Solution of Time Dependent Diffusion Equations with Variable Coefficients using Multiwavelets

appeared in J. Computational Physics, Vol. 150, pp. 394-424, 1999.

A. Averbuch †, M. Israeli ‡ and L. Vozovoi †
† School of Mathematical Sciences, Tel Aviv University, Tel Aviv 69978, Israel
‡ Faculty of Computer Science, Technion, Haifa 32000, Israel
appeared in J. of Computational Physics,Vol. 150, pp. 394-424, 1999

Abstract

A new numerical algorithm is developed for the solution of time dependent differential equations of the diffusion type. It allows for an accurate and efficient treatment of multidimensional problems with variable coefficients, nonlinearities and general boundary conditions. For space discretization we use the multiwavelet bases introduced in [1] which were then applied to the representation of differential operators and functions of operators in [3]. An important advantage of multiwavelet basis functions, is the fact that they are supported only on non overlapping subdomains . Thus multiwavelet bases are attractive for solving problems in finite (non periodic) domains.

Boundary conditions are imposed with a penalty technique of [12], it can be used to impose rather general boundary conditions.

The penalty approach was extended to a procedure for ensuring the continuity of the solution and its first derivative across interior boundaries between neighboring subdomains while time stepping the solution of a time dependent problem. This penalty procedure on the interfaces allows for a simplification and sparsification of the representation of differential operators by discarding the elements responsible for interactions between neighboring subdomains. Consequently the matrices representing the differential operators (on the finest scale) have block diagonal structure.

For a fixed order of multiwavelets (i.e. a fixed number of vanishing moments) the computational complexity of the present algorithm is proportional to the number of subdomains.

The time discretization method of [8] is used in view of its favorable stability properties.

Numerical results are presented for evolution equations with variable coefficients in one and two dimensions.

Key words: multiwavelets, nonlinear evolution problems, variable coefficients, boundary conditions, penalty procedure, multidimensional problems.

I Introduction

We present a new numerical algorithm for the solution of nonlinear time dependent evolution equations

$$u_t = \mathcal{L}u + \mathcal{N}(u)$$

in finite domains, where $\mathcal{L}$ represents the linear part, and $\mathcal{N}(\cdot)$ the nonlinear part of the evolution operator. We focus on linear operators of the diffusion type with variable coefficients. For example, in one dimension $\mathcal{L} = a(x)\frac{\partial^2}{\partial x^2}$.

In this paper we address the following issues:

– stable time integration method (time discretization schemes with good stability properties);

– efficient computation of operators with variable coefficients;

– efficient computation of global differential operators (for example exponential functions of differential operators);

– treatment of general boundary conditions (periodic, Dirichlet, Neumann, Robin);

– multidimensional problems.

The present algorithm incorporates several techniques. For the *time discretization* we employ a new method proposed in [8]. A distinctive feature of this method is the exact evaluation of the contribution of the linear part (therefore, the corresponding schemes are labeled as "exact linear part" (*ELP) schemes*). As a result, this method has very good stability properties since possible instability may be due only to the nonlinear term. Typically, the stability of time-discretization schemes for advection-diffusion equations are controlled by the linear, diffusion, term and therefore these equations require implicit treatment in order to avoid the use of unreasonably small time steps. In contrast, using the explicit ELP scheme, it is possible to achieve stable time steps usually associated with implicit schemes.

Implementation of these new schemes requires the evaluation of functions of operators (e.g. exponentials). Computing and applying exponential or other functions of operators typically requires evaluating dense matrices and, therefore, expensive. An exception is the case where there is a (fast) transform that diagonalizes the operator. For example, if $\mathcal{L}$ is a convolution (or a circulant) matrix which is diagonalized by the Fourier Transform (FT), then the computation of functions of operators can be accomplished by a fast algorithm, for example the FFT.

Differential operators with non-constant coefficients cannot be diagonalized by the FFT. However, it turns out [6] that a wide class of operators with non-constant coefficients have sparse (finite accuracy) representations in the wavelet basis. In particular, computing exponentials of elliptic operators with variable coefficients in the wavelet system of coordinates always results in sparse matrices. For the present algorithm we use *multiwavelet* bases for the *spatial discretization*. These bases were introduced in [1]. In [3] representations of differential operators and functions of operators in bases of multiwavelets were constructed. A discrete version of multiwavelets was studied and used for representing integral operators in [2].

Multiwavelet bases possess most of the properties of wavelet bases such as vanishing moments, orthogonality and compact support. However, in contrast with "conventional" wavelet bases, the multiwavelet basis functions do not overlap on a given scale and are organized in small groups of several functions sharing the same support. Again, wide classes of operators have sparse finite accuracy representations in these bases due to the vanishing moments property of the basis functions. These properties make the multiwavelet bases a useful tool for solving partial differential equations. In particular, it allows for accommodating boundary conditions in an accurate and simple manner.

Typically, for smooth wavelets with overlapping support, such as Daubechies' wavelets [9], accommodating boundary conditions results in a loss of quality of approximation near the boundary either due to the loss of the order of approximation or due to the high sensitivity to a change in the boundary values (large condition number of the corresponding operators). It was shown in [3] by considering representation of differential operators in multiwavelet bases, that the high order of the scheme is maintained up to the boundary.

On the other hand, the multiwavelet basis functions are discontinuous (similar to the Haar basis functions) and may not fit the definitions of wavelets which typically require regularity. In particular, representations of differential operators in such a basis do not exist in the ordinary sense. It is possible , however, to construct weak representations, i.e. representations which are accurate up to an appropriate order for a class of smooth test functions, e.g. $C^\infty([0,1])$. Such representations appear to be perfectly adequate for computational purposes.

When a time dependent problem is solved using an explicit time integration scheme, boundary conditions should be imposed, and the values on boundaries have to be computed such that they satisfy the correct time dependent boundary conditions. In the present algorithm we adopt a *penalty procedure* of [12] to impose the *boundary conditions*. The penalty term is introduced as a forcing in the evolution equation. Its amplitude is proportional to the difference between the numerical and the prescribed boundary values.

As we already mentioned, the implementation of the time discretization method using ELP schemes requires applying exponential functions of operators. We also pointed out that functions of operators with non-constant coefficients are sparse in the wavelet system of coordinates. However the speed of evaluation and application of such operators in two and three spatial dimensions remains an important consideration in assessing the practicality of ELP schemes. Although the representations of differential operators in multiwavelet bases, constructed in [3], scale properly with size in all dimensions, reducing the constants in operation counts remains an important task.

In this paper we propose a procedure which allows a drastic reduction in the computational complexity of multiwavelet algorithms, especially in more than one dimensions. The idea is to simplify the construction of the linear operator $\mathcal{L}$ in the multiwavelet basis by discarding the off-diagonal blocks in the matrices $\mathcal{L}$ which are responsible for the interaction with neighboring intervals. In effect, the matrix $\mathcal{L}$ in the multiwavelet basis (on the finest scale) obtains a block diagonal structure with the block size equal to the number of vanishing moments, k, where the number of blocks is equal to the number of subintervals, N.

In order to restore the *interaction between subintervals* we developed a *penalty approach on the interfaces*, similar to that used on the boundaries. The corresponding forcing term in the evolution equation has a very simple structure in the physical domain (exponentials attached to the interfaces) which can be described by a few multiwavelet coefficients. As a result, the complexity of the computational algorithm reduces drastically. For example, the operation count for evaluating in $d-$ dimensions (global) exponential functions of differential operators on the finest scale drops from $O(N^{3d}k^{3d})$ (when the interaction was incorporated into the operator) to the order of $O(N^{d}k^{3d})$ (when using block-diagonal matrices along with the penalty procedure on the interfaces).

The paper has the following structure. In section II we formulate the problem and describe the time discretization method of [8]. Here we also describe a *generalized scaling and squaring method* for computing the operator valued quadrature coefficients of the ELP schemes. In section III we summarize results of [1] and [3] on multiwavelet bases and representing operators in this basis. In section IV we describe a penalty procedure on the boundaries which ensure the prescribed boundary conditions when solving a time evolution problem. In section V we introduce a simplified construction of the operators and functions of operators (without incorporating the interaction between the neighboring subdomains), and a novel penalty procedure on the interfaces which preserves the continuity of the solution and its first derivative. In section VI we generalize the algorithm to the two dimensional case. We also provide some numerical tests in one and two spatial dimensions.

II Time Discretization Method

II.1 Mathematical Formulation

We are interested in the numerical solution of nonlinear evolution equations of the form

$$u_t = \mathcal{L}u + \mathcal{N}(u), \qquad \text{in} \quad \Omega, \tag{2.1}$$

where $\mathcal{L} = a(x)\frac{\partial^2}{\partial x^2}$ is the linear diffusion operator, $\mathcal{N}(\cdot)$ is the nonlinear part of the operator, $u = u(x,t)$, $x \in \Omega = [0,1]^d$, $d = 1,2,3$ and $t \in [0,T]$. The important examples of nonlinear diffusion equations of the type (2.1) are the advection-diffusion equation and, in particular, the Navier-Stokes equations which can be rewritten in the form (2.1), see [8].

We also supply the initial conditions

$$u(x,0) = u_0(x), \qquad \text{in} \quad \Omega, \tag{2.2}$$

and the linear boundary conditions

$$\mathcal{B}u(x,t) = 0, \qquad \text{on} \quad \partial\Omega, \quad t \in [0,T]. \tag{2.3}$$

where $\partial\Omega$ is the boundary of the computational region Ω.

II.2 ELP Schemes

For time discretization we use the method introduced in [8]. A distinctive feature of this method is the exact evaluation of the contribution of the linear part. Namely, if the nonlinear part is zero, then the method reduces to the evaluation of the exponential function of the operator (or matrix) $\mathcal{L}$ that represents the linear part. There was derived a family of implicit and explicit schemes called the "exact linear part" (ELP) schemes.

It was shown that such schemes have very good stability properties since the instability is related solely to the nonlinear term. For example, when explicit ELP schemes are applied to the advection-diffusion equation with the linear operator $\mathcal{L} = \frac{\partial^2}{\partial x^2}$, the stability restriction on the time step is $\Delta t \sim O(h^{-1})$ where h is the grid size (we recall that for the explicit schemes the typical condition is $\Delta t \sim O(h^{-2})$). Thus, the *explicit* ELP schemes have the stability regions similar to those of typical *implicit* schemes used in e.g. fluid dynamics applications. In this section we summarize briefly the main technique of the ELP method.

We start by converting the initial value problem (2.1)-(2.2) to a nonlinear integral equation of the form

$$u(x,t) = e^{(t-\eta)\mathcal{L}}u(x,\eta) + \int_0^t e^{(t-\tau)\mathcal{L}}\mathcal{N}(u(x,\tau))d\tau, \tag{2.4}$$

where $0 \le \eta < t$. It can be verified (by differentiating with respect to t) that (2.4) is equivalent to (2.1).

Next, we discretize (2.4) in time. Consider the function $u(x,t)$ at the discrete moments of time $t_n = n\Delta t$ where Δt is the time step. Denote $u_n = u(x, t_n)$. The discretized equation reads

$$u_{n+1} = e^{m\Delta t\mathcal{L}} u_{n-m} + \Delta t \left(\gamma N_{n+1} + \sum_{\mu=0}^{M-1} \beta_\mu N_{n-\mu} \right), \tag{2.5}$$

where $u_n = u(x,t_n)$, $t_n = n\Delta t$, $N_n = \mathcal{N}(u_n)$, and $M+1$ is the number of time levels.

The discrete parameter m can be chosen arbitrarily in the interval $1 \le m \le M$. From the approximation point of view any of these choices is equally good. In the particular case where $l = 2, \gamma = 0$ and $M = 1$, equation (2.5) turns into the explicit scheme known as the ''slave-frog" scheme,

$$u_{n+1} = e^{2\Delta t\mathcal{L}} u_{n-1} + \Delta t \beta_0 N_n, \qquad \beta_0 = \frac{e^{2\Delta t\mathcal{L}} - 1}{\Delta t\mathcal{L}}. \tag{2.6}$$

This scheme has been used in computational fluid dynamics, see e.g. [10]. Authors do not know other examples of temporal schemes related to the family (2.5).

The stability analysis shows [8], that the schemes with $m = 2$, in particular (2.5), do not have good stability properties. From the stability point of view the most preferable schemes with $m = 1$ ($t - \eta = \Delta t$).

The coefficients γ and β_m are the operators. More precisely, they are functions of the operator $\Delta t\mathcal{L}$. Denote

$$\mathrm{Q}_j(\Delta t\mathcal{L}) = \frac{\mathrm{e}^{\Delta t\mathcal{L}} - \mathrm{E}_j(\Delta t\mathcal{L})}{(\Delta t\mathcal{L})^j}, \qquad \mathrm{E}_j(\Delta t\mathcal{L}) = \sum_{k=0}^{j-1} \frac{(\Delta t\mathcal{L})^j}{j!}. \tag{2.7}$$

Thus,

$$\mathrm{Q}_0(x) = \mathrm{e}^x, \quad \mathrm{Q}_1(x) = \frac{\mathrm{e}^x - 1}{x}, \quad \mathrm{Q}_2(x) = \frac{\mathrm{e}^x - 1 - x}{x^2}, \ \ldots \tag{2.8}$$

and $\mathrm{E}_j(x)$ is a truncated expansion of the exponential e^x.

Table 1 gives the expressions for the coefficients of the $1st$, $2d$ and $3d$ order explicit schemes ($\gamma = 0$) in terms of $\mathrm{Q}_j = \mathrm{Q}_j(\Delta t\mathcal{L})$.

