

DOMAIN DECOMPOSITION METHODS
FOR ELLIPTIC PDEs:
THE LINEAR ALGEBRA VIEWPOINT

Irene Moulitsas

Advisor : Apostolos Hadjidimos

University of Crete
Department of Mathematics
Heraklion, Greece

July 2, 1997

Contents

1	Introduction	5
2	Formulation of the Domain Decomposition Method for the Continuous Problem	7
2.1	The First Scheme	7
2.2	The Second Scheme	8
3	Formulation of the Domain Decomposition Method for the Discrete Problem	10
3.1	Derivation of the Equations of the Linear System	10
3.1.1	First Approach	10
3.1.2	Second Approach	13
3.2	Formulation of the Iterative Scheme in Matrix Form	15
3.2.1	Formulation of the First Approach	15
3.2.2	Formulation of the Second Approach	17
3.2.3	Iterative Solution of the Linear System	18
4	Convergence Analysis	19
4.1	The Iteration Matrix T of the Procedure	20
4.2	Determination of the Eigenvalues of T	21
4.3	Determination of the Elements of the Matrix T	23
4.4	Determination of a and b : First Approach	25
4.4.1	Determination of $m'_{p-2,p-1}$ $m'_{p-1,p-1}$ $m'_{p,p}$ and $m'_{p+1,p}$	25
4.4.2	Determination of $m'_{3p-2,3p-2}$ $m'_{3p-1,3p-1}$	27
4.4.3	Determination of $m'_{3p-2,p-1}$, $m'_{3p-2,p}$, $m'_{3p-1,p-1}$ and $m'_{3p-1,p}$	28
4.4.4	Elements of the Matrix T : First Approach	29
4.4.5	Optimum Values of a and b	29
4.5	Determination of a and b : Second Approach	31
4.5.1	Determination of $m'_{p-2,p-1}$ $m'_{p-1,p-1}$ $m'_{p,p}$ and $m'_{p+1,p}$	31
4.5.2	Determination of $m'_{3p-2,3p-2}$ and $m'_{3p-1,3p-1}$	31
4.5.3	Determination of $m'_{3p-2,p-1}$, $m'_{3p-2,p}$, $m'_{3p-1,p-1}$ and $m'_{3p-1,p}$	33
4.5.4	Elements of the Matrix T : Second Approach	33
4.5.5	Optimum Values of a and b	34

5	The Poisson Equation	36
5.1	Determination of a and b : First Approach	36
5.1.1	Determination of the Elements of the Matrix M'	36
5.1.2	Elements of the Matrix T : First Approach	38
5.1.3	Optimum Values of a and b	38
5.2	Determination of a and b : Second Approach	39
5.2.1	Determination of the Elements of the Matrix M'	39
5.2.2	Elements of the Matrix T : Second Approach	39
5.2.3	Optimum Values of a and b	40
6	Numerical Experiments	41
6.1	Results from the First Approach–Helmholtz	42
6.2	Results from the Second Approach–Helmholtz	43
6.3	Results from the First Approach–Poisson	44
6.4	Results from the Second Approach–Poisson	45
7	Conclusions	49
A	Source Code developed in Fortran 77	50
A.1	First Approach	50
A.2	Second Approach	62

List of Tables

1	Example 1, Helmholtz, First Approach	42
2	Example 2, Helmholtz, First Approach	42
3	Example 3, Helmholtz, First Approach	42
4	Example 1, Helmholtz, Second Approach	43
5	Example 2, Helmholtz, Second Approach	43
6	Example 3, Helmholtz, Second Approach	43
7	Example 1, Poisson, First Approach	44
8	Example 2, Poisson, First Approach	44
9	Example 3, Poisson, First Approach	44
10	Example 1, Poisson, Second Approach	45
11	Example 2, Poisson, Second Approach	45
12	Example 3, Poisson, Second Approach	45

List of Figures

1	Discretization of Domain Ω	10
2	Example 1, Helmholtz equation	46
3	Example 2, Helmholtz equation	47
4	Example 3, Helmholtz equation	48

1 Introduction

Elliptic partial differential equations (PDEs) are important tools for mathematical modeling in a wide variety of fields. Many important advances in Science and Engineering depend on the ability to solve elliptic PDEs quickly and accurately. As a result, in the past few decades much research activity has been directed at proposing, analyzing, studying and improving numerical methods for this class of problems. Lately, domain decomposition methods have been proven to be very effective and powerful tools for the solution of elliptic PDEs. It has been observed that they can offer increased computational efficiency especially in modern multiprocessor computer environments. On the other hand they might be helpful in treating accurately some inherent difficulties arising in Physics problems, e.g., singularities, boundary layers, etc.

In the domain decomposition methods the domain under consideration is decomposed into a number of smaller subdomains and the solution in the entire domain is computed via sequences of solutions computed in the subdomains. There are two main approaches characterized by the way the subdomains are defined. Namely, the overlapping (Schwarz) approach and the non-overlapping (Schur complement) one. The convergence of the underlying iterative process is critical to the success of the specific method proposed and has therefore attracted a great deal of attention by many researchers in the area.

Numerous articles that propose, study, compare and review various overlapping methods (see, e.g., [6], [15], [16]), non-overlapping ones (see, e.g., [11]) as well as survey preconditioners for domain decomposition (see, e.g., [4]) have appeared in the literature. The comparison of the main characteristics of these two classes of methods and the existence of possible equivalence between them have received a great deal of study (see, e.g., [2], [3]). Both approaches have already been used in a real life environment to effectively model large scale, industrial, ill-conditioned problems (see, e.g., [9], [13], [14]). Nevertheless it is believed that both theoretical and experimental analysis is required before such methods become practical and useful tools for non-experts.

In this study we are interested in non-overlapping domain decomposition methods which are formulated as iterative interface smoothing procedures. From this interface relaxation viewpoint non-overlapping domain decomposition methods consist of partitioning the domain in a set of non-overlapping subdomains and appropriate boundary conditions on the interface lines defined by the partitioning are imposed. Then, using arbitrary initial guesses on the interfaces, the set of the resulting PDE problems is solved. In each iteration subproblems, with Dirichlet and Neumann boundary conditions on the interfaces of each subdomain, are solved alternatively. Since, in general, the obtained solutions do not satisfy the interface boundary conditions an interface relaxation is applied to obtain new interface boundary values, which, hopefully, will satisfy them better, and the PDE subproblems are solved again using these new values. If the relaxation parameter(s) is(are) chosen appropriately the process just described, when applied iteratively, produces sequences of solutions in each subdomain who eventually converge to the solution of the original PDE problem in the entire domain.

