}$ of the numbers $\#\mathcal{O}$ of all quadrature knots in the rule of Subsection 3.1 depending on the collocation point P. Similarly, by $knot_W$ we have denoted the number of all quadrature knots in all the rules of Subsection 3.2. These numbers are also the numbers of evaluations of kernel functions in the Algorithms

L	0	1	2	3	4	5
N_L	4	7	13	25	49	97
$(N_L-2)^2$	4	25	121	529	2 209	9 025
$a \cdot 2^L$	0	0	0	0.1	0.5	1.4
rat	1	1.19	2.69	6.82	12.64	13.99
$knot_C$	592	6 400	83 248	1 278 064	20 605 552	
$knot_W$	592	6 400	58 120	416 920	3 132 784	34 085 800
tim_C	2.84s	2.80s	5.34s	43.49s	656.79s	
tim_W	2.87s	3.14s	5.23s	19.72s	139.65s	2 165.30s

Table 2: Compression rates, number of quadrature knots, and computation time for the algorithms without and with wavelets. Approximation of the density function.

2 and 3, respectively. The last two rows in Table 2 contain the computing times tim_C for Algorithm 1 without wavelets and tim_W for the wavelet algorithm Algorithm 2.

We note that the storage capacity of 96 MB of our work station was not sufficient for solving (1.5) on level $L = 5$ without wavelets. The introduction of wavelets admits compressions of the matrix up to 8% for $N = N_5$. Of course, the saving of storage is a little bit less since additional working space is required. The smaller numbers of kernel evaluations lead to a reduction in the computing time from $tim_C = 657s$ to $tim_W = 140s$ for $N = N_4$ and from estimated $tim_C = 10\,509s$ to $tim_W = 2\,165s$ for $N = N_5$.

The compression rates and computing times depend strongly on the threshold for the compression. In a second experiment we have computed a directional derivative of the potential δw_h from (1.7) over eight points placed in a height of 0.01m, 0.1m, 1m, 10m, 100m, 1 000m, 10 000m, and 100 000m over the surface point with longitude 9.333 and latitude 52.666. The parameter a in the compression has been chosen such that the relative compression errors for the resulting eight derivatives of the potential are less than $4^{5-L} \cdot 10^{-5}$. The values a, the compression rates rat, the numbers of quadrature knots $knot_C$, $knot_W$, as well as the computation times tim_C, tim_W are presented in Table 3 (cf. also 9). It has turned out that, for the computation of the potential far from the points of the boundary ∂S, much higher compression rates are possible. For $N = N_5$, the stiffness matrix can be compressed up to 5.1%. The smaller numbers of kernel evaluations lead to a reduction in the computing time from $tim_C = 669s$ to $tim_W = 125s$ for $N = N_4$ and from estimated $tim_C = 10\,557s$ to $tim_W = 890s$ for $N = N_5$. The same is likely to be true for the determination of the density u_h over a part of S which has a positive distance to ∂S.

All the computations have been performed on a DEC 3 000 AXP 400 α- processor work station.

L	0	1	2	3	4	5
N_L	4	7	13	25	49	97
$(N_L - 2)^2$	4	25	121	529	2 209	9 025
$a \cdot 2^L$	0	0	0	0	0.2	0.5
rat	1	1.19	2.69	6.83	19.44	36.90
$knot_C$	592	6 400	83 248	1 278 064	20 605 552	
$knot_W$	592	6 400	58 120	416 920	2 486 392	17 579 560
tim_C	3.55s	3.67s	7.06s	47.12s	669.19s	
tim_W	3.68s	4.43s	7.10s	24.58s	124.71s	889.78s

Table 3: Compression rates, number of quadrature knots, and computation time for the algorithms without and with wavelets. Approximation of the potential.