II.3 Evaluation of the Operator-Valued Quadrature Coefficients

We will now describe the method that permits us to compute the operators Q_0, Q_1, Q_2, etc. without computing directly the exponential $e^{\Delta t\mathcal{L}}$ and the inverse operator $(\Delta t\mathcal{L})^{-1}$.

M	β_0	β_1	β_2	order
1	Q_1	0	0	1
2	$\mathrm{Q}_1+\mathrm{Q}_2$	$-\mathrm{Q}_2$	0	2
3	$\mathrm{Q}_1+3\mathrm{Q}_2/2+\mathrm{Q}_3$	$-2(\mathrm{Q}_2+\mathrm{Q}_3)$	$\mathrm{Q}_2/2+\mathrm{Q}_3$	3

Table 1: Coefficients of the $1st$, $2d$ and $3d$ order explicit schemes; $\gamma=0$, $\mathrm{Q}_j=\mathrm{Q}_j(\Delta t\mathcal{L})$.

A simple way for computing the exponential $e^{\Delta t\mathcal{L}}$ is to use a first order Taylor expansion

$$e^{\Delta t\mathcal{L}} \approx \mathcal{I}+\Delta t\mathcal{L} \tag{2.9}$$

where $\mathcal{I}$ is the identity operator. The problem with using such an approximation is that it will result in a loss of accuracy due to possibly large singular values of $\Delta t\mathcal{L}$.

A much more accurate *scaling and squaring* method [4] is based on the identity

$$e^x=\left(e^{x/n}\right)^n. \tag{2.10}$$

Approximating the exponential in the right hand side of (2.10) by the first order Taylor expansion gives

$$e^x\approx\left(1+\frac{x}{n}\right)^n. \tag{2.11}$$

We note that the above approximation is accurate when n is large enough even though x is not small. In the limit $n\to\infty$ the estimation (2.11) is merely exact. It can be shown that the relative error of the approximation (2.11) is $x^2/2n$ as opposed to the error $x^2/2$ for the first order Taylor expansion $e^x\approx 1+x$.

The scaling and squaring method can be generalized in order to compute the exponential operators $\mathrm{Q}_j(\Delta t\mathcal{L})$, $j=1,2,\ldots$, see [8]. The principal step here is to express these functions in terms of the functions of a half argument. Based on the formulas (2.7) it is easy to verify that

$$\begin{aligned}
\mathrm{Q}_0(2x) &= \mathrm{Q}_0(x)\mathrm{Q}_0(x),\\
\mathrm{Q}_1(2x) &= \tfrac{1}{2}\left(\mathrm{Q}_0(x)\mathrm{Q}_1(x)+\mathrm{Q}_1(x)\right),\\
\mathrm{Q}_2(2x) &= \tfrac{1}{4}\left(\mathrm{Q}_1(x)\mathrm{Q}_1(x)+2\mathrm{Q}_2(x)\right),\\
\mathrm{Q}_3(2x) &= \tfrac{1}{8}\left(\mathrm{Q}_1(x)\mathrm{Q}_2(x)+\mathrm{Q}_2(x)+2\mathrm{Q}_3(x)\right),\\
\mathrm{Q}_4(2x) &= \tfrac{1}{16}\left(\mathrm{Q}_2(x)\mathrm{Q}_2(x)+2\mathrm{Q}_4(x)+2\mathrm{Q}_3(x)\right)
\end{aligned} \tag{2.12}$$

etc. We note that a function $\mathrm{Q}_j(2x)$ is expressed in terms of the functions $\mathrm{Q}_k(x)$, $k=0,\ldots,j$.

Below we summarize the modified scaling and squaring method for computation of the operator-valued quadrature coefficients of the ELP schemes. We start by computing $\mathrm{Q}_0(\Delta t_1\mathcal{L})$, $\mathrm{Q}_1(\Delta t_1\mathcal{L})$, $\mathrm{Q}_2(\Delta t_1\mathcal{L})$ etc, $\Delta t_1=2^{-J}\Delta t$, for some J selected so that the largest singular value of all operators $\mathrm{Q}_j(\Delta t_1 \mathcal{L})$ is less than one. For this evaluations we use the Taylor series

$$\begin{aligned}
\mathrm{Q}_0(\Delta t_1 \mathcal{L}) &= \mathcal{I} + \Delta t_1 \mathcal{L}, && (2.13)\\
\mathrm{Q}_1(\Delta t_1 \mathcal{L}) &= \mathcal{I} + \Delta t_1 \mathcal{L}/2, && (2.14)\\
\mathrm{Q}_2(\Delta t_1 \mathcal{L}) &= \mathcal{I}/2 + \Delta t_1 \mathcal{L}/6, && (2.15)\\
\mathrm{Q}_3(\Delta t_1 \mathcal{L}) &= \mathcal{I}/6 + \Delta t_1 \mathcal{L}/24, \quad etc. && (2.16)
\end{aligned}$$

where $\mathcal{I}$ is the identity operator and $\Delta t_1 = 2^{-J}\Delta t$. We then proceed by using the identities in (2.12) recursively J times to compute the operators for the required value of the argument, $\Delta t \mathcal{L}$. Finally, we compute the coefficients β_k which are the linear combinations of $\mathrm{Q}_j(\Delta t \mathcal{L})$ (for example as in Table 1 for explicit ELP schemes).

Thus, the computation of the operators Q_j is reduced to the evaluation of a much more simple differential operator $\Delta t \mathcal{L}(x) = \Delta t a(x) \frac{\partial^2}{\partial x^2}$. The numerical procedure for computing this operator is described in the following sections.

III Discretization in Space. Multiwavelet basis

For space discretization we use expansion of functions into the multiwavelet basis. This basis was introduced in [1]. Representation of differential constant coefficient operators in this basis was constructed in [3].

III.1 Construction and Properties of the Multiwavelet basis

In this section we summarize some properties of the multiwavelet bases [1].

For $k = 1, 2, \ldots$ and $n = 0, 1, 2, \ldots$ we define $\mathbf{V}_n^k$ as a space of piecewise polynomial functions,

$$\mathbf{V}_n^k = \{f : \begin{array}{l} \text{the restriction of } f \text{ to the interval } [2^{-n}l,\ 2^{-n}(l+1)] \text{ is} \\ \text{a polynomial of degree less than } k, \text{ for } l = 0, \ldots, 2^n - 1, \\ \text{and } f \text{ vanishes elsewhere.}\} \end{array} \tag{3.1}$$

The space $\mathbf{V}_n^k$ has dimension $2^n k$ and

$$\mathbf{V}_0^k \subset \mathbf{V}_1^k \ldots \subset \mathbf{V}_n^k \subset \ldots \tag{3.2}$$

We define $\mathbf{W}_n^k$, $n = 0, 1, 2, \ldots$ as the orthogonal complement of $\mathbf{V}_n^k$ in $\mathbf{V}_{n+1}^k$,

$$\mathbf{V}_n^k \bigoplus \mathbf{W}_n^k = \mathbf{V}_{n+1}^k, \qquad \mathbf{W}_n^k \perp \mathbf{V}_n^k, \tag{3.3}$$

and note that $\mathbf{W}_n^k$ is of dimension $2^n k$. Therefore, we have

$$\mathbf{V}_n^k = \mathbf{V}_0^k \bigoplus \mathbf{W}_0^k \bigoplus \mathbf{W}_1^k \bigoplus \cdots \bigoplus \mathbf{W}_{n-1}^k. \tag{3.4}$$

We define $\mathbf{V}^k$ as the union of all subspaces $\mathbf{V}_n^k$,

$$\mathbf{V}^k = \bigcup_{n=0}^{\infty} \mathbf{V}_n^k \tag{3.5}$$

and observe that $\mathbf{V}^k$ is dense in $\mathbf{L}^2[0,1]$ with respect to the norm,

$$||f|| = \langle f, f \rangle^{1/2}, \qquad \langle f, g \rangle = \int_0^1 f(x)g(x)dx.$$

Let $\{\phi_0, \ldots, \phi_{k-1}\}$ denote an orthonormal basis for $\mathbf{V}_0^k$ (scaling functions). Such a basis can be constructed using the system of the first k Legendre polynomials $P_j(x)$, $x \in [-1,1]$. We define the scaling functions ϕ_j, $j = 0, \ldots, k-1$ as

$$\phi_j(x) = \begin{cases} \sqrt{2j+1}\, P_j(2x-1), & x \in [0,1], \\ 0, & x \notin [0,1]. \end{cases} \tag{3.6}$$

and note that these functions satisfy the orthonormality conditions

$$\int_0^1 \phi_i(x)\phi_j(x)dx = \delta_{ij}. \tag{3.7}$$

The space $\mathbf{V}_n^k$ is spanned by $2^n k$ functions which are obtained from $\phi_0, \ldots, \phi_{k-1}$ by dilation and translation,

$$\phi_{jl}^n(x) = 2^{n/2}\phi_j(2^n x - l). \tag{3.8}$$

We also introduce an orthonormal basis $\psi_0, \ldots, \psi_{k-1}$ for $\mathbf{W}_0^k$ such that

$$\int_0^1 \psi_i(x)\psi_j(x)dx = \delta_{ij}, \tag{3.9}$$

(here we drop the index k in our notation). Since $\mathbf{W}_0^k \perp \mathbf{V}_0^k$, the first k moments of the basis functions ψ_i vanish,

$$\int_0^1 \psi_i(x)x^m dx = 0, \qquad i, m = 0, 1, \ldots, k-1 \tag{3.10}$$

This basis was constructed in [1] in the form of piecewise polynomial functions.

The space $\mathbf{W}_n^k$ is spanned by $2^n k$ functions obtained from $\psi_0, \ldots, \psi_{k-1}$ by dilation and translation,

$$\psi_{jl}^n(x) = 2^{n/2}\psi_j(2^n x - l), \quad \text{and} \quad \text{supp } \psi_{jl}^n = I_{nl} \tag{3.11}$$

where I_{nl} denotes the interval $[2^{-n}l,\ 2^{-n}(l+1)]$. The condition of orthonormality of the basis functions yields

$$\int_0^1 \psi_{il}^n(x)\psi_{jm}^{n'}(x)dx = \delta_{ij}\delta_{lm}\delta_{nn'}. \tag{3.12}$$

Thus,

$$\begin{aligned} \mathbf{W}_0^k = &\ \text{linear span}\{\psi_j : j = 0, ..., k-1\}, \\ \mathbf{W}_n^k = &\ \text{linear span}\{\psi_{jl}^n : j = 0, ..., k-1; l = 0, ..., 2^n - 1\} \end{aligned} \tag{3.13}$$

In view of (3.4), (3.5) and (3.13) the orthonormal system

$$B_k = \{\phi_j :\ j = 0, \dots, k-1\} \bigcup \{\psi_{jl}^n :\ j = 0, \dots, k-1; n = 0, 1, \dots; l = 0, \dots, 2^n - 1\} \tag{3.14}$$

spans $\mathbf{L}^2[0,1]$. We refer to B_k as the multiwavelet basis of order k for $\mathbf{L}^2[0,1]$.

The relations (3.2) and (3.3) between the subspaces may be expressed via the so-called two-scale difference equations,

$$\phi_i(x) = \sqrt{2} \sum_{j=0}^{k-1} \left(h_{ij}^{(0)} \phi_j(2x) + h_{ij}^{(1)} \phi_j(2x-1) \right), \tag{3.15}$$

$$\psi_i(x) = \sqrt{2} \sum_{j=0}^{k-1} \left(g_{ij}^{(0)} \phi_j(2x) + g_{ij}^{(1)} \phi_j(2x-1) \right), \tag{3.16}$$

The matrices of coefficients $H^{(0)} = \left\{ h_{ij}^{(0)} \right\}, H^{(1)} = \left\{ h_{ij}^{(1)} \right\}, G^{(0)} = \left\{ g_{ij}^{(0)} \right\}, H^{(1)} = \left\{ h_{ij}^{(1)} \right\}$ are analogs of the quadrature mirror filters.

Several first coefficients $h_{ij}^{(0)}$ and $g_{ij}^{(0)}$ are shown in Table 2 and Table 3 for $k = 1, \dots, 4$. Note that the coefficients $g_{ij}^{(0)}$ depend on the choice of the order k.