For our study we select two such methods whose formulation and theoretical analysis at differential equation level can be found in [10] and [12]. Although their convergence has been theoretically analyzed to some extent in the aforementioned references, important questions (like the selection of the relaxation parameter values involved in both methods) are left unanswered. Our main objective here is an attempt to analyze and formulate these two domain decomposition schemes at linear algebra level hoping that we will elucidate the role of the relaxation parameters in the iteration process.

The rest of this thesis is organized as follows. In Chapter 2 we present the formulation of the two domain decomposition methods at differential equation level. In Chapter 3 we present in detail the formulation of the two domain decomposition methods at discrete (linear algebra) level for a Helmholtz Equation in the one-dimensional case. The convergence analysis carried out at linear algebra level for both methods is presented in Chapter 4. In Chapter 5 we present the formulation of the two domain decomposition methods and the convergence analysis for the corresponding Poisson Equation. Chapter 6 contains a description of the implementation of our schemes using software that we have developed in Fortran 77. It also contains sets of experimental data that support our theoretical results. Finally, Chapter 7 contains a summary of our results as well as some concluding remarks.

2 Formulation of the Domain Decomposition Method for the Continuous Problem

We consider the PDE problem

$$Lu = f \text{ in } \Omega, \quad u = g \text{ on } \partial\Omega \quad (1)$$

where L is a second order elliptic differential operator, f, g are such that $f \in L^2(\Omega)$ while $g \in H^{\frac{1}{2}}(\partial\Omega)$, where $\Omega \subset R^d$, $d = 1, 2, \dots$, is an open convex domain with piecewise smooth boundary $\partial\Omega$. Assume that Ω is split into k open subdomains Ω_i , $i = 1, \dots, k$, such that $\bar{\Omega} = \cup_{i=1}^k \bar{\Omega}_i$, $\Omega_i \cap \Omega_j = \emptyset$ and $\partial\Omega \cap \partial\Omega_i \neq \emptyset$, $i, j = 1, \dots, k$. For reasons related either to the characteristics of this problem or to the computing resources available one would like to replace (1) with a system of similar problems defined on the subdomains Ω_i . There are many ways to realize the coupling of the problems in this system (see [11]). From them we select one, consider two different approaches and subsequently we present the corresponding iterative schemes. They both use Dirichlet boundary conditions on the interfaces. The first approach does not use the fact that the operator L acts on the interface while the second one does.

2.1 The First Scheme

For simplicity we choose $k = 2$ and denote the interface of the two subdomains by $\Gamma = \partial\Omega_1 \cap \partial\Omega_2$. Under certain conditions the original problem is equivalent to the following split problem

$$\left. \begin{array}{l} Lu_1 = f \quad \text{in} \quad \Omega_1 \\ u_1 = g \quad \text{on} \quad \partial\Omega \cap \partial\Omega_1 \\ u_1 = u_2 \quad \text{on} \quad \Gamma \\ \frac{\partial u_1}{\partial \nu^1} + \frac{\partial u_2}{\partial \nu^2} = 0 \quad \text{on} \quad \Gamma \end{array} \right| \begin{array}{l} Lu_2 = f \quad \text{in} \quad \Omega_2 \\ u_2 = g \quad \text{on} \quad \partial\Omega \cap \partial\Omega_2 \\ u_2 = u_1 \quad \text{on} \quad \Gamma \\ \frac{\partial u_2}{\partial \nu^2} + \frac{\partial u_1}{\partial \nu^1} = 0 \quad \text{on} \quad \Gamma \end{array}$$

where for $n = 1, 2$, $u_n = u|_{\Omega_n}$ and where ν^n is the outward unit normal vector to $\partial\Omega_n$.

This allows us to define the following domain decomposition method. We arbitrarily choose $u_n^{(0)} \in H^1(\Omega_n)$ with $u_n^{(0)}|_{\partial\Omega \cap \partial\Omega_n} = g$ and, for $i = 0, 1, 2, \dots$, construct the sequence $u_n^{(i+1)} \in H^1(\Omega_n)$ with $u_n^{(i+1)}|_{\partial\Omega \cap \partial\Omega_n} = g$, $n = 1, 2$, satisfying

$$\begin{aligned}
 Lu_1^{(2i+1)} &= f \text{ in } \Omega_1, & u_1^{(2i+1)} &= bu_1^{(2i)} + (1-b)u_2^{(2i)} \text{ on } \Gamma \\
 Lu_2^{(2i+1)} &= f \text{ in } \Omega_2, & u_2^{(2i+1)} &= (1-b)u_1^{(2i)} + bu_2^{(2i)} \text{ on } \Gamma
 \end{aligned} \tag{2}$$

$$\begin{aligned}
 Lu_1^{(2i+2)} &= f \text{ in } \Omega_1, & \frac{\partial u_1^{(2i+2)}}{\partial \nu^1} &= a \frac{\partial u_1^{(2i+1)}}{\partial \nu^1} + (1-a) \frac{\partial u_2^{(2i+1)}}{\partial \nu^1} \text{ on } \Gamma \\
 Lu_2^{(2i+2)} &= f \text{ in } \Omega_2, & \frac{\partial u_2^{(2i+2)}}{\partial \nu^2} &= (1-a) \frac{\partial u_1^{(2i+1)}}{\partial \nu^2} + a \frac{\partial u_2^{(2i+1)}}{\partial \nu^2} \text{ on } \Gamma
 \end{aligned}$$

where a, b are relaxation parameters to be determined later.