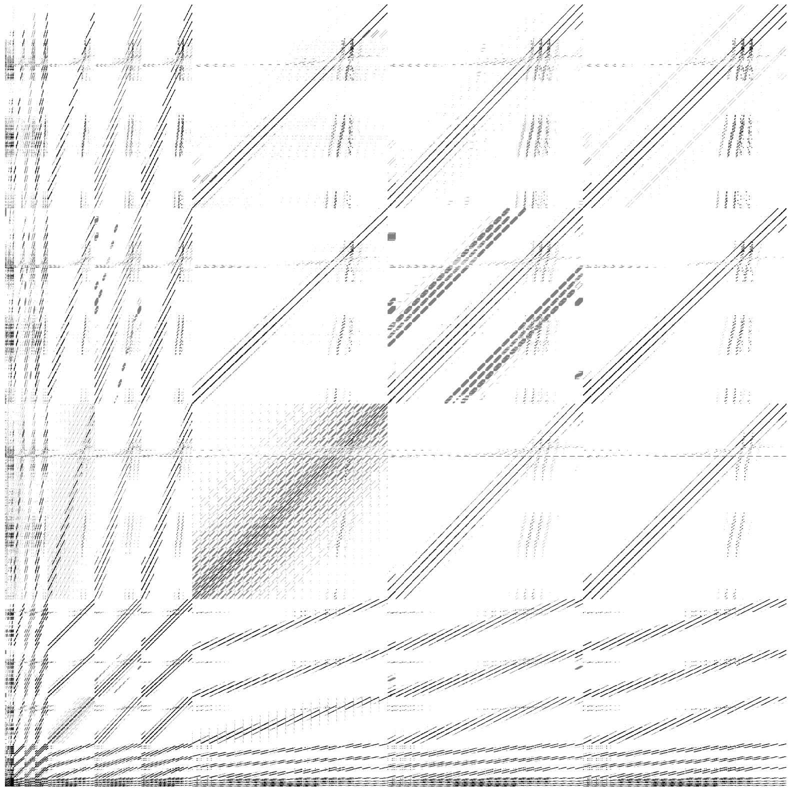


Figure 8: Matrix pattern of the stiffness matrix B_h, i.e., of the wavelet transform of A_h.

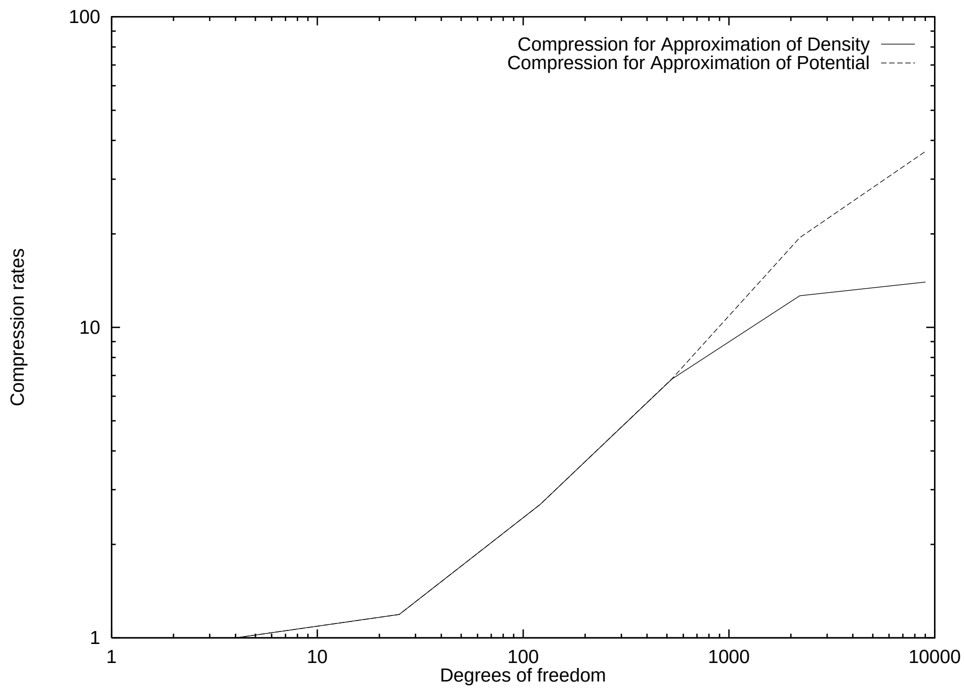


Figure 9: Compression rates of the wavelet algorithm depending on the degrees of freedom $(N_L - 2)^2$ for the approximation of the density function.