$\frac{1}{\sqrt{2}}$	0	0	0
$-\frac{\sqrt{3}}{2\sqrt{2}}$	$\frac{1}{2\sqrt{2}}$	0	0
0	$-\frac{\sqrt{3}\sqrt{5}}{4\sqrt{2}}$	$\frac{1}{4\sqrt{2}}$	0
$\frac{\sqrt{7}}{8\sqrt{2}}$	$\frac{\sqrt{3}\sqrt{7}}{8\sqrt{2}}$	$-\frac{\sqrt{5}\sqrt{7}}{8\sqrt{2}}$	$\frac{1}{8\sqrt{2}}$

Table 2: Coefficients $h_{ij}^{(0)}$.

$$\begin{bmatrix} -\frac{1}{\sqrt{2}} \end{bmatrix} \quad \begin{bmatrix} 0 & -\frac{1}{\sqrt{2}} \\ \frac{1}{2\sqrt{2}} & \frac{\sqrt{3}}{2\sqrt{2}} \end{bmatrix} \quad \begin{bmatrix} \frac{1}{3\sqrt{2}} & \frac{1}{\sqrt{6}} & -\frac{5}{3\sqrt{2}} \\ 0 & \frac{1}{4\sqrt{2}} & \frac{\sqrt{15}}{4\sqrt{2}} \\ -\frac{\sqrt{5}}{6\sqrt{2}} & -\frac{5}{2\sqrt{6}} & -\frac{2}{3} \end{bmatrix} \quad \begin{bmatrix} 0 & \frac{\sqrt{2}}{\sqrt{85}} & \frac{\sqrt{6}}{\sqrt{17}} & -\frac{\sqrt{21}}{\sqrt{170}} \\ -\frac{1}{\sqrt{42}} & -\frac{1}{\sqrt{14}} & -\frac{\sqrt{5}}{2\sqrt{42}} & \frac{\sqrt{3}}{2\sqrt{2}} \\ 0 & -\frac{\sqrt{21}}{4\sqrt{170}} & -\frac{3\sqrt{7}}{4\sqrt{34}} & -\frac{4\sqrt{2}}{\sqrt{85}} \\ \frac{5\sqrt{5}}{8\sqrt{42}} & \frac{5\sqrt{5}}{8\sqrt{14}} & \frac{23}{8\sqrt{42}} & \frac{15}{8\sqrt{2}} \end{bmatrix}$$

Table 3. Coefficients $g_{ij}^{(0)}$ for $k = 1, ..., 4$

From the symmetry of the functions $\phi_j(x)$ and $\psi_j(x)$ and the relations (3.15),(3.16) it follows that

$$h_{ij}^{(1)} = (-1)^{i+j} \quad h_{ij}^{(0)}, \tag{3.17}$$
$$g_{ij}^{(1)} = (-1)^{i+j+k} g_{ij}^{(0)}. \tag{3.18}$$

In the subspace $\mathbf{V}_n^k$ the function $f(x)$ is represented by an expansion

$$f(x) = \sum_{l=0}^{2^n-1} \sum_{j=0}^{k-1} s_{jl}^n \phi_{jl}^n(x), \tag{3.19}$$

with the expansion coefficients

$$s_{jl}^n = \int_{2^{-n}l}^{2^{-n}(l+1)} f(x)\phi_{jl}^n(x)dx. \tag{3.20}$$

The decomposition of $f(x)$ into the multiwavelet basis (3.14) reads

$$f(x) = \sum_{j=0}^{k-1} \left(s_{j,0}^0 \phi_j(x) + \sum_{m=0}^{n-1} \sum_{l=0}^{2^m-1} d_{jl}^m \psi_{jl}^m(x) \right), \tag{3.21}$$

where the coefficients d_{jl}^n are computed as follows,

$$d_{jl}^m = \int_{2^{-n}l}^{2^{-n}(l+1)} f(x)\psi_{jl}^m(x)dx. \tag{3.22}$$

This is a collection of $2^n k$ functions from $m = 0, 1, \ldots, n-1$ levels. On the coarsest level, $m = 0$, there are two sets of functions, $\phi_j(x)$ and $\psi_j(x)$, supported on the whole interval $[0, 1]$. On the m-th level, $m \geq 1$, there are $2^m k$ functions $\psi_{jl}^m(x)$, supported on the interval $[2^{-m}l,\ 2^{-m}(l+1)]$.

There is no need to compute integrals (3.22) for all levels $m = 0, 1, ..., n$. In fact, it is sufficient to compute only coefficients s_{jl}^n on the finest level n. In the expression (3.20) we perform rescaling to the interval $[-1, 1]$ using the relations (3.8) and (3.6). We obtain

$$s_{jl}^n = \beta_{nj} \int_{-1}^{1} \tilde{f}^{(l)}(\xi) P_j(\xi) d\xi, \tag{3.23}$$

where $\beta_{nj} = 2^{-(n/2+1)}\sqrt{2j+1}$ is the scaling factor, $P_j(\xi)$ is the jth Legendre polynomial and $\tilde{f}^{(l)}(\xi) = f(x^{(l)})$, $x^{(l)} = 2^{-n}(\xi/2 + l + 1/2)$ so that the interval $\xi = [-1, 1]$ is mapped to $x^{(l)} = [2^{-n}l, 2^{-n}(l+1)]$ (here we omitted the superscript "n", required at $\tilde{f}^{(l)}$ and $x^{(l)}$, for the sake of brevity). The most accurate way of computing the integral in (3.23) is to use the Gauss-Legendre quadrature formula

$$\int_{-1}^{1} \tilde{f}(\xi) P_j(\xi) d\xi = \sum_{i=1}^{k} f(\xi_i) P_j(\xi_i) w_i, \tag{3.24}$$

where $1 < \xi_i < 1$ are the Gauss-Legendre nodes and w_i are the standard Legendre weights.

Thus, given $2^n k$ node values of a function $f(x)$ at the local Gauss-Legendre nodes $x_i^{(l)}$,

$$x_i^{(l)} = \bar{x}_l + \frac{\bar{x}_{l+1} - \bar{x}_l}{2}(\xi_i + 1), \tag{3.25}$$

we compute the expansion coefficients s_{jl}^n using (3.20), (3.23)-(3.25). Then the values d_{jl}^m can be obtained using the relations between the coefficients on two consecutive levels (*decomposition step*),

$$s_{il}^m = \sqrt{2} \sum_{j=0}^{k-1} \left(h_{ij}^{(0)} s_{j,2l}^{m+1} + h_{ij}^{(1)} s_{j,2l+1}^{m+1} \right), \tag{3.26}$$

$$d_{il}^m = \sqrt{2} \sum_{j=0}^{k-1} \left(g_{ij}^{(0)} s_{j,2l}^{m+1} + g_{ij}^{(1)} s_{j,2l+1}^{m+1} \right). \tag{3.27}$$

These relations follow immediately from the two-scale equation, (3.15), (3.16), and the expressions (3.20), (3.22).

The inverse transformation from a coarser to a fine scale reads as follows (*reconstruction step*),

$$s_{i,2l}^{m+1} = \frac{1}{\sqrt{2}} \sum_{j=0}^{k-1} \left(h_{ji}^{(0)} s_{jl}^m + g_{ji}^{(0)} d_{jl}^m \right), \tag{3.28}$$

$$s_{i,2l+1}^{m+1} = \frac{1}{\sqrt{2}} \sum_{j=0}^{k-1} \left(h_{ji}^{(1)} s_{jl}^m + g_{ji}^{(1)} d_{jl}^m \right). \tag{3.29}$$

III.2 Representation of Differential Operators in Multiwavelet Bases: The Case of Constant Coefficients.

Representation of differential operators with constant coefficients in the multiwavelet basis was constructed in [3]. In this section we summarize some results which will be used in further sections. Since we are interested in the solution of the diffusion equation (2.1) we concentrate on the construction of the second derivative operator $\frac{\partial^2}{\partial x^2}$ and the exponential operator $e^{\Delta t \frac{\partial^2}{\partial x^2}}$.

The operator $\mathcal{L} = \frac{\partial^2}{\partial x^2}$ is a homogeneous operator of the second degree, that is $\mathcal{L}(f)(\lambda x) = \lambda^2 \mathcal{L}(f)(x)$. For homogeneous operators it is sufficient to obtain their representations on a the finest scale n (in the $\mathbf{V}_n^k$ subspace). Then representations on all other scales may be obtained by rescaling (see e.g. [5]).

Consider a twice (at least) differentiable function $f(x)$. Project this function and its second derivative onto the subspace $\mathbf{V}_n^k$, that is expand $f(x)$ into the basis $\left\{\phi_{jl}^n(x)\right\}$

$$
\begin{aligned}
f(x) &= \sum_{l=0}^{2^n-1}\sum_{j=0}^{k-1} s_{jl}^n \phi_{jl}^n(x), && (3.30)\\
\frac{\partial^2}{\partial x^2} f(x) &= \sum_{l=0}^{2^n-1}\sum_{j=0}^{k-1} \zeta_{jl}^n \phi_{jl}^n(x). && (3.31)
\end{aligned}
$$

The problem of computing the second derivative thus reduces to finding the transition matrices of coefficients, $[\sigma_{lm}^n]_{ij}$,

$$
\zeta_{il}^n = \sum_{m=0}^{2^n-1}\sum_{j=0}^{k-1} [\sigma_{lm}^n]_{ij}\, s_{jm}^n. \tag{3.32}
$$

Since $\frac{\partial^2}{\partial x^2}$ is a homogeneous operator, it is sufficient to construct its representation $[\sigma_l]_{ij}$ on the coarsest level $n=0$,

$$
[\sigma_l]_{ij} = \int_0^1 \phi_i(x) \frac{\partial^2}{\partial x^2}\phi_j(x+l)dx. \tag{3.33}
$$

Then the representation on the level n can be obtained by rescaling

$$
[\sigma_{lm}^n]_{ij} = 2^{2n}\,[\sigma_{l-m}]_{ij}\,. \tag{3.34}
$$

The matrix $[\sigma_l]$ gives the representation of the operator $\frac{\partial^2}{\partial x^2}$ in $\mathbf{V}_n^k$. We note that for homogeneous operators the matrix elements which represent the complete non-standard form of an operator in the multiwavelet basis, can be expressed in terms of $[\sigma_l]$, see [5].

Since $\frac{\partial^2}{\partial x^2}$ is a local operator, only interactions between neighboring intervals are involved, that is in (3.34) $l-m=0,\pm1$. Therefore, we can rewrite (3.33) as follows

$$
\zeta_{il}^n = 2^{2n}\sum_{j=0}^{k-1}\left([\sigma_1]_{ij}\, s_{j,l-1}^n + [\sigma_0]_{ij}\, s_{jl}^n + [\sigma_{-1}]_{ij}\, s_{j,l+1}^n\right). \tag{3.35}
$$

By introducing notations

$$
Z_l^n = \zeta_{il}^n,\quad S_m^n = s_{jm}^n,\quad \Sigma_{lm}^n = 2^{2n}\,[\sigma_{l-m}]_{ij}\,, \tag{3.36}
$$

we can rewrite (3.35) in the form

$$
Z_l^n = \Sigma_{lm}^n S_m^n. \tag{3.37}
$$

The transition matrix Σ^n_{lm} has a block three-diagonal structure

$$\Sigma^n_{lm} = \begin{pmatrix} . & . & . & . & . & . & . \\ . & [\sigma_1] & [\sigma_0] & [\sigma_{-1}] & . & . & . \\ . & . & [\sigma_1] & [\sigma_0] & [\sigma_{-1}] & . & . \\ . & . & . & [\sigma_1] & [\sigma_0] & [\sigma_{-1}] & . \\ . & . & . & . & . & . & . \end{pmatrix}_{N\times N} \tag{3.38}$$

where $N = 2^n$ and each block $[\sigma_l]$ is a $k \times k$ matrix. The matrix blocks $[\sigma_1]_{ij}$ and $[\sigma_{-1}]_{ij}$ describe interactions with the left and the right neighboring intervals respectively. Such an interaction is necessary to insure the CFL condition when a time-dependent problem is under consideration.

Follow the expressions for the matrices $[\sigma_l]_{ij}$:

$$\begin{aligned} [\sigma_{-1}] &= \beta_I\,[\sigma_{-1}]^I + \beta_{II}\,[\sigma_{-1}]^{II}, \\ [\sigma_1] &= \alpha_I\,[\sigma_1]^I + \alpha_{II}\,[\sigma_1]^{II}, \\ [\sigma_0] &= [\sigma_0]^0 + \beta_I\,[\sigma_0]^I + \beta_{II}\,[\sigma_0]^{II} + \alpha_I\,[\sigma_0]^{III} + \alpha_{II}\,[\sigma_0]^{IV}, \end{aligned} \tag{3.39}$$

where

$$\begin{aligned} [\sigma_{-1}]^I_{ij} &= (-1)^j\Gamma_{ij}\left(\Omega_i - \Omega_j\right), & [\sigma_{-1}]^{II}_{ij} &= (-1)^{j+1}\Gamma_{ij}\Omega_j, \\ [\sigma_1]^I_{ij} &= (-1)^{i+1}\Gamma_{ij}\left(\Omega_i - \Omega_j\right), & [\sigma_1]^{II}_{ij} &= (-1)^{i+1}\Gamma_{ij}\Omega_j, \\ [\sigma_0]^I_{ij} &= -\Gamma_{ij}\left(\Omega_i + \Omega_j\right), & [\sigma_0]^{II}_{ij} &= -\Gamma_{ij}\Omega_j, \\ [\sigma_0]^{III}_{ij} &= (-1)^{i+j}\Gamma_{ij}\left(\Omega_i + \Omega_j\right), & [\sigma_0]^{IV}_{ij} &= (-1)^{i+j+1}\Gamma_{ij}\Omega_j, \end{aligned} \tag{3.40}$$

$$[\sigma_0]^0_{ij} = \Gamma_{ij}\left(\Omega_j - \Omega_i\right)\tilde{\mu}_{ij}, \qquad \tilde{\mu}_{ij} = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & . \\ 0 & 0 & 0 & 1 & 0 & . \\ 0 & 0 & 0 & 0 & 1 & . \\ 0 & 0 & 0 & 0 & 0 & . \\ 0 & 0 & 0 & 0 & 0 & . \\ . & . & . & . & . & . \end{pmatrix} \tag{3.41}$$

and

$$\Gamma_{ij} = 2\sqrt{2i+1}\sqrt{2j+1}, \qquad \Omega_i = j(j+1).$$

The representation of the second derivative contains four free parameters: β_I, β_{II}, α_I and α_{II}. These parameters are related to the interaction with neighboring intervals. If we put them to be zero then only the matrix $[\sigma_0]^0$ in (3.39) will remain, which represents the second derivative operator in the Legendre polynomial basis.

The construction of functions of operators in the multiwavelet basis, in particular the exponential $e^{\Delta t \frac{\partial^2}{\partial x^2}}$, is discussed in [3] in the case of constant coefficient operators.