2.2 The Second Scheme

Again we choose $k = 2$ and denote the interface of the two subdomains by $\Gamma = \partial\Omega_1 \cap \partial\Omega_2$. This time the split problem is

$$\begin{array}{l|l}
 \begin{array}{l}
 Lu_1 = f \quad \text{in } \Omega_1 \cup \Gamma \\
 u_1 = g \quad \text{on } \partial\Omega \cap \partial\Omega_1 \\
 u_1 = u_2 \quad \text{on } \Gamma \\
 \frac{\partial u_1}{\partial \nu^1} + \frac{\partial u_2}{\partial \nu^2} = 0 \quad \text{on } \Gamma
 \end{array} &
 \begin{array}{l}
 Lu_2 = f \quad \text{in } \Omega_2 \cup \Gamma \\
 u_2 = g \quad \text{on } \partial\Omega \cap \partial\Omega_2 \\
 u_2 = u_1 \quad \text{on } \Gamma \\
 \frac{\partial u_2}{\partial \nu^2} + \frac{\partial u_1}{\partial \nu^1} = 0 \quad \text{on } \Gamma
 \end{array}
 \end{array}$$

where for $n = 1, 2$, $u_n = u|_{\Omega_n}$ and where ν^n is the outward unit normal vector to $\partial\Omega_n$.

As we have already mentioned, this second approach differs from the first one because operator L acts not only on domains Ω_1 and Ω_2 but on the interface Γ as well.

In the corresponding domain decomposition method we arbitrarily choose $u_n^{(0)} \in H^1(\Omega_n)$ with $u_n^{(0)}|_{\partial\Omega \cap \partial\Omega_n} = g$ and, for $i = 0, 1, 2, \dots$, construct the sequence $u_n^{(i+1)} \in H^1(\Omega_n)$ with $u_n^{(i+1)}|_{\partial\Omega \cap \partial\Omega_n} = g$, $n = 1, 2$, satisfying

$$\begin{aligned}
 Lu_1^{(2i+1)} &= f \text{ in } \Omega_1 \cup \Gamma, & u_1^{(2i+1)} &= bu_1^{(2i)} + (1-b)u_2^{(2i)} \text{ on } \Gamma \\
 Lu_2^{(2i+1)} &= f \text{ in } \Omega_2 \cup \Gamma, & u_2^{(2i+1)} &= (1-b)u_1^{(2i)} + bu_2^{(2i)} \text{ on } \Gamma
 \end{aligned} \tag{3}$$

$$\begin{aligned}
 Lu_1^{(2i+2)} &= f \text{ in } \Omega_1 \cup \Gamma, & \frac{\partial u_1^{(2i+2)}}{\partial \nu^1} &= a \frac{\partial u_1^{(2i+1)}}{\partial \nu^1} + (1-a) \frac{\partial u_2^{(2i+1)}}{\partial \nu^1} \text{ on } \Gamma \\
 Lu_2^{(2i+2)} &= f \text{ in } \Omega_2 \cup \Gamma, & \frac{\partial u_2^{(2i+2)}}{\partial \nu^2} &= (1-a) \frac{\partial u_1^{(2i+1)}}{\partial \nu^2} + a \frac{\partial u_2^{(2i+1)}}{\partial \nu^2} \text{ on } \Gamma
 \end{aligned}$$

where a, b are relaxation parameters to be determined.

Most researchers work at the differential equation level. This means that they try to determine values for the parameters a, b so that the scheme proposed produces a sequence of solutions that converges to the solution of the original PDE problem; in other words, to make the errors vanish asymptotically along the interface and also to determine those (optimal) values of a, b that make the scheme converge asymptotically as fast as possible. Then, they discretize the (continuous) PDE problem(s) and the linear system(s) produced is(are) solved by using as values for the parameters a, b the ones determined from the continuous problem(s). For convergence results at the PDE level see e.g., [11].

On the other hand, very few research works at the linear algebra level can be found in the literature (see e.g. [7], [8], [15], [16], etc.). This is because of the inherent difficulties the solution to the corresponding linear problem presents. In this case, first the entire domain is discretized, next the subdomains are defined and then finite difference, finite element or even collocation methods are used to approximate the PDE problem and subproblems. Following exactly the same idea as the one in the continuous case, the solution to the linear system is found by means of an analogous iterative scheme. So, the problem this time is to determine the values of the parameters a, b in such a way as to make the iterative scheme that solves the linear system at hand converge and if possible converge in an optimal sense. In general, the determination of the parameters a, b this time constitutes a much more difficult problem.

3 Formulation of the Domain Decomposition Method for the Discrete Problem

To illustrate the difficulties the method presents in this case we restrict ourselves to studying the solution of the one-dimensional Helmholtz equation where for simplicity we will consider the case of two subdomains. More specifically, we consider that the model problem is given by

$$\begin{aligned} -u'' + cu &= f & \text{in } \Omega \equiv (0, 1) \\ u(0) &= \alpha & \text{and } u(1) = \beta \end{aligned} \tag{4}$$

where c is a positive constant and α, β are the given boundary values.

We then split the domain Ω into the two subdomains $\Omega_1 \equiv (0, \frac{1}{2})$ and $\Omega_2 \equiv (\frac{1}{2}, 1)$ so that the interface Γ is at $x = \frac{1}{2}$. A uniform grid of mesh size h is imposed on $\bar{\Omega}$, where $h = \frac{1}{n+1}$, $n \geq 5$ odd, and a three-point finite difference discretization formula is used to approximate the values of the unknown function at the grid points. Let $p = \frac{n+1}{2}$ be the index of the point x_i that corresponds to the interface. For reasons that will become clear in the sequel, in case we are working in subdomain Ω_1 the index p will be used as p while if we are working in subdomain Ω_2 p' will be used instead.