Acknowledgements. I am grateful to R. Klees for helpful discussions and for supplying me with data for the numerical tests.

References

[1] B.K. Alpert, *Construction of simple multiscale bases for fast matrix operation*, in *Wavelets and their applications*, M.B. Ruskai, G. Beylkin, R. Coifman, I. Daubechies, S. Mallat, Y. Meyer, and L. Raphael, eds: Jones and Bartlett Publishers, Boston, 211–226, 1992.

[2] G. Beylkin, R. Coifman, and V. Rokhlin, *Fast wavelet transforms and numerical algorithms I*, Comm. Pure Appl. Math. **44** (1991), 141–183.

[3] A. Brandt and A.A. Lubrecht, *Multilevel matrix multiplication and fast solution of integral equation*, J. Comput. Phys. **90** (1991), 348–370.

[4] C.K. Chui, *An introduction to wavelets*, Academic Press, Boston, 1992.

[5] A. Cohen, I. Daubechies, and J.-C. Feauveau, *Biorthogonal bases of compactly supported wavelets*, Comm. Pure and Appl. Math. **45** (1992), 485–560.

[6] W. Dahmen, B. Kleemann, S. Prößdorf, and R. Schneider, *A multiscale method for the double layer potential equation on a polyhedron*, in *Advances of Computational Mathematics*, H.P. Dikshit and C.A. Micchelli, eds., World Scientific Publ. Co., Inc., New Delhi, 15–57, 1994.

[7] W. Dahmen, B. Kleemann, S. Prößdorf, and R. Schneider, *Multiscale methods for the solution of the Helmholtz and Laplace equations*, in *Boundary Element Methods*, W. Wendland, ed., Reports from the Final Conference of the Priority Research Programme 1989-1995 of the German Research Foundation, Oct. 2-4, 1995, Stuttgart, to appear in Springer-Verlag.

[8] W. Dahmen and A. Kunoth, *Multilevel preconditioning*, Numer.Math. **63** (1992), 315–344.

[9] W. Dahmen, S. Prößdorf, and R. Schneider, *Multiscale methods for pseudo-differential equations*, in *Recent Advances in Wavelet Analysis*, L.L. Schumaker and G. Webb, eds., Wavelet Analysis and its Application **3**, (1994), 191–235 .

[10] W. Dahmen, S. Prößdorf, and R. Schneider, *Wavelet approximation methods for pseudo-differential equations I: Stability and convergence*, Math. Zeitschr. **215** (1994), 583–620.

[11] W. Dahmen, S. Prößdorf, and R. Schneider, *Wavelet approximation methods for pseudo-differential equations II: Matrix compression and fast solution*, Advances in Comp. Math. **1** (1993), 259-335.

[12] W. Dahmen, S. Prößdorf, and R. Schneider, *Multiscale methods for pseudo-differential equations on smooth manifolds*, in *Wavelets: Theory, Algorithms, and Applications*, C.K. Chui, L. Montefusco, and L. Puccio., eds., Academic Press, 1–40, 1994.

[13] I. Daubechies, *Ten lectures on wavelets*, CBMS Lecture Notes, No. 61, SIAM, Philadelphia, 1992.

[14] M. Dorobantu, *Potential integral equations of the 2D Laplace operator in wavelet basis*, Preprint TRITA-NA-9401, Royal Institute of Technology, University of Stockholm, 1994.