III.3 Representation of Differential Operators in the Multiwavelet Basis: The Case of Variable Coefficients

Our goal is to construct a representation of the non-constant coefficient operator $\mathcal{L} = a(x)\frac{\partial^2}{\partial x^2}$ in the multiwavelet basis. Then the exponential operators $Q_0 = e^{\Delta t \mathcal{L}}$, $Q_1 = \left(e^{\Delta t \mathcal{L}} - \mathcal{I}\right)(\Delta t \mathcal{L})^{-1}$ etc., can be computed via the generalized scaling and squaring method described in section II.3. It will be shown below that although the operators Q_j are expressed in terms of the local operator $\mathcal{L}$ they are represented by global matrices in accordance with the global nature of these operators.

In this paper we concentrate on the construction of operators in the $\mathbf{V}_n^k$ subspace (that is on the finest scale) to illustrate the main traits of our approach. We note that since the scaling functions for the multiwavelet basis are the (scaled and shifted) Legendre polynomials, the multiwavelet method on the finest scale $\mathbf{V}_n^k$ is equivalent to the multidomain Legendre method.

The representation on the finest scale is an essential element for constructing the multiresolution representation of operators in the non-standard form. This is because the matrices representing an operator on coarser scales are expressed in terms of the matrices on the finest scale [5]. Unlike the constant coefficient operator $\frac{\partial^2}{\partial x^2}$, the operator $a(x)\frac{\partial^2}{\partial x^2}$ is not homogeneous and thus the scaling property (3.34) for the representations on different scales does not take place. Therefore, the corresponding matrices have to be computed on each scale independently. This will be a subject of our future work.

We emphasize here again that the motivation for having a multiresolution representation of operators is that the corresponding matrices are sparse, unlike their representations on the finest scale which are typically dense (or even full for the global operators like Q_j).

Consider a twice differentiable function $f(x)$. In order to represent the functions $f(x)$ and $\frac{\partial^2}{\partial x^2}f(x)$ in the subspace $\mathbf{V}_n^k$, we expand them into the basis $\left\{\phi_{jl}^n(x)\right\}$

$$\begin{aligned} f(x) &= \sum_{l=0}^{2^n-1}\sum_{j=0}^{k-1} s_{jl}^n \phi_{jl}^n(x), && (3.42)\\ \frac{\partial^2}{\partial x^2} f(x) &= \sum_{l=0}^{2^n-1}\sum_{j=0}^{k-1} \zeta_{jl}^n \phi_{jl}^n(x). && (3.43)\end{aligned}$$

Similarly, the functions $a(x)$ and $a(x)\frac{\partial^2}{\partial x^2}f(x)$ are represented in $\mathbf{V}_n^k$ by the expansions

$$\begin{aligned} a(x) &= \sum_{l=0}^{2^n-1}\sum_{j=0}^{k-1} \alpha_{jl}^n \phi_{jl}^n(x), && (3.44)\\ a(x)\frac{\partial^2}{\partial x^2} f(x) &= \sum_{l=0}^{2^n-1}\sum_{j=0}^{k-1} \tilde{\zeta}_{jl}^n \phi_{jl}^n(x). && (3.45)\end{aligned}$$

The operator $\mathcal{L} = a(x)\frac{\partial^2}{\partial x^2}$ is represented in the subspace $\mathbf{V}_n^k$ by the transition matrix, $[r_{lm}^n]$, which connects the expansion coefficients of $f(x)$ and those of $a(x)\frac{\partial^2}{\partial x^2}f(x)$,

$$\tilde{\zeta}_{il}^n = \sum_{m=0}^{2^n-1}\sum_{j=0}^{k-1}[r_{lm}^n]_{ij}\, s_{jm}^n. \tag{3.46}$$

Projecting both sides of Eq. (3.45) onto $\phi_{il}^n(x)$ and using (3.44), (3.43) we obtain

$$\begin{aligned}
\tilde{\zeta}_{il}^n &= \sum_{m,m'=0}^{2^n-1}\sum_{j,j'=0}^{k-1}\zeta_{jm}^n\alpha_{j'm'}^n\int_{2^{-n}l}^{2^{-n}(l+1)}\phi_{il}^n(x)\phi_{jm}^n(x)\phi_{j'm'}^n(x)\;dx \\
&= 2^{n/2}\sum_{m,m'=0}^{2^n-1}\sum_{j,j'=0}^{k-1}\zeta_{jm}^n\alpha_{j'm'}^n\delta_{lm}\delta_{lm'}\int_0^1\phi_i(x)\phi_j(x)\phi_{j'}(x)dx \\
&= 2^{n/2}\sum_{j,j'=0}^{k-1}t_{ijj'}\zeta_{jl}^n\alpha_{j'l}^n
\end{aligned} \tag{3.47}$$

where we defined a triple matrix

$$t_{ijj'} = \int_0^1\phi_i(x)\phi_j(x)\phi_{j'}(x)dx. \tag{3.48}$$

Introducing a block-diagonal matrix $[g_l^n]$

$$[g_l^n]_{ij} = \sum_{j'=0}^{k-1}t_{ijj'}\alpha_{j'l}^n. \tag{3.49}$$

and using (3.32), (3.34) and (3.35) we can rewrite (3.47) as follows

$$\begin{aligned}
\tilde{\zeta}_{il}^n &= 2^{n/2}\sum_{j=0}^{k-1}[g_l^n]_{ij}\,\zeta_{jl}^n \\
&= 2^{n/2}\sum_{j,j'=0}^{k-1}\sum_{m=0}^{2^n-1}[g_l^n]_{ij'}\,[\sigma_{lm}^n]_{j'j}\,s_{jl}^n \\
&= 2^{5n/2}\sum_{j,j'=0}^{k-1}\sum_{m=-1}^{1}[g_l^n]_{ij'}\,[\sigma_{l-m}]_{j'j}\,s_{jl}^n.
\end{aligned} \tag{3.50}$$

Thus, the sought transition matrix is

$$[r_{lm}^n]_{ij} = 2^{5n/2}\sum_{j'=0}^{k-1}[g_l^n]_{ij'}\,[\sigma_{l-m}]_{j'j}\,, \qquad m = 0, \pm 1. \tag{3.51}$$

We note that since the matrix $[g_l^n]$ is block-diagonal, the transition matrix $[r_{lm}^n]$ has the same block three-diagonal structure as the matrix $[\sigma_{lm}^n]$ in (3.38).

The exponential of the operator, $e^{\Delta t_1 \mathcal{L}}$, is represented in the subspace $\mathbf{V}_n^k$ by a dense $\bar{N} \times \bar{N}$ matrix, where $\bar{N} = 2^n k = Nk$. Indeed, the approximation (2.13) of the exponential using Taylor expansion, projected onto $\mathbf{V}_n^k$, reads

$$[E_{lm}^n]_{ij} = [\delta_{lm}^n]_{ij} + \Delta t_1 \, [r_{lm}^n]_{ij} \qquad l, m = 1, ..., 2^n, \quad i, j = 1, ..., k \tag{3.52}$$

where $[E_{lm}^n]$ stands for the matrix representation of the operator $\mathrm{Q}_0(\Delta t_1 \mathcal{L}) = e^{\Delta t_1 \mathcal{L}}$ in $\mathbf{V}_n^k$, $[r_{lm}^n]$ represents the operator $\mathcal{L} = \frac{\partial^2}{\partial x^2}$, and $[\delta_{lm}^n]$ is a notation for the $\bar{N} \times \bar{N}$ identity matrix. Since the matrix $[r_{lm}^n]$ is a block three-diagonal, the same is true for the matrix $[E_{lm}^n]$. To compute the exponential operator $\mathrm{Q}_0(\Delta t \mathcal{L})$, where $\Delta t = 2^J \Delta t_1$, the matrix $[E_{lm}^n]$ has to be squared J times. Each squaring of a block three-diagonal matrix results in a matrix with more non-zero blocks. Evidently, for the number of squaring steps J large enough, the matrix $[E_{lm}^n]^{2^J}$ will be dense, in agreement with the global nature of the exponential operator $e^{\Delta t \frac{\partial^2}{\partial x^2}}$.

When solving a time dependent problem (2.1) with the linear operator $\mathcal{L}(x) = a(x) \frac{\partial^2}{\partial x^2}$, the matrix $[E_{lm}^n]^{2^J}$ can be precomputed and stored since the variable coefficient $a(x)$ is assumed to be independent of time. This precomputational step requires $O(J \bar{N}^3)$ operations (multiplication of two $\bar{N} \times \bar{N}$ matrices J times). At each time step a $\bar{N} \times \bar{N}$ matrix is applied to a $\bar{N}$- length vector. This requires $O(\bar{N}^2)$ operations per time step. A far more efficient approach for computing the differential operators and functions of operators in the multiwavelet basis is discussed in section V.

We notice that the above constructions of the operators $\mathcal{L}$ and $e^{\Delta t \frac{\partial^2}{\partial x^2}}$ does not incorporate the boundary conditions. Thus, if the function $u(x)$ satisfies at the beginning some boundary conditions, the successive application of the operator $e^{\Delta t \frac{\partial^2}{\partial x^2}}$ will not preserve these conditions. We will address this point in section IV.

The following numerical test illustrates the above algorithm.

Example 1. Linear heat equation

$$u_t(x) = a(x) u_{xx}, \qquad x \in [0, 1] \tag{3.53}$$

with the Dirichlet boundary conditions

$$u(0, t) = g_1(t), \quad u(1, t) = g_2(t). \tag{3.54}$$

Equation (3.53) has the exact solution

$$u^{ref}(x, t) = e^{-\mu t} e^{-x^2/2} H_\omega(x), \tag{3.55}$$

where $H_\omega(x)$ is the $\omega-$th Hankel polynomial. This solution corresponds to the variable coefficient $a(x) = \mu / (2\omega + 1 - x^2)$ in (3.53).

The time stepping algorithm (2.5) for the linear equation (3.53) reduces to the form

$$u_{n+1} = e^{\Delta t \mathcal{L}} u_n, \qquad \mathcal{L} = a(x) \frac{\partial^2}{\partial x^2}. \tag{3.56}$$

The numerical results below are obtained for $\mu = 1$, $\omega = 4$, $H_4(x) = 16x^4 - 48x^2 + 12$.

In Fig. 1 we plot the pointwise error at the time $t = 2.09 \times 10^{-2}$ for $\Delta t = 10^{-6} \times 2^7$, $N = 32, k = 5$; the parameters of the second derivative operator in (3.39) are: $\alpha_I = -0.6$, $\beta_I = 0.6$, $\alpha_{II} = \beta_{II} = 0.5$ (these parameters are found to be the optimal ones, that is they provide the best accuracy for the computed second derivative). Since the numerical procedure, as it is described so far, does not provide any explicit treatment of the boundary conditions, the error on the boundaries grows and propagates into the region as time evolves.

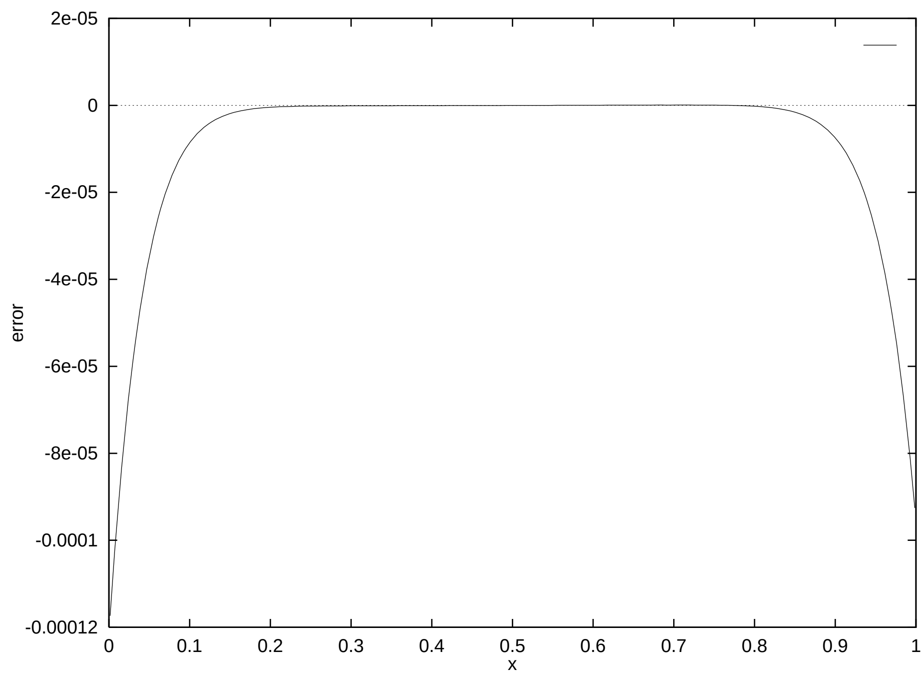

Figure 1: Pointwise error at $t = 2.09 \times 10^{-2}$. The error propagates from the boundaries into the region.

IV Penalty Procedure on the Boundaries

To enforce the boundary conditions (3.54) we implement a penalty procedure suggested in [12], [11]. The penalty term is introduced as a forcing in the evolution equation

$$u_t = a(x) u_{xx} + f_B(x,t), \qquad x \in [0,1], \tag{4.1}$$

where the function $f_B(x,t)$ is defined as

$$f_B(x,t) = -\tau_1 Q^-(x)\left[u(0,t) - g_1(t)\right] - \tau_2 Q^+(x)\left[u(1,t) - g_2(t)\right]. \tag{4.2}$$

$Q^-(x)$ and $Q^+(x)$ may be defined as delta-functions at the left and the right boundaries respectively. However, for numerical implementation it is convenient to define them as continuous functions. For example,

$$Q^-(x) = e^{-\lambda x}, \qquad Q^+(x) = e^{-\lambda(1-x)}. \tag{4.3}$$

The parameter λ has to be large enough so that $Q^\pm(x)$ are well localized near the boundaries.