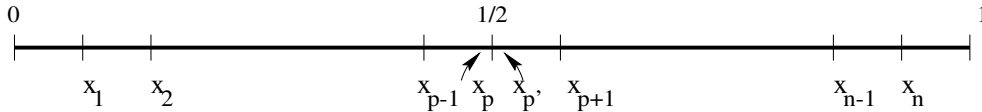


Figure 1: Discretization of Domain Ω

3.1 Derivation of the Equations of the Linear System

3.1.1 First Approach

Using Taylor series expansion about any interior grid point x_i we can obtain that

$$u''(x_i) = \frac{u(x_{i-1}) - 2u(x_i) + u(x_{i+1}))}{h^2} + O(h^2).$$

Let u_i denote the approximation of $u(x_i)$ resulting from the numerical scheme. Then considering the given differential equation (4) at all grid points, excluding the one on the interface, and following closely the iterative scheme of the domain decomposition method described previously in (2), after the first sweep of the iterative method on the subdomain Ω_1 the following equations will be yielded

$$\begin{aligned}
 (2 + ch^2)u_1^{(1)} - u_2^{(1)} &= h^2 f(x_1) + \alpha \\
 -u_1^{(1)} + (2 + ch^2)u_2^{(1)} - u_3^{(1)} &= h^2 f(x_2) \\
 &\vdots \\
 -u_{p-3}^{(1)} + (2 + ch^2)u_{p-2}^{(1)} - u_{p-1}^{(1)} &= h^2 f(x_{p-2}) \\
 -u_{p-2}^{(1)} + (2 + ch^2)u_{p-1}^{(1)} - u_p^{(1)} &= h^2 f(x_{p-1}) \\
 u_p^{(1)} &= bu_p^{(0)} + (1 - b)u_{p'}^{(0)}
 \end{aligned} \tag{5}$$

Obviously, the last equation satisfies the domain decomposition method requirement on the interface. For the subdomain Ω_2 , the corresponding equations will be

$$\begin{aligned}
 u_{p'}^{(1)} &= (1 - b)u_p^{(0)} + bu_{p'}^{(0)} \\
 -u_{p'}^{(1)} + (2 + ch^2)u_{p+1}^{(1)} - u_{p+2}^{(1)} &= h^2 f(x_{p+1}) \\
 -u_{p+1}^{(1)} + (2 + ch^2)u_{p+2}^{(1)} - u_{p+3}^{(1)} &= h^2 f(x_{p+2}) \\
 &\vdots \\
 -u_{n-2}^{(1)} + (2 + ch^2)u_{n-1}^{(1)} - u_n^{(1)} &= h^2 f(x_{n-1}) \\
 -u_{n-1}^{(1)} + (2 + ch^2)u_n^{(1)} &= h^2 f(x_n) + \beta
 \end{aligned} \tag{6}$$

If we now proceed to the next iteration, the domain decomposition method requirement on the interface for Ω_1 will be $u_p^{(2)} = au_p^{(1)} + (1 - a)u_{p'}^{(1)}$ while for Ω_2 will be $u_{p'}^{(2)} = (1 - a)u_p^{(1)} + au_{p'}^{(1)}$. To find a way to determine the derivative at the points x_p and $x_{p'}$, using in each case points only from Ω_1

and Ω_2 , respectively, we work as follows ([5]). For u'_p we write down

$$\begin{aligned} u(x) &= u(x) \\ u(x-h) &= u(x) - hu'(x) + \frac{h^2}{2}u''(x) + O(h^3) \\ u(x-2h) &= u(x) - 2hu'(x) + 2h^2u''(x) + O(h^3) \end{aligned}$$

from which one can very easily get that

$$-3u(x) + 4u(x-h) - u(x-2h) = -2hu'(x) + O(h^3)$$

or, equivalently,

$$u'(x) = \frac{3u(x) - 4u(x-h) + u(x-2h)}{2h} + O(h^2)$$

Similarly, for Ω_2 we will have

$$u'(x) = \frac{-3u(x) + 4u(x+h) - u(x+2h)}{2h} + O(h^2)$$

To satisfy the aforementioned requirements on the interface, we will have

$$\begin{aligned} u_p^{(2)} &= au_p^{(1)} + (1-a)u_{p'}^{(1)} \iff \\ 3u_p^{(2)} - 4u_{p-1}^{(2)} + u_{p-2}^{(2)} &= \\ a(3u_p^{(1)} - 4u_{p-1}^{(1)} + u_{p-2}^{(1)}) + (1-a)(-3u_{p'}^{(1)} + 4u_{p+1}^{(1)} - u_{p+2}^{(1)}) &\iff \\ 3u_p^{(2)} - 4u_{p-1}^{(2)} + u_{p-2}^{(2)} &= \\ a(3(bu_p^{(0)} + (1-b)u_{p'}^{(0)}) - 4u_{p-1}^{(1)} + u_{p-2}^{(1)}) + & \\ (1-a)(-3((1-b)u_p^{(0)} + bu_{p'}^{(0)}) + 4u_{p+1}^{(1)} - u_{p+2}^{(1)}) &\iff \\ -au_{p-2}^{(1)} + 4au_{p-1}^{(1)} - 4(1-a)u_{p+1}^{(1)} + (1-a)u_{p+2}^{(1)} + u_{p-2}^{(2)} - 4u_{p-1}^{(2)} + 3u_p^{(2)} &= \\ 3(a+b-1)u_p^{(0)} + 3(a-b)u_{p'}^{(0)} & \end{aligned}$$

So the equations for Ω_1 for the second iteration will be

$$\begin{aligned}
 (2 + ch^2)u_1^{(2)} - u_2^{(2)} &= h^2 f(x_1) + \alpha \\
 -u_1^{(2)} + (2 + ch^2)u_2^{(2)} - u_3^{(2)} &= h^2 f(x_2) \\
 &\vdots \\
 -u_{p-3}^{(2)} + (2 + ch^2)u_{p-2}^{(2)} - u_{p-1}^{(2)} &= h^2 f(x_{p-2}) \\
 -u_{p-2}^{(2)} + (2 + ch^2)u_{p-1}^{(2)} - u_p^{(2)} &= h^2 f(x_{p-1}) \\
 -au_{p-2}^{(1)} + 4au_{p-1}^{(1)} - 4(1-a)u_{p+1}^{(1)} + (1-a)u_{p+2}^{(1)} + u_{p-2}^{(2)} - 4u_{p-1}^{(2)} + 3u_p^{(2)} &= \\
 3(a+b-1)u_p^{(0)} + 3(a-b)u_{p'}^{(0)} &
 \end{aligned} \tag{7}$$

Again to satisfy the requirements on Γ for Ω_2 we will have

$$\begin{aligned}
 u_{p'}^{(2)} &= (1-a)u_p^{(1)} + au_{p'}^{(1)} \iff \\
 (1-a)u_{p-2}^{(1)} - 4(1-a)u_{p-1}^{(1)} + 4au_{p+1}^{(1)} - au_{p+2}^{(1)} + 3u_{p'}^{(2)} - 4u_{p+1}^{(2)} + u_{p+2}^{(2)} &= \\
 3(a-b)u_p^{(0)} + 3(a+b-1)u_{p'}^{(0)} &
 \end{aligned}$$