[15] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulation*, J. Comput. Phys. **73** (1987), 325–348.

[16] W. Hackbusch and Z.P. Nowak, *On the fast matrix multiplication in the boundary element method by panel clustering*, Numer. Math. **54** (1989), 463–491.

[17] A. Harten and I. Yad-Shalom, *Fast multiresolution algorithms for matrix-vector multiplication*, SIAM J. Numer. Anal. **31** (1994), 1191–1218.

[18] R. Klees and J.Engels, *Galerkin versus collocation - Comparison of two discretization methods for the boundary integral equations in* $I\!R^3$, Manuscripta Geodetica **17** (1992), 245–256.

[19] R. Klees, *Lösung des geodätischen Randwertproblems mit Hilfe der Randelementmethode*, German Geodetic Kommission (DGK), Series C, No.382, Munich, 1992.

[20] S.G. Mikhlin and S. Prößdorf, *Singular integral operators*, Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, 1986.

[21] C. Miranda, *Partial differential equations of elliptic type*, Ergebnisse der Mathematik und ihrer Randgebiete, Band 2, Springer-Verlag, Berlin, Heidelberg, 1970.

[22] H. Moritz, *Advanced physical geodesy*, Abacus Press, Tunbridge Wells, Kent, and Herbert Wichmann Verlag, Karlsruhe, 1980.

[23] A.W. Overhauser, *Analytic definition of curves and surfaces by parabolic blending.*, Techn. Report No. S168-40, Scientific Research Staff Publication, Ford Motor Company, Detroit, 1968.

[24] T.v. Petersdorff and C. Schwab, *Boundary element methods with wavelets and mesh refinement*, Research Report No.95-10, SAM, Eidgenössische Technische Hochschule Zürich, 1995, to appear in: Proc. of ICIAM 1995, Hamburg, Special issue of ZAMM.

[25] T.v. Petersdorff and C. Schwab, *Fully discrete multiscale Galerkin BEM*, Research Report No.95-08, SAM, Eidgenössische Technische Hochschule Zürich, 1995, to appear in *Multiresolution Analysis and PDE*, A.J. Kurdial and P. Oswald, eds., Series: Wavelet Analysis and its Applications, Academic Press 1996.

[26] T.v. Petersdorff and C. Schwab, *Wavelet approximations for first kind boundary integral equations on polygons*, Technical Note BN-1157, Institute for Physical Science and Technology, University of Maryland at College Park, 1994, submitted to Numer.Math. .

[27] S. Prößdorf and R. Schneider, *A spline collocation method for multidimensional strongly elliptic pseudodifferential operators of order zero*, Integral Equations and Operator Theory 14 (1991), 399–435.

[28] A. Rathsfeld, *A wavelet algorithm for the solution of a singular integral equation over a smooth two-dimensional manifold*, to appear.

[29] A. Rathsfeld, *A wavelet algorithm for the solution of the double layer potential equation over polygonal boundaries*, Journal of Integral Equations and Applications **7** (1995), 47–97.

[30] V. Rokhlin, *Rapid solution of integral equations of classical potential theory*, J. Comput. Phys. **60** (1983), 187–207.

[31] S. Sauter, *Der Aufwand der Panel-Clustering-Methode für Integralgleichungen*, Bericht Nr. 9115, Inst. f. Inform. u. Prakt. Math., Christian-Albrechts-Univ. Kiel, 1991.

[32] R. Schneider, *Multiskalen- und Waveletkompression: Analysisbasierte Methoden zur effizienten Lösung großer vollbesetzter Gleichungssysteme*, Habilitationsschrift, Fachbereich Mathematik, Technische Hochschule Darmstadt, 1995.

[33] C. Zenger, *Sparse grids*, in *Parallel algorithms for Partial Differential Equations, Proceedings of the sixth GAMM Seminar 1990*, W. Hackbusch, ed., Vieweg-Verlag, Braunschweig, 241–251, 1991.