The amplitudes $\tau_1, \tau_2 > 0$ are positive constant parameters. The sign in (4.2) is chosen so that if the deviation, say, on the left boundary, $u(0,t) - g_1(t)$, is positive, then the sign of the forcing term (and the concomitant time derivative $\partial u/\partial t$) is negative, and vice versa. Similarly on the right boundary. Thus, the forcing term strives to recover the prescribed boundary values. The coefficients τ_1, τ_2 must be large enough to stabilize the time marching procedure. However, when they are too large the computation becomes unstable (see next section). In the examples below $\tau_1 = \tau_2 = \tau_b$.

More general boundary conditions can be treated in a similar way using the above penalty procedure. Namely, if the boundary conditions are given in the form

$$\begin{aligned} \alpha u(0,t) - \beta u_x(0,t) - g_1(t) &= 0, \\ \gamma u(1,t) - \delta u_x(1,t) - g_2(t) &= 0, \end{aligned} \tag{4.4}$$

then in (4.2) the expressions in the brackets have be replaced by those in the left hand sides of (4.4).

IV.1 Numerical Test with Penalty Procedure on the Boundaries

Now we turn back to the Example 1 and compute the solution using the penalty procedure as described in section IV. For the time discretization we use the second order explicit ELP scheme described in sections II.2 where $\mathcal{N} = f_B(x,t)$, along with the scaling and squaring method of section II.3 for computing the quadrature coefficients $\mathrm{Q}_j(\Delta t\mathcal{L})$. The space discretization is performed via the method of section III.3.

The pointwise error in the numerical solution is plotted in Fig.2 for $\tau_b = 25$, $\Delta t = (8. \times 10^{-7}) \times 2^7$, $N = 32$ (number of subintervals), $k = 5$ (order of multiwavelets), $\alpha_I = -0.6$, $\beta_I = 0.6$, $\alpha_{II} = \beta_{II} = 0.5$ (parameters of the second derivative operator) at two instances: $t_1 = 2.09 \times 10^{-2}$ and $t_2 = 4.1 \times 10^{-2}$. In this case the amplitude of the penalty term τ_b is too small so that the error loosely “floats” in a wide range: when

$u(0,t)$ and $u(1,t)$ in (3.10) deviate too far away from their prescribed positions, $g_1(t)$ and $g_2(t)$, the forcing returns them back.

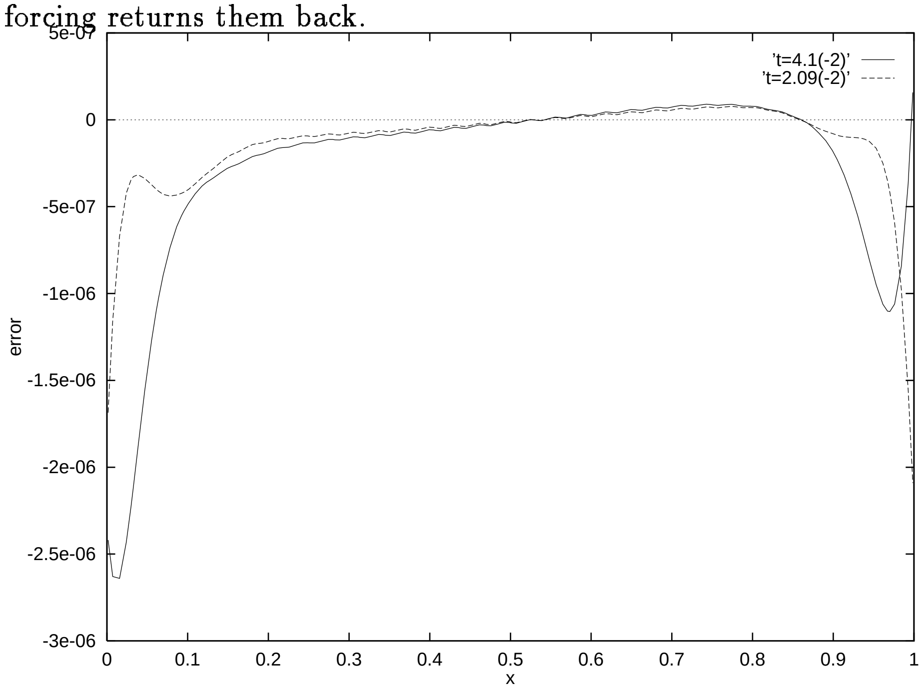

Figure 2: Pointwise error when the penalty term in (4.1) has a small amplitude, $\tau_b = 25$.

In Fig. 3 the pointwise error is plotted for $\tau_b = 10^4$, $\Delta t = (8. \times 10^{-7}) \times 2^7$, $N = 32$, $k = 5$ and at $t = 2.9 \times 10^{-2}$. In this case the amplitude of the penalty term is high enough to restrain the error on the boundaries within a small limit, $\mid$ error $\mid < 1.\ 10^{-9}$.

However, for a given set of parameters τ_b cannot exceed some critical value, otherwise the computation becomes unstable. Fig. 4 demonstrates the region of stability on the plane of parameters $(\tau_b, \Delta t)$ for $N = 32, k = 5$. The stable time marching corresponds to the part of region below the curve. For $t > 10^{-5}$ the time marching is unstable due to CFL restriction on the time step Δt at the given resolution $\bar{N} = Nk = 160$.

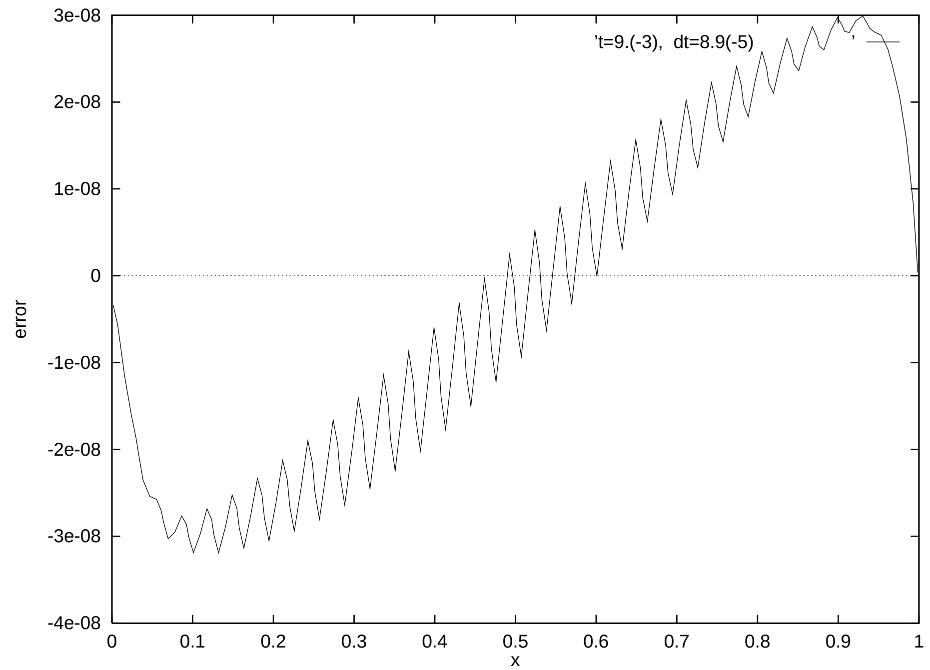

Figure 3: Pointwise error for a large penalty term, $\tau_b = 10^4$.

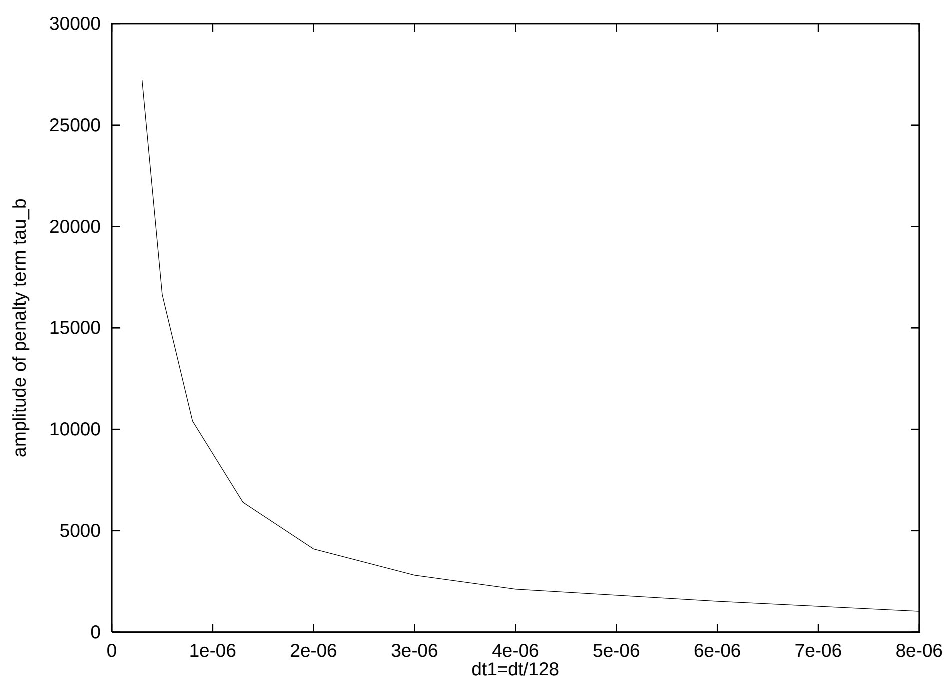

Figure 4: Stability diagram on the plane of parameters $(\tau_b, \Delta t/2^J)$

Although the error on the boundaries remains small due to implementation of the penalty procedure, inside the region it grows slowly with time. The maximal relative error is plotted in Fig. 5 for $\Delta t = (8. \times 10^{-7}) \times 2^7$, $k = 5$, $N = 16$ (solid line). The reason for such a growth of the error is that the parameter μ in (3.55) (the rate of decay), is slightly different for the numerical and exact solutions. As a result, both solutions slowly deviate from one another with time. The boundary conditions are fixed by the penalty procedure so that the error on the boundaries (dashed line) remains small.

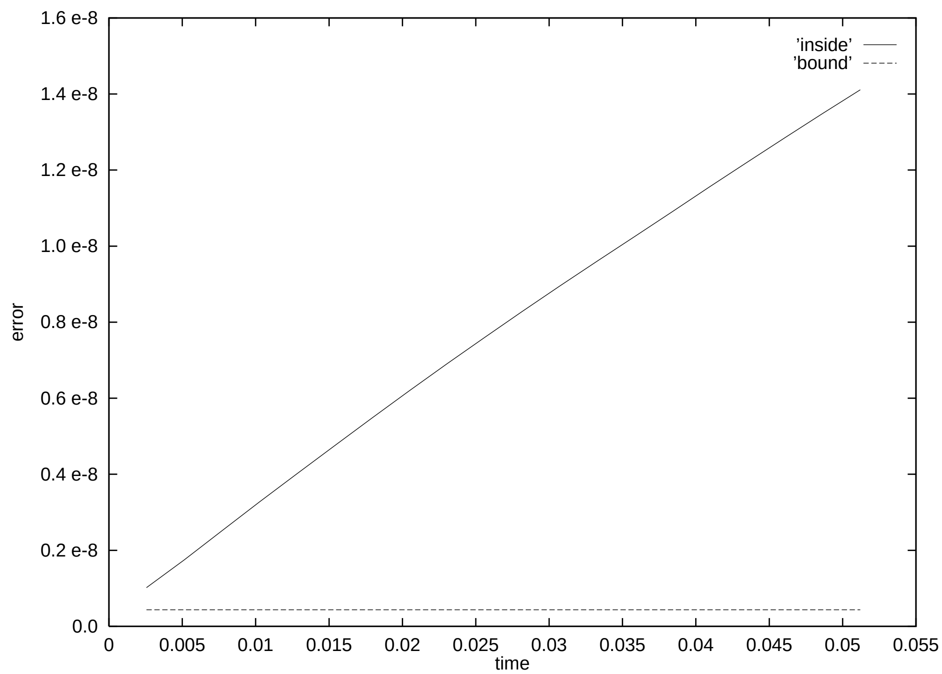

Figure 5: Evolution of the maximum relative error inside the interval (solid line) and on the boundaries (dashed line) for example (5.1).

In order to convince ourselves that the present algorithm provides a stable long-time integration, we considered the solution of the linear heat equation driven by the forcing $f(x,t)$:

$$u_t = a(x)u_{xx} + f_(x,t), \quad x \in [0,1], \tag{4.5}$$

where

$$\begin{aligned} u^{ref}(x,t) &= \cos 2\pi t \cos 2\pi(x - 0.12), \\ a(x) &= x, \\ f(x,t) &= f_B(x,t) + u_t^{ref}(x,t) - a(x)\, u_{xx}^{ref}(x,t), \end{aligned} \tag{4.6}$$

and f_B is the penalty term defined in (4.2). The error oscillates within a small range of values $1.5 \times 10^{-8} <$ error $< 6.5 \times 10^{-8}$.