So the equations for Ω_2 for the second iteration will be

$$\begin{aligned}
 (1-a)u_{p-2}^{(1)} - 4(1-a)u_{p-1}^{(1)} + 4au_{p+1}^{(1)} - au_{p+2}^{(1)} + 3u_{p'}^{(2)} - 4u_{p+1}^{(2)} + u_{p+2}^{(2)} &= \\
 3(a-b)u_p^{(0)} + 3(a+b-1)u_{p'}^{(0)} & \\
 -u_{p'}^{(2)} + (2 + ch^2)u_{p+1}^{(2)} - u_{p+2}^{(2)} &= h^2 f(x_{p+1}) \\
 -u_{p+1}^{(2)} + (2 + ch^2)u_{p+2}^{(2)} - u_{p+3}^{(2)} &= h^2 f(x_{p+2}) \\
 &\vdots \\
 -u_{n-2}^{(2)} + (2 + ch^2)u_{n-1}^{(2)} - u_n^{(2)} &= h^2 f(x_{n-1}) \\
 -u_{n-1}^{(2)} + (2 + ch^2)u_n^{(2)} &= h^2 f(x_n) + \beta
 \end{aligned} \tag{8}$$

3.1.2 Second Approach

As we have already mentioned, the second approach differs from the first one, because we use the differential equation on the interface node as well. So in this case the last equation of the first and third block and the first equation of the second and fourth block, will not only satisfy the requirements

of our domain decomposition method, but they will also be the equations at the interface point(s) and thus will implicitly satisfy the requirements of our method.

So for the first iteration, in Ω_1 , we will have

$$-u_{p-2}^{(1)} + (2 + ch^2)u_{p-1}^{(1)} - u_p^{(1)} = h^2 f(x_{p-1})$$

We substitute $u_p^{(1)}$ with $bu_p^{(0)} + (1 - b)u_{p'}^{(0)}$ to get

$$-u_{p-2}^{(1)} + (2 + ch^2)u_{p-1}^{(1)} = h^2 f(x_{p-1}) + bu_p^{(0)} + (1 - b)u_{p'}^{(0)}$$

Similarly, in Ω_2 we will have

$$-u_{p'}^{(1)} + (2 + ch^2)u_{p+1}^{(1)} - u_{p+2}^{(1)} = h^2 f(x_{p+1})$$

We now substitute $u_{p'}^{(1)}$ with $(1 - b)u_p^{(0)} + bu_{p'}^{(0)}$.

Thus our new equation will be

$$(2 + ch^2)u_{p+1}^{(1)} - u_{p+2}^{(1)} = h^2 f(x_{p+1}) + (1 - b)u_p^{(0)} + bu_{p'}^{(0)}$$

For the second iteration things are slightly different. If we tried to apply the discretization scheme on the interface we would have

$$-u_{p-1}^{(2)} + (2 + ch^2)u_p^{(2)} - u_{p+1}^{(2)} = h^2 f(x_p)$$

But now u_{p+1} is only a fictitious point for Ω_1 . So we need to find a way to replace u_{p+1} with values at points in the left subdomain. For this we know that

$$u'(x) = \frac{u(x + h) - u(x - h)}{2h} + O(h^2).$$

So, applying this relation at the point x_p we equivalently have that

$$u(x_{p+1}) = 2hu'(x_p) + u(x_{p-1}) + O(h^2)$$

and the equation is now transformed into

$$-u_{p-1}^{(2)} + (2 + ch^2)u_p^{(2)} - 2hu_p^{\prime(2)} - u_{p-1}^{(2)} = h^2 f(x_p)$$

We now substitute $u_p^{(2)}$ with $au_p^{\prime(1)} + (1-a)u_{p'}^{(1)}$. Thus our new equation will be

$$-2hau_p^{\prime(1)} - 2h(1-a)u_{p'}^{(1)} - 2u_{p-1}^{(2)} + (2 + ch^2)u_p^{(2)} = h^2 f(x_p)$$

However, since we have already found expressions for u_p' and $u_{p'}'$, using points only from Ω_1 and Ω_2 respectively, we substitute these expressions in the equation above to obtain an equivalent one. Namely,

$$\begin{aligned} & -au_{p-2}^{(1)} + 4au_{p-1}^{(1)} - 4(1-a)u_{p+1}^{(1)} + (1-a)u_{p+2}^{(1)} - 2u_{p-1}^{(2)} + (2 + ch^2)u_p^{(2)} = \\ & h^2 f(x_p) + 3au_p^{(0)} - 3(1-a)u_{p'}^{(0)} \end{aligned}$$

Similarly, for Ω_2 , the equation that we will obtain eventually is

$$\begin{aligned} & (1-a)u_{p-2}^{(1)} - 4(1-a)u_{p-1}^{(1)} + 4au_{p+1}^{(1)} - au_{p+2}^{(1)} + (2 + ch^2)u_{p'}^{(2)} - 2u_{p+1}^{(2)} = \\ & h^2 f(x_p) - 3(1-a)u_p^{(0)} + 3au_{p'}^{(0)} \end{aligned}$$

The rest remains the same as in the previous approach.

3.2 Formulation of the Iterative Scheme in Matrix Form

In the previous section we saw how we could obtain two successive iterations for the domain decomposition method we proposed. We will now consider these two iterations in one. In this way, a new iteration will consist of a Dirichlet-type half-iteration and a Neumann-type half-iteration. For the sake of simplicity in the calculations that will follow we will make the substitution

$$2 + ch^2 = 2 \cosh \phi.$$

3.2.1 Formulation of the First Approach

The sets of equations (5), (6), (7) and (8) lead to the equivalent iterative scheme

3.2.3 Iterative Solution of the Linear System

Each one of the two iterative schemes in the two different approaches solves a linear system. This system is formed from the equations that approximate the given differential equation at all the interior grid points of the two sub-domains and also from the equations obtained from conditions imposed on the interface in both the Dirichlet and the Neumann iterations. Each linear system has the form $Au = b$ or, equivalently, $(M - N)u = b$. The matrices M and N are the two matrices shown on the left and the right hand sides, respectively, of the iterative schemes in sections (3.2.1) and (3.2.2), the vector b is the known vector of the right hand sides, while the unknown vector u is the vector that contains the approximate values of $u(x)$ at the grid points and the point on the interface. This vector is the limit of the sequence of vectors $u^{(i)}$ produced by each one of the two iterative schemes proposed provided they converge.

The dimensions of the first two tri-diagonal blocks of the matrix M are $(p-1) \times (p-1)$, while those of the last two tri-diagonal blocks are $p \times p$, where as we have already mentioned, p is the index of the point that corresponds to the interface point $p = \frac{n+1}{2}$. Thus the dimensions of the two matrices M and N in each iteration scheme are $2n \times 2n$

4 Convergence Analysis

In this section our aim is to determine the optimum values for the parameters a, b so that our iterative schemes converge as fast as possible (see, e.g., [1], [17], and [18]). The iterative schemes that we have formed in the previous section can be described by the following procedure $Mu^{(i+1)} = Nu^{(i)} + b$. In both approaches the matrices M and N involved can be written in general in the forms below

$$M =$$

$$\left(\begin{array}{c|c|c|c} M_{11} & & & \\ \hline & M_{22} & & \\ \hline M_{31} & M_{32} & M_{33} & \\ \hline & & & \\ \hline M_{41} & M_{42} & & M_{44} \end{array} \right)$$

and

$$N =$$

$$\left(\begin{array}{c|c|c|c} & & N_{13} & N_{14} \\ \hline & & & \\ \hline & & N_{23} & N_{24} \\ \hline & & N_{33} & N_{34} \\ \hline & & N_{43} & N_{44} \end{array} \right)$$

It is noted that the symbol " – " in the previous matrices indicates the **only** possible positions in the corresponding blocks where the elements may be non-zero ones.

4.1 The Iteration Matrix T of the Procedure

Let $T = M'N$, where $M' = M^{-1}$, be the iteration matrix of the procedure. Exploiting the block form as well as the sparsity of the matrix M we may avoid computing M^{-1} explicitly since this is a very demanding and unaffordable task. For this reason, we shall take advantage of the properties that M appears to have. For this we can easily realize and prove that

1. M^{-1} will not have the 'Zero Blocks' filled in with non-zero elements,
2. The blocks $M_{11}^{-1}, M_{22}^{-1}, M_{33}^{-1}$ and M_{44}^{-1} will be filled with non-zero elements and will not preserve the tridiagonal form any more.

These remarks make of course our approach much easier because we now need to compute only the blocks that appear in the figure below

$$M' = \begin{pmatrix} M'_{11} & & & \\ & M'_{22} & & \\ M'_{31} & M'_{32} & M'_{33} & \\ & M'_{42} & & M'_{44} \end{pmatrix}$$

where

$$M \times M' = I.$$

The blocks of the matrix M^{-1} , we need to compute satisfy the equations

$$\begin{aligned}
 M_{11}M'_{11} &= I \\
 M_{22}M'_{22} &= I \\
 M_{33}M'_{33} &= I \\
 M_{44}M'_{44} &= I \\
 M_{31}M'_{11} + M_{33}M'_{31} &= 0 \\
 M_{32}M'_{22} + M_{33}M'_{32} &= 0 \\
 M_{41}M'_{11} + M_{44}M'_{41} &= 0 \\
 M_{42}M'_{22} + M_{44}M'_{42} &= 0
 \end{aligned} \tag{9}$$

Because of the sparsity of N it can be readily checked that $T = M^{-1}N$ will have the following sparsity pattern

$$\left(\begin{array}{cc|cc}
 & & - & - \\
 & & T_{13} & \vdots & \vdots & T_{14} \\
 \hline
 & & - & - \\
 & & T_{23} & \vdots & \vdots & T_{24} \\
 \hline
 & & - & - \\
 & & T_{33} & \vdots & \vdots & T_{34} \\
 \hline
 & & - & - \\
 & & T_{43} & \vdots & \vdots & T_{44} \\
 & & - & -
 \end{array} \right)$$

The sparsity pattern of T indicated above makes the task of the determination of the spectrum of it much easier.

4.2 Determination of the Eigenvalues of T

Since our objective is to solve the optimization problem $\min_{a,b} \rho(T)$ we must find the eigenvalues of T . For this we form the expression below

$$T - \lambda I = \begin{pmatrix} -\lambda I & & T_{13} & T_{14} \\ & -\lambda I & T_{23} & T_{24} \\ & & T_{33} - \lambda I & T_{34} \\ & & T_{41} & T_{42} - \lambda I \end{pmatrix}$$

Our problem is now to find all $\lambda \in \mathcal{C}$ such that $\det(T - \lambda I) = 0$.

If we expand the determinant about the elements of its first column etc., we can readily see that because of the presence of zero elements in the first $2(p-1)$ columns we will obtain

$$\det(T - \lambda I) = (-\lambda)^{n-1} \det \left(\begin{array}{ccc|cc} & & - & - & \\ & & \vdots & \vdots & T_{34} \\ \hline & & - & - & \\ T_{43} & & \vdots & \vdots & T_{44} - \lambda I \\ & & - & - & \end{array} \right)$$

and so we need to compute the determinant of the last two blocks only. Using the same idea again and expanding successively about the elements of the first and the last $p-1$ columns, we will eventually obtain

$$\det(T - \lambda I) = (-\lambda)^{2n-2} \det \left(\begin{array}{c|c} -\lambda + t_{3p-2,3p-2} & t_{3p-2,3p-1} \\ \hline t_{3p-1,3p-2} & -\lambda + t_{3p-1,3p-1} \end{array} \right)$$

So, we observe that in order to determine explicitly all the eigenvalues of the iteration matrix T , **only four** of its elements are needed!

Putting

$$\det(T - \lambda I) = 0 \iff (-\lambda)^{2n-2}((-\lambda + t_{3p-2,3p-2})(-\lambda + t_{3p-1,3p-1}) - t_{3p-2,3p-1}t_{3p-1,3p-2}) = 0$$

we can see that T has the eigenvalue $\lambda = 0$ with a multiplicity $2n - 2$ and two other eigenvalues that are given as the zeros of a quadratic equation whose coefficients are functions of the two parameters a , b . From the observation just made and bearing in mind that our problem is that of the minimization of $\rho(T)$ we are just wondering if we can find values for the two parameters that make the zeros of the aforementioned quadratic be zero. For this the following two equations must be satisfied

$$t_{3p-2,3p-2} + t_{3p-1,3p-1} = 0 \quad (10)$$

$$t_{3p-2,3p-2} t_{3p-1,3p-1} - t_{3p-1,3p-2} t_{3p-2,3p-1} = 0 \quad (11)$$

4.3 Determination of the Elements of the Matrix T

At this point we should see how we can determine the elements of the iterative matrix $T = M'N$ that appear in equations (10) and (11).

For $n \leq i \leq 2n$

$$\begin{aligned} t_{i,3p-2} &= \sum_{k=1}^n m'_{i,k} n_{k,3p-2} \\ &= m'_{i,p-1} n_{p-1,3p-2} + m'_{i,p} n_{p,3p-2} + m'_{i,3p-2} n_{3p-2,3p-2} + m'_{i,3p-1} n_{3p-1,3p-2} \end{aligned} \quad (12)$$

$$\begin{aligned} t_{i,3p-1} &= \sum_{k=1}^n m'_{i,k} n_{k,3p-1} \\ &= m'_{i,p-1} n_{p-1,3p-1} + m'_{i,p} n_{p,3p-1} + m'_{i,3p-2} n_{3p-2,3p-1} + m'_{i,3p-1} n_{3p-1,3p-1} \end{aligned} \quad (13)$$

If we apply equations (12) and (13) for determining the four elements of T that we need in equations (10) and (11) we will have

$$t_{3p-2,3p-2} = m'_{3p-2,p-1}n_{p-1,3p-2} + m'_{3p-2,p}n_{p,3p-2} + m'_{3p-2,3p-2}n_{3p-2,3p-2} \quad (14)$$

$$t_{3p-2,3p-1} = m'_{3p-2,p-1}n_{p-1,3p-1} + m'_{3p-2,p}n_{p,3p-1} + m'_{3p-2,3p-2}n_{3p-2,3p-1} \quad (15)$$

$$t_{3p-1,3p-2} = m'_{3p-1,p-1}n_{p-1,3p-2} + m'_{3p-1,p}n_{p,3p-2} + m'_{3p-1,3p-1}n_{3p-1,3p-2} \quad (16)$$

$$t_{3p-1,3p-1} = m'_{3p-1,p-1}n_{p-1,3p-1} + m'_{3p-1,p}n_{p,3p-1} + m'_{3p-1,3p-1}n_{3p-1,3p-1} \quad (17)$$

In the aforementioned equations, there are **only six** elements of M' ($= M^{-1}$) involved. So, what we need is to determine these six elements only.

So far, we have taken, in some way, advantage of the form and sparsity of all the matrices and submatrices involved in order to avoid heavy and unnecessary computations. In the sequel, we will take full advantage of the properties of all the matrices, to obtain analytic expressions for the very few elements that we need for our computations in the most convenient way. In order to find them, difference equations will be used ([5]).

From equations (9) we obtain equivalently the equations we need to compute M'

$$\begin{aligned} M'_{11} &= M_{11}^{-1} \\ M'_{22} &= M_{22}^{-1} \\ M'_{33} &= M_{33}^{-1} \\ M'_{44} &= M_{44}^{-1} \\ M'_{31} &= M'_{33}M_{31}M'_{11} \\ M'_{32} &= M'_{33}M_{32}M'_{22} \\ M'_{41} &= M'_{44}M_{41}M'_{11} \\ M'_{42} &= M'_{44}M_{42}M'_{22} \end{aligned}$$

If we use the equations above we have that

$$m'_{3p-2,p-1} = -m'_{3p-2,3p-2}(m_{3p-2,p-2}m'_{p-2,p-1} + m_{3p-2,p-1}m'_{p-1,p-1}) \quad (18)$$

$$m'_{3p-2,p} = -m'_{3p-2,3p-2}(m_{3p-2,p}m'_{p,p} + m_{3p-2,p+1}m'_{p+1,p}) \quad (19)$$

$$m'_{3p-1,p-1} = -m'_{3p-1,3p-1}(m_{3p-1,p-2}m'_{p-2,p-1} + m_{3p-1,p-1}m'_{p-1,p-1}) \quad (20)$$

$$m'_{3p-1,p} = -m'_{3p-1,3p-1}(m_{3p-1,p}m'_{p,p} + m_{3p-1,p+1}m'_{p+1,p}) \quad (21)$$

4.4 Determination of a and b : First Approach

We substitute the elements of the matrix N in equations (14), (15), (16) and (17). So we equivalently have

$$t_{3p-2,3p-2} = bm'_{3p-2,p-1} + (1-b)m'_{3p-2,p} + 3(a+b-1)m'_{3p-2,3p-2} \quad (22)$$

$$t_{3p-2,3p-1} = (1-b)m'_{3p-2,p-1} + bm'_{3p-2,p} - 3(b-a)m'_{3p-2,3p-2} \quad (23)$$

$$t_{3p-1,3p-2} = bm'_{3p-1,p-1} + (1-b)m'_{3p-1,p} - 3(b-a)m'_{3p-1,3p-1} \quad (24)$$

$$t_{3p-1,3p-1} = (1-b)m'_{3p-1,p-1} + bm'_{3p-1,p} + 3(a+b-1)m'_{3p-1,3p-1} \quad (25)$$

4.4.1 Determination of $m'_{p-2,p-1}$ $m'_{p-1,p-1}$ $m'_{p,p}$ and $m'_{p+1,p}$

Elements $m'_{p-2,p-1}$ and $m'_{p-1,p-1}$ belong to the block M'_{11} and specifically they are the last two elements of the last column. We know that $M_{11}M'_{11} = I$. Let then the elements of the last column of M'_{11} be denoted by x_i , $i = 1, 2, \dots, p-1$. Thus, doing the multiplication indicated previously, and writing down the equations that arise from the last column we will have

$$\begin{aligned} 2 \cosh \phi x_1 - x_2 &= 0 \\ -x_1 + 2 \cosh \phi x_2 - x_3 &= 0 \\ &\vdots \\ -x_{p-3} + 2 \cosh \phi x_{p-2} - x_{p-1} &= 0 \\ -x_{p-2} + 2 \cosh \phi x_{p-1} &= 1 \end{aligned}$$