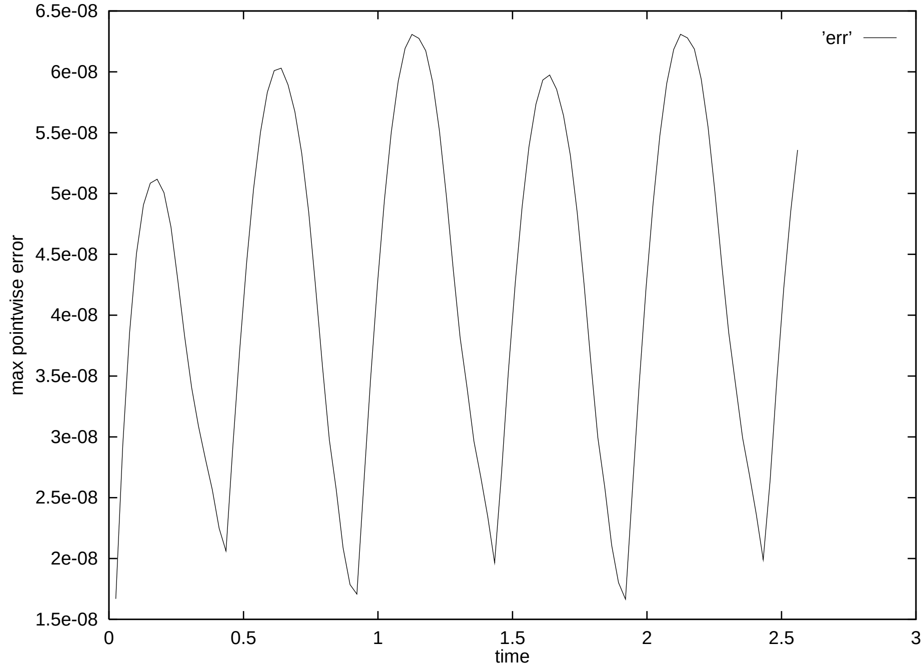

Figure 6: Evolution of the maximum error for the solution (4.6) generated by forcing, number of time steps is about 26,000.

V Penalty Procedure on the Interfaces

As we pointed out in section III.2, the second derivative operator is represented in the subspace $\mathbf{V}_n^k$ by a block three-diagonal matrix. The exponential operator, $e^{\Delta t \mathcal{L}}$, $\mathcal{L} = a(x)\frac{\partial^2}{\partial x^2}$, computed via the scale and squaring method, is represented in $\mathbf{V}_n^k$ by a dense matrix. The computation of such a matrix requires $O(JN^3k^3)$ operations, where J is the number of squaring step, $N = 2^n$ is the number of subintervals and k is the order of multiwavelets. When a time-dependent problem is solved, making a time step in the $\mathbf{V}_n^k$ space requires $O(N^2k^2)$ operations.

In this section we describe a much more economical way of constructing the second derivative operator and the exponential operator $e^{\Delta t \mathcal{L}}$ in the subspace $\mathbf{V}_n^k$. Such a construction results in an algorithm whose computational complexity is proportional to the number of subdomains N. The saving in computational complexity (operation count and memory storage) is especially noticeable for two dimensional problems, see section VI.

We recall that the full matrix, representing the exponential operator $e^{\Delta t \mathcal{L}}$ in $\mathbf{V}_n^k$, results due to repeating squaring of the block three-diagonal matrix (3.38) for the second derivative operator $\frac{\partial^2}{\partial x^2}$. As was mentioned in section (III.2), the off-diagonal blocks $[\sigma_{-1}]$ and $[\sigma_1]$ are responsible for the interaction with the neighboring intervals. Such an interaction is necessary to maintain the continuity of the solution and its first derivative on the interfaces when solving a time-dependent problem.

The idea underlying our new approach consists in treating the interface conditions as a separate procedure, without incorporating them into operators. Thus, we consider the second derivative operator in the *block-diagonal* form

$$\hat{\Sigma}^n_{lm} = \begin{pmatrix} . & . & . & . & . \\ . & [\sigma_0] & . & . & . \\ . & . & [\sigma_0] & . & . \\ . & . & . & [\sigma_0] & . \\ . & . & . & . & . \end{pmatrix}_{N\times N} \tag{5.1}$$

The crucial advantage of this form over (3.38) is that squaring of such a matrix keeps its block-diagonal structure. As a result, the matrix of the exponential operator in the subspace $\mathbf{V}^k_n$ is also block-diagonal

$$\left[\hat{E}^n_{lm}\right] = \delta_{lm}\left[E_l\right], \tag{5.2}$$

where

$$[E_l] = [e_l]^{2^J}, \qquad [e_l]_{ij} = \delta^n_{ij} + \Delta t\,[r^n_l]_{ij}, \qquad [r^n_l]_{ij} = 2^{5n/2}\sum_{j'=0}^{k-1}[g^n_l]_{ij'}\,[\sigma_0]_{j'j} \tag{5.3}$$

and δ^n_{ij} is the $k\times k$ identity matrix (compare with the corresponding formulas (3.51) and (3.52) when using the previous block three-diagonal form of the second derivative matrix Σ^n_{lm}). The computation of the matrix $\left[\hat{E}^n_{lm}\right]$ in (5.2)- (5.3) is accomplished in $O(JNk^3)$ operations, the application of this matrix to a Nk- length vector requires $O(Nk^2)$ operations.

If we apply this operator at each time step when performing the time integration, the discontinuity in the solution and its first derivative will appear and grow rapidly at the interface points $\bar{x}_l = 2^{-n}l,\ l = 0,...,2^n-1$ due to the lack of interaction between the neighboring intervals. In order to preserve the continuity of the solution we apply a penalty procedure on the interfaces similar to that described in section IV for the boundaries. Specifically, we introduce the forcing term in the form

$$f(x,t) = f_B(x,t) + f_I(x,t). \tag{5.4}$$

Here $f_B(x,t)$ is the boundary term defined in (4.2). The interface term $f_I(x,t)$ is defined as follows

$$f_I(x,t) = \sum_{l=0}^{2^n-1}\left(A_l e^{-\lambda(x-\bar{x}_l)} + B_l e^{-\lambda(\bar{x}_{l+1}-x)}\right) \tag{5.5}$$

where

$$\begin{aligned}
A_0 &= B_{2^n-1} = 0, \\
A_l &= -\tau_f\left(u^{(l)}(\bar{x}_l) - \bar{u}_l\right) + \tau_d\left(\frac{du^{(l)}(\bar{x}_l)}{dx} - \bar{d}_l\right), \qquad l \neq 0, \\
B_l &= -\tau_f\left(u^{(l)}(\bar{x}_{l+1}) - \bar{u}_{l+1}\right) - \tau_d\left(\frac{du^{(l)}(\bar{x}_{l+1})}{dx} - \bar{d}_{l+1}\right), \; l \neq 2^n - 1. \qquad (5.6)
\end{aligned}$$

Here $\tau_f > 0, \tau_d > 0$ are the constant quantities. $u^{(l)}(x)$ is a restriction to $u(x)$ on the $l-$th interval

$$u^{(l)}(x) = \begin{cases} u(x), & x \in [\bar{x}_l, \bar{x}_{l+1}] \\ 0, & \text{otherwise.} \end{cases}$$

$\bar{u}_l$ and $\bar{d}_l$ are the mean values of the function $u(x)$ and its first derivative $\frac{d}{dx}u(x)$ on the interface $x = \bar{x}_l$, that is

$$\bar{u}_l = \frac{1}{2}\left[u^{(l)}(x) + u^{(l-1)}(x)\right]_{x=\bar{x}_l}, \qquad \bar{u}_{l+1} = \frac{1}{2}\left[u^{(l)}(x) + u^{(l+1)}(x)\right]_{x=\bar{x}_{l+1}}, \qquad (5.7)$$

$$\bar{d}_l = \frac{1}{2}\left[\frac{du^{(l)}(x)}{dx} + \frac{du^{(l-1)}(x)}{dx}\right]_{x=\bar{x}_l}, \qquad \bar{d}_{l+1} = \frac{1}{2}\left[\frac{du^{(l)}(x)}{dx} + \frac{du^{(l+1)}(x)}{dx}\right]_{x=\bar{x}_{l+1}}. \qquad (5.8)$$

The structure of the forcing is shown in Fig. 7. In each subinterval there exist two exponential functions attached to the left and the right interior boundaries (interfaces) and decaying away from the boundaries with the rate λ.

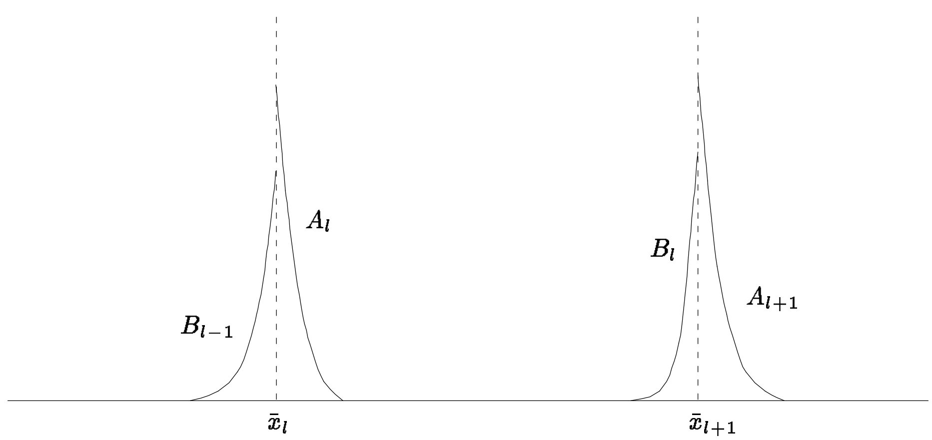

Figure 7: The structure of the forcing term f_I.

The amplitudes A_l and B_l of the exponentials are proportional to the jumps in the solution and its first derivative across the interfaces. Since $\partial u/\partial t \propto f(x,t)$, deviations in the solution (from their mean values at the interfaces) appear with a negative sign so that the forcing suppresses these deviations. The selection of a sign at the terms that contain jumps in the derivative (the terms in (5.6) proportional to τ_d) is not so obvious and requires more consideration.

Suppose that the solution is distorted as shown in Fig. 8 so that the first derivative has jumps on the interfaces.

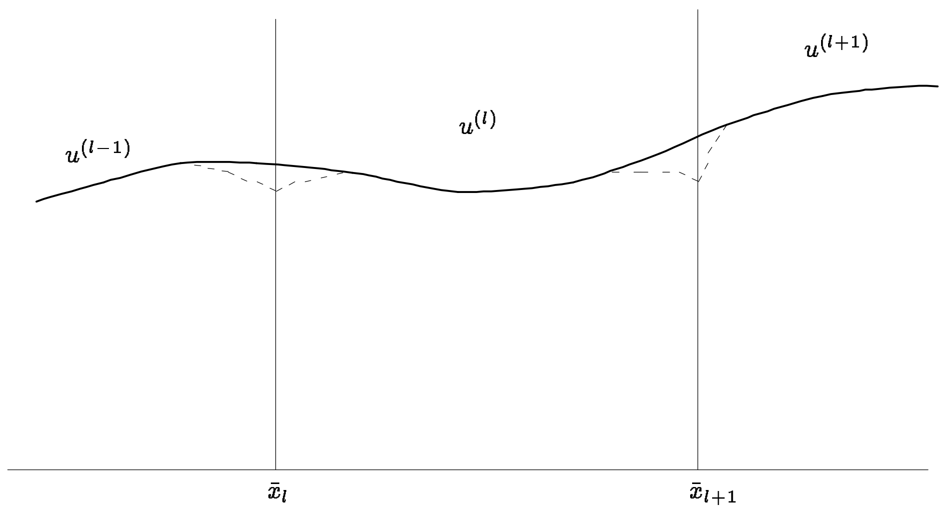

Figure 8: Smooth solution (solid line) and the disturbed solution (dashed line) having a discontinuous first derivative on the interfaces.

It is easy to see that in this case $du^{(l-1)}/dx < du^{(l)}/dx$ at $x = \bar{x}_l$, and $du^{(l)}/dx < du^{(l+1)}/dx$ at $x = \bar{x}_{l+1}$. Therefore, the sign of the deviations on the interfaces is $\left(du^{(l)}(\bar{x}_l)/dx - \bar{d}_l\right) > 0$ and $\left(du^{(l)}(\bar{x}_{l+1})/dx - \bar{d}_{l+1}\right) < 0$. On the other hand, the sign of the forcing must be positive for both interfaces in order to restore the solution to the smooth shape. This dictates the proper choice of signs at the corresponding forcing terms in (5.6). Note that the forcing term in (5.5) is given in the physical domain. The transformation into the $\mathbf{V}_n^k$ subspace (the Legendre coefficients domain) can be obtained via (3.20), (3.23)-(3.25).

V.1 Examples

We illustrate the above approach using the block-diagonal matrices for the second derivative and the exponential operators, along with the penalty procedure on the boundaries and interfaces, by the following examples.

Example 2. Linear heat equation with the forcing term (4.5). The solution of reference

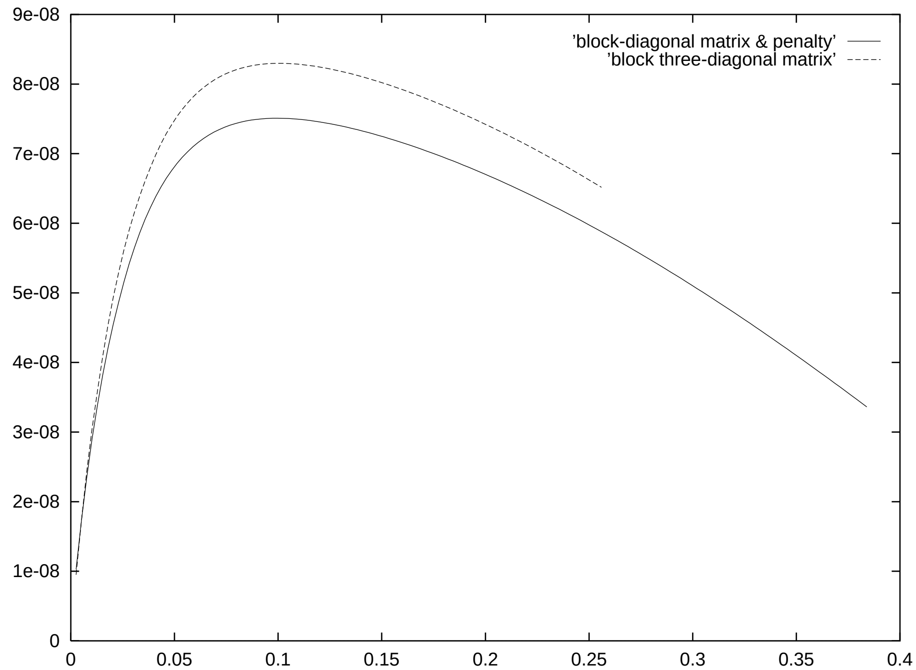

Figure 9: The maximal numerical error for the example (5.9) when using Approach I (dashed line) and Approach II (solid line).

is given by

$$u^{ref}(x,t) = \cos \pi t \cos \pi (x - 0.12), \tag{5.9}$$

the variable coefficient $a(x) = x$. The driving force $f(x,t)$ and the boundary conditions $u(0,t) = g_1(t)$, $u(1,t) = g_2(t)$ are computed accordingly. In time we use the $2d$ order explicit ELP scheme.

The maximum pointwise error as a function of time is plotted in Fig. 9 (solid line) for the parameters: $N = 32$, $k = 5$, $\Delta t = 1.10^{-7} \times 2^7$, $\tau_b = \tau_f = 40000$, $\tau_d = 400$, $\lambda = 6$. The dashed line gives the error obtained by the previous approach (Approach I) based on the use of a block three-diagonal matrix for the second derivative operator (with the parameters $\alpha_I = -0.6$, $\beta_I = 0.6$, $\alpha_{II} = \beta_{II} = 0.5$) and correspondingly the dense matrix $[E^n_{lm}]$ in (3.52) for the exponential operator $e^{\Delta t \mathcal{L}}$. Both approaches give approximately the same numerical errors. However, the latter approach, using the penalty procedure on the interfaces (Approach II), is faster by a factor of $O(N^2)$ as compared to Approach I when computing the exponential $[E^n_{lm}]$ in the subspace $\mathbf{V}^k_n$, and by a factor of $O(N)$ per time step when performing time integration.

Example 3. Burgers equation. Consider the periodic solutions of the Burgers equation

$$u_t = \nu u_{xx} - u u_x, \qquad x \in [0,1] \tag{5.10}$$

The exact solution is given by the formula (Whitham, 1974)

$$u^{(ref)} = -2\nu \frac{\phi_x(x - ct, t + \tau)}{\phi(x - ct, t + \tau)}, \quad \tau > 0 \tag{5.11}$$

$$\phi(x,t) = \sum_{n=-\infty}^{\infty} e^{-(x-n)^2/4\nu t} \tag{5.12}$$

Figure 10 shows the numerical solution for $c = 4$, $\nu = 0.1/\pi$ and $\tau = 1/(2\pi)$ (these are parameters of the standard test case). The profile moves at the speed $c = 4$.

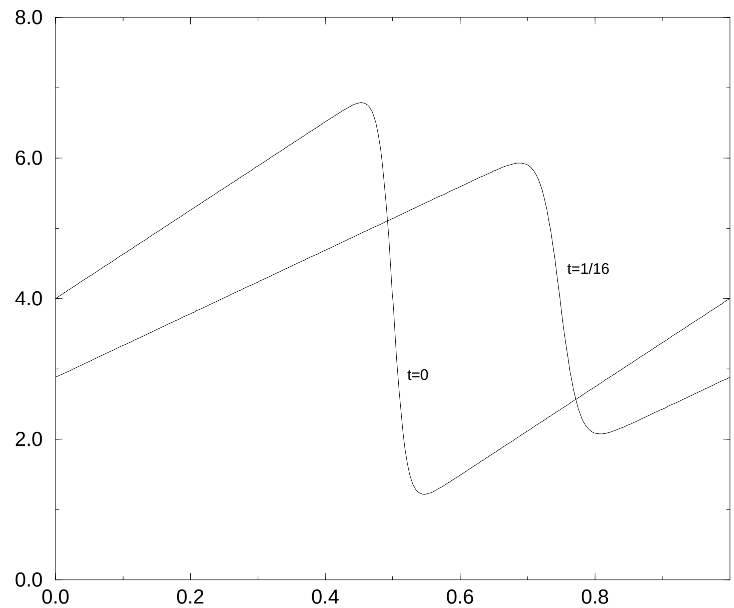

Figure 10: Solution of the periodic Burgers equation at $t = 0$ and $t = 1/16$; $\nu = 0.1/\pi$, $c = 4$.

The pointwise numerical error for the solution at $t = 1/16$ is plotted in Figure 11.

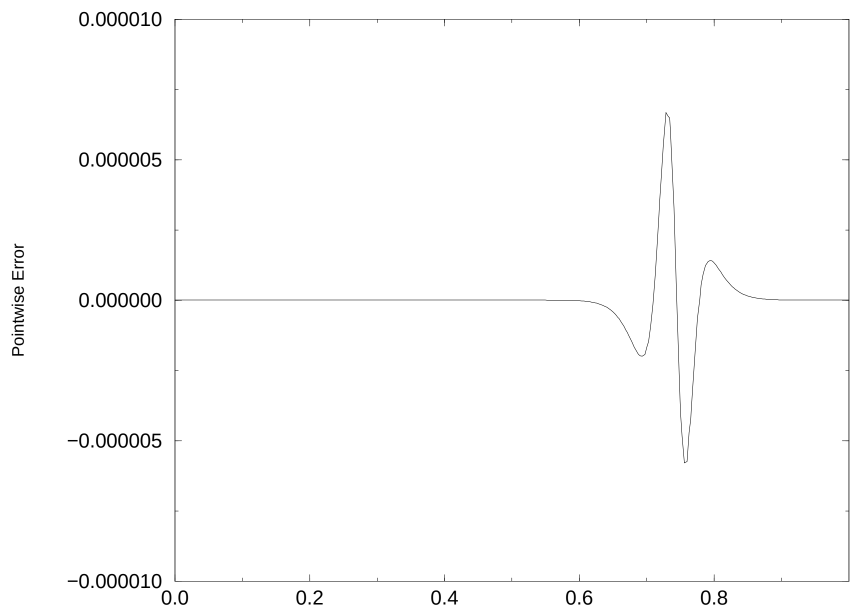

Figure 11: Pointwise error for the solution of periodic Burgers equation at $t = 1/16$.

The spatial resolution and the parameters of the penalty term are chosen as in the previous example. For such a choice the temporal errors are dominant over the spatial ones. The maximum numerical error is given in Table 3 for the explicit first, second and third order ELP schemes for $\Delta t = 10^{-4}$, $t = 1/16$ and $c = 0, c = 4$.

c	ϵ_1	ϵ_2	ϵ_3
0	9.6 (-4)	3.6 (-7)	5.9 (-10)
4	3.0 (-2)	4.2 (-4)	6.5 (-6)

Table 3: Maximal numerical error in the solution of the periodic Burgers equation at the time $t = 1/16$; $\nu = 0.1/\pi$, $c = 0$ and 4.

We conclude this section by making a remark on the choice of parameters of the penalty procedure that provides stable time integration. Since a rigorous stability analysis is not available at the moment, these parameters can be found experimentally for a given space resolution N, k, time step Δt and the number J of squaring steps in the scaling and squaring method. For example, the following sets of "stable" parameters can be used (along with the one in the above examples):
1) $N = 32, k = 5, \Delta t = 1.10^{-6} \times 2^6$, $\tau_b = \tau_f = 14400$, $\tau_d = 64$, $\lambda = 6$;

2) $N=32, k=5, \Delta t = 6.10^{-5} \times 2^6$, $\tau_b = \tau_f = 10000$, $\tau_d = 64$, $\lambda = 5$.
Note that once the stable parameters are found (say for an example where the solution is known in advance) they could be used for stable computation of unknown solutions as the stability properties depend on the space and time resolution rather than on the particular form of a solution.

VI Two-Dimensional Problems

The performance of Approach II is especially advantageous as compared to Approach I in multi dimensions. Thus, for two-dimensional problems the cost of computing the exponential operator $e^{dt_1 \mathcal{L}}$, $\mathcal{L} = a(x,y)\left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}\right)$ in the subspace $\mathbf{V}_n^k$ using Approach I amounts to $O(N^6 k^6)$ operations, and performing a time step requires $O(N^4 k^4)$ operations, where N^2 is the number of subdomains (boxes). When using Approach II, computing the exponential operator requires $O(N^2 k^6)$ operations, and making a time step takes $O(N^2 k^4)$ operations. Therefore, for a fixed order of multiwavelets, k, the computations in the subspace $\mathbf{V}_n^k$ have the complexity that is proportional to the number of subdomains N^2.

A two-dimensional heat equation reads

$$u_t(x,y) = a(x,y)\Delta u(x,y) + f(x,y), \qquad \text{in}\Omega = [0,1] \times [0,1] \tag{6.1}$$

where $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$.

The generalization of the above algorithm to the two-dimensional case is straightforward. A computational region $\Omega = \{0 \le x \le 1, 0 \le y \le 1\}$ is divided into $N \times N$, $N = 2^n$ square boxes as in Fig. 12.

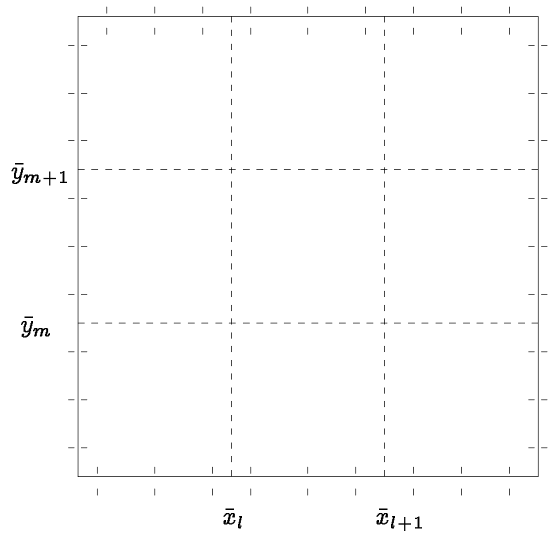

Figure 12: Subdomains and the Legendre nodes

The projection of functions onto the subspace $\mathbf{V}_n^k$ reads

$$\begin{aligned}
f(x,y) &= \sum_{l,m=0}^{2^n-1} \sum_{i,j=0}^{k-1} s^n_{il,jm} \phi^n_{il}(x) \phi^n_{jm}(y), && (6.2)\\
\Delta f(x_1,x_2) &= \sum_{l,m=0}^{2^n-1} \sum_{i,j=0}^{k-1} \zeta^n_{il,jm} \phi^n_{il}(x) \phi^n_{jm}(y), && (6.3)\\
a(x,y) &= \sum_{l,m=0}^{2^n-1} \sum_{i,j=0}^{k-1} \alpha^n_{il,jm} \phi^n_{il}(x) \phi^n_{jm}(y), && (6.4)\\
a(x,y)\Delta f(x,y) &= \sum_{l,m=0}^{2^n-1} \sum_{i,j=0}^{k-1} \tilde{\zeta}^n_{il,jm} \phi^n_{jl}(x) \phi^n_{j'l'}(y) && (6.5)
\end{aligned}$$

A transition matrix $[\sigma^n_{lm}]_{ij,i'j'}$ is defined as follows

$$\tilde{\zeta}^n_{il,jm} = \sum_{i',j'=0}^{k-1} [\sigma^n_{lm}]_{ij,i'j'} \, s_{i'l,j'm} \qquad (6.6)$$

It has a block-diagonal structure with the number of blocks equal to the number of subdomains, 2^{2n}, and the size of each block is $k^2 \times k^2$. Thus, the transition matrix transfers a block of k^4 elements of $s^n_{il,jm}$ to a block of k^4 elements of $\tilde{\zeta}^n_{il,jm}$ for each subdomain (l,m), without interaction between the neighboring subdomains.

We project both sides of (6.5) onto the basis functions $\phi^n_{il}(x_1)\phi^n_{jm}(x_2)$ and use

(6.3), (6.4) to obtain

$$\tilde{\zeta}^n_{il,jm} \;=\; 2^n \sum_{i',j'=0}^{k-1} \sum_{i'',j''=0}^{k-1} t_{ii'i''} t_{jj'j''} \zeta^n_{i'l,j'm} \alpha^n_{i''l,j''m} \tag{6.7}$$

where $t_{ii'i''}$ is the triple matrix define in (3.48).