Or, equivalently,

$$x_0 = 0 \quad (26)$$

$$-x_{i-1} + 2 \cosh \phi x_i - x_{i+1} = 0 \quad i = 1, \dots, p-1 \quad (27)$$

$$x_p = 1 \quad (28)$$

Equation (27) is the general difference equation, whereas equations (26) and (28) can be regarded as its boundary conditions. Boundary conditions are obtained by applying the general equation for $i = 1$ and $i = p-1$ and requiring that x_0 and x_p satisfy them. If in addition x_{-1} and x_{p+1} are defined in such a way so that x_0 and x_p are given by the solution of the difference equation then this difference equation will hold for $i = 0$ and $i = p$ as well. To solve (27) we consider its characteristic equation

$$\rho^2 - (2 \cosh \phi) \rho + 1 = 0$$

whose solutions are

$$\rho_1 = \cosh \phi + \sinh \phi \quad \text{and} \quad \rho_2 = \cosh \phi - \sinh \phi$$

Thus the general solution of (27) is

$$\begin{aligned} x_i &= c_1(\cosh \phi + \sinh \phi)^i + c_2(\cosh \phi - \sinh \phi)^i \\ &= c_1 e^{i\phi} + c_2 e^{-i\phi} \end{aligned}$$

Applying the boundary condition (27) we have

$$x_0 = 0 \iff c_1 + c_2 = 0 \iff c_2 = -c_1$$

which makes the general solution of (27)

$$x_i = c_1 e^{i\phi} - c_1 e^{-i\phi} = 2c_1 \sinh i\phi \tag{29}$$

If we now apply (28) we have

$$x_p = 1 \iff 2c_1 \sinh p\phi = 1 \iff c_1 = \frac{1}{2 \sinh p\phi}$$

However, from (29) we can obtain all the elements of the last column of M'_{11} from the expression

$$x_i = \frac{\sinh i\phi}{\sinh p\phi}$$

Therefore

$$m'_{p-2,p-1} = \frac{\sinh(p-2)\phi}{\sinh p\phi} \quad m'_{p-1,p-1} = \frac{\sinh(p-1)\phi}{\sinh p\phi}$$

To go on with our analysis we observe that the elements $m'_{p,p}$ and $m'_{p+1,p}$ belong to the block M'_{22} and specifically they are the first two elements of its first column. We also note that the matrices M_{11} and M_{22} are centro-symmetric. So will then be their inverses. Thus M'_{11} and M'_{22} are also centro-symmetric. Therefore, the first two elements of the first column of M'_{22} are the same as the last two elements of the last column of M'_{11} in reverse order.

Consequently,

$$m'_{p,p} = \frac{\sinh(p-1)\phi}{\sinh p\phi} \quad m'_{p+1,p} = \frac{\sinh(p-2)\phi}{\sinh p\phi}$$

4.4.2 Determination of $m'_{3p-2,3p-2}$ $m'_{3p-1,3p-1}$

We observe that the element $m'_{3p-2,3p-2}$ belongs to the block M'_{33} and specifically it is the last element of its last column.

Again, starting with $M_{33}M'_{33} = I$ and denoting the elements of the last column of M'_{33} by x_i , $i = 1, 2, \dots, p$, we can write down the equations that arise in the last column which are the following

$$\begin{aligned} 2 \cosh \phi x_1 - x_2 &= 0 \\ -x_1 + 2 \cosh \phi x_2 - x_3 &= 0 \\ &\vdots \\ -x_{p-2} + 2 \cosh \phi x_{p-1} - x_p &= 0 \\ (-4 + 2 \cosh \phi) x_{p-1} + 2 x_p &= 1 \end{aligned}$$

Or, equivalently,

$$x_0 = 0 \tag{30}$$

$$-x_{i-1} + 2 \cosh \phi x_i - x_{i+1} = 0 \quad i = 1, \dots, p \tag{31}$$

$$(2 \cosh^2 \phi - 4 \cosh \phi + 1) x_{p-1} + x_{p+1} = \cosh \phi \tag{32}$$

Again, equation (31) is the general difference equation, whereas equations (30) and (32) are the boundary conditions. Working in a similar way as in the previous case we have to solve the difference equation (31). Its characteristic equation is

$$\rho^2 - (2 \cosh \phi) \rho + 1 = 0$$

and as we have already seen in section 4.4.1 the general solution of (31) is given by

$$x_i = c_1 e^{i\phi} + c_2 e^{-i\phi}$$

Applying the boundary condition (30) we have

$$x_0 = 0 \iff c_1 + c_2 = 0 \iff c_2 = -c_1$$

which makes the general solution of (31)

$$x_i = c_1 e^{i\phi} - c_1 e^{-i\phi} = 2c_1 \sinh i\phi \tag{33}$$

If we now apply (32) we have

$$\begin{aligned} (2 \cosh^2 \phi - 4 \cosh \phi + 1) x_{p-1} + x_{p+1} &= \cosh \phi && \iff \\ (2 \cosh^2 \phi - 4 \cosh \phi + 1) 2c_1 \sinh(p-1)\phi + 2c_1 \sinh(p+1)\phi &= \cosh \phi && \iff \\ c_1 &= \frac{\cosh \phi}{2 \sinh(p+1)\phi + 2 \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)} \end{aligned}$$

From (33) we can now obtain all the elements of the last column of M'_{33} from the expression

$$x_i = \frac{\cosh \phi \sinh i\phi}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

Therefore,

$$m'_{3p-2,3p-2} = \frac{\sinh p\phi \cosh \phi}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

Now element $m'_{3p-1,3p-1}$ belongs to the block M'_{44} and specifically it is the first element of its first column. We note that this time the matrices M_{33} and M_{44} are centro-symmetric to each other meaning that the same property is possessed by their inverses M'_{33} and M'_{44} . So the first element of the first column of M'_{44} is the same as the last element of the last column of M'_{33} . Therefore,

$$m'_{3p-1,3p-1} = \frac{\sinh p\phi \cosh \phi}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

4.4.3 Determination of $m'_{3p-2,p-1}$, $m'_{3p-2,p}$, $m'_{3p-1,p-1}$ and $m'_{3p-1,p}$

If we use equations (18) and (