Next, we introduce a block-diagonal transition matrix for the second derivative operator,

$$\zeta^n_{il,jm} = \sum_{i',j'=0}^{k-1} [\sigma^n_{lm}]_{ij,i'j'}\, s^n_{i'l,j'm}. \tag{6.8}$$

We also introduce a matrix $[g^n_{lm}]$ as follows,

$$[g^n_{lm}]_{ij,i'j'} = \sum_{i'',j''=0}^{k-1} t_{ii'i''} t_{jj'j''} \alpha^n_{i''l,j''m}. \tag{6.9}$$

Using (6.8) and (6.9) we rewrite (6.7) as follows

$$\begin{aligned} \tilde{\zeta}^n_{il,jm} \;&=\; 2^n \sum_{i',j'=0}^{k-1} [g^n_{lm}]_{ij,i'j'}\, \zeta^n_{i'l,j'm} \\ &=\; \sum_{i',j'=0}^{k-1} [\sigma^n_{lm}]_{ij,i'j'}\, \zeta^n_{i'l,j'm} \end{aligned}$$

where

$$[\sigma^n_{lm}]_{ij,i'j'} = 2^n \sum_{i'',j''=0}^{k-1} [g^n_{lm}]_{ij,i''j''}\, [\sigma^n_{lm}]_{i''j'',i'j'} \tag{6.10}$$

is the sought transition matrix.

The penalty term on the interfaces in two dimensions has the following form

$$\begin{aligned} f_I(x,y,t) \;&=\; \sum_{l,m=0}^{2^n-1} \sum_{i=1}^{k} \left(A_i^{(lm)} e^{-\lambda(x-\bar{x}_l)} + B_i^{(lm)} e^{-\lambda(\bar{x}_{l+1}-x)} \right) \delta(y - y_i^{(m)}) \\ &+\; \sum_{l,m=0}^{2^n-1} \sum_{i=1}^{k} \left(C_i^{(lm)} e^{-\lambda(y-\bar{y}_m)} + D_i^{(lm)} e^{-\lambda(\bar{y}_{m+1}-y)} \right) \delta(x - x_i^{(l)}), \end{aligned} \tag{6.11}$$

where $\bar{x}_l, \bar{y}_m$ are the coordinates of the interfaces, and $x_i^{(l)}$, $y_i^{(m)}$ are the local Gauss-Legendre nodes (3.25). The amplitudes $A_i^{(lm)}$ and $B_i^{(lm)}$ are proportional to the jumps in the function and its first derivative across the interfaces $x = \bar{x}_l$, as in (5.6), whereas the amplitudes $C_i^{(lm)}$ and $D_i^{(lm)}$ are proportional to the jumps across the interfaces $y = \bar{y}_m$. The structure of the forcing in two dimensions is shown in Fig. 13.

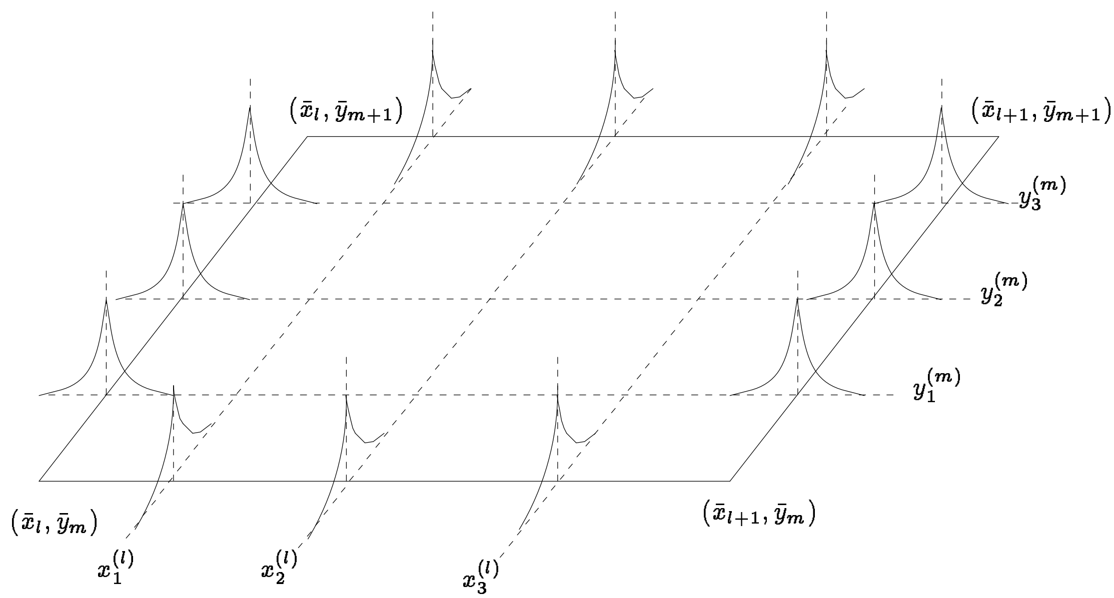

Figure 13: The structure of the forcing in the subdomain $[\bar{x}_l, \bar{x}_{l+1}] \times [\bar{y}_m, \bar{y}_{m+1}]$.

VI.1 Numerical Algorithm

Here we summarize the main techniques incorporated in the present algorithm.

1. **Time discretization method** is based on an explicit ELP scheme with the stability restriction on the time step $\Delta t \sim O(h^{-1})$, where h is the spatial mesh size.

2. **A generalized scaling and squaring method** is used to evaluate the operator valued quadrature coefficients of the ELP scheme.

3. **Discretization method in space** makes use of the expansion of functions into the multiwavelet basis and representation of operators in this basis. This involves:

   – decomposition of a computational domain into N subdomains (in each spatial direction);

   – approximation of local pieces of functions by the Legendre polynomials up to the order $k-1$ (the scaling functions);

   – computing the Legendre coefficients (which constitute the representation in the $\mathbf{V}_n^k$ subspace) using the Gauss-Legendre quadrature formula;

   – the discrete fast wavelet transform using the two-scale equation for computing the multiwavelet coefficients (not implemented yet);

   – computation of the transition matrices for the operators with variable coefficients which involves multiplication of large size matrices.

4. **The boundary conditions** (Dirichlet, Neumann or mixed) are imposed using a penalty procedure in the form of an additional non-homogeneous (forcing) term in the evolution equation.

5. **The penalty procedure on the interfaces** (interior boundaries) is implemented to establish the interaction between neighboring subdomains. This allows for drastic simplification of differential operators in the multiwavelet basis. For example, on the finest scale the (global) exponential functions of operators are represented by block-diagonal matrices rather than by full matrices.

VI.2 2D Example

We illustrate the algorithm in two dimensions by the following example.

Example 4. Linear heat equation with the forcing (6.1).

The forcing term $F(x,y)$ corresponds to the exact solution

$$u^{ref}(x,y) = \cos \pi t \left[\cos \pi(x-0.12)\ \cos \pi(y-1.3)\right]. \tag{6.12}$$

The complete forcing including the boundary and the interface penalty terms has the form

$$f(x,y) = f_B(x,y) + f_I(x,y) + F(x,y). \tag{6.13}$$

In Fig.14 the maximum absolute error is plotted for $\Delta t = 10^{-7}\times 2^7$, $k=5, N=32$, $\lambda = 6$ (the exponential factor in (6.11), and several sets of values τ_b, τ_f and τ_d (the amplitudes of the penalty terms f_B, f_I)): $\tau_b = \tau_f = 10^4, \tau_d = 225$ (line 1), $\tau_b = \tau_f = 8100, \tau_d = 225$ (line 2) and $\tau_b = \tau_f = 6400, \tau_d = 196$ (line 3). These plots demonstrate the stable and accurate long-time computation (they are obtained by 30,000 time iterations).

The following sets of parameters can be also used for stable numerical integration: (they give the maximal error about 10^{-4} for the present example):
$N = 8, k = 4, \Delta t = 1.10^{-5}\times 2^5, \tau_b = \tau_f = 1600, \tau_d = 25, \lambda = 14$ or
$N = 16, k = 4, \Delta t = 1.10^{-5}\times 2^4, \tau_b = \tau_f = 1600, \tau_d = 25, \lambda = 7$.

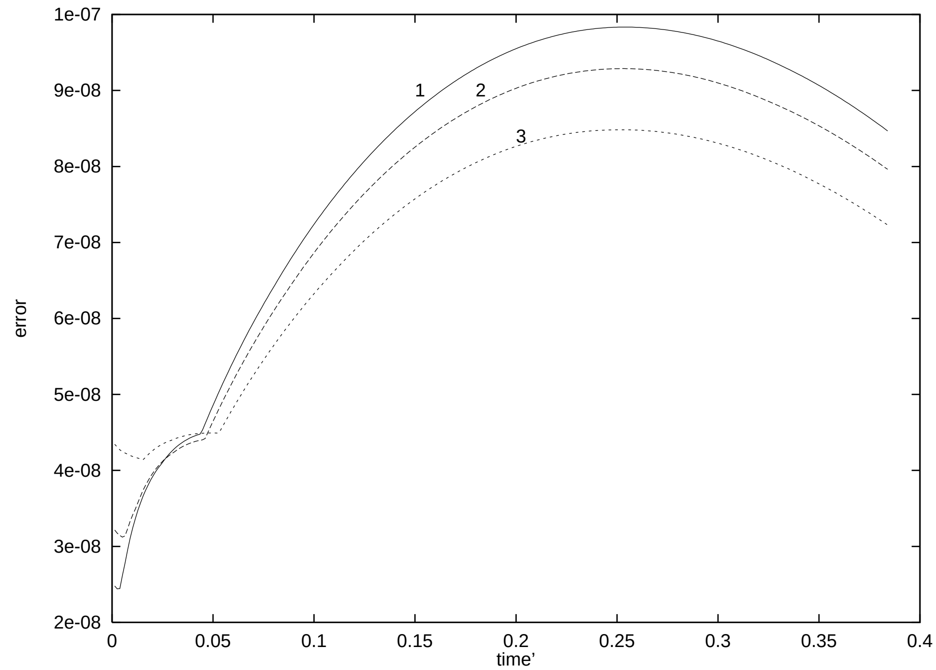

Figure 14: The maximal pointwise error for the example (6.12). The numerical parameters are specified in the text.

VII Conclusions and Further Developments

We present an efficient numerical algorithm for the solution of nonlinear time dependent problems with variable coefficients in one and two dimensions. The time discretization method employs new numerical schemes with improved stability properties. The space discretization is makes use of the expansion into multiwavelets. The boundary conditions and the continuity on the interfaces (interior boundaries between the neighboring subintervals) are imposed via a penalty procedure. For a fixed order of multiwavelets (order of polynomials or number of local grid points in each subdomain) the operation count per time step is proportional to the number of intervals.

The present paper concentrates on the implementation on the multiwavelet method on finest scale. When implemented on the finest scale, this algorithm is equivalent to the multidomain Legendre method (the scaling functions of the multiwavelet basis are the scaled and shifted Legendre polynomials). Representations of operators on other (coarser) scales can be computed via the representation on the finest scale using fast wavelet transforms [5]. That will be the subject of a future work.

The present algorithm can be extended to treat more complicated domains, for example channels with periodically excited boundaries or rectangles with two (opposite) curvilinear boundaries. By a transformation of coordinates one can obtain a problem in plane (rectangular) geometry with variable coefficient operators, see for example [13].

Such a problem can be solved using the present method.

Although the algorithm is described in one and two dimensions, its extension to three dimensional problems is straightforward.

The present algorithm based on the multiwavelet technique fits naturally the domain decomposition ideology, therefore it can be easily parallelized.

References

[1] B. Alpert. A Class of Bases in l^2 for the Sparse Representation of Integral Operators, *SIAM J. Math. Anal*, **24** (1993), p. 246–262.

[2] B. Alpert, G. Beylkin, R. R. Coifman, and V. Rokhlin. Wavelets for the fast solution of second-kind integral equations. *SIAM Journal of Scientific and Statistical Computing*, **14**, No. 1, (1993) p. 159–174. Technical report, Department of Computer Science, Yale University, New Haven, CT, 1990.

[3] B. Alpert, G. Beylkin, L. Vozovoi. Representation of operators in the multiwavelet basis, in preperation.

[4] T.M. Apostol. Calculus, Vol. 2, Chap. 7, John Wiley, 1969.

[5] G. Beylkin. On the representation of operators in bases of compactly supported wavelets. *SIAM J. Numer. Anal.*, **29**, No. 6, (1992), p. 1716–1740.

[6] G. Beylkin, R. R. Coifman, and V. Rokhlin. Fast wavelet transforms and numerical algorithms I. *Comm. Pure and Appl. Math.*, **44**, (1991), p. 141–183. Yale University Technical Report YALEU/DCS/RR-696, August (1989).

[7] G. Beylkin, J.M. Keiser. On the Adaptive Numerical Solution of Nonlinear Partial Differential Equations in Wavelet Bases. *J. Comput. Phys.*, **132**, (1997), p. 233-259.

[8] G. Beylkin, J.M. Keiser, L. Vozovoi. A new class of time discretization schemes for the solution of nonlinear PDEs. PAM report 347, March 1998, also *J. Comput. Phys.*, to appear.

[9] I. Daubechies. Orthonormal bases of compactly supported wavelets. *Comm. Pure and Appl. Math.*, **41**, (1988), p. 909-996.

[10] U. Frish, Zhen Su She, O. Thual. Viscoelastic behaviour of cellular solutions to the kuramoto-sivashinsky model, *J. Fluid Mech.*, **168**, (1986) p. 221-240.

[11] D. Funaro, D. Gottlieb. Convergence results for pseudospectral approximations of hyperbolic systems by a penalty type boundary treatment, *Math. Comput.*, Vol. 57, No. 196, (1991), p. 585-596.

[12] J.S. Hesthaven, D. Gottlieb. A stable penalty method for the compressible Navier-Stokes equations. I. Open boundary conditions, *SIAM J. Scient. Comp.*, (1996), p.579-612.

[13] L. Vozovoi, M. Israeli, A. Averbuch. Multidomain local Fourier method for PDEs in complex geometries, *J. Comput. Applied Math.*, **66**, (1996), p. 543-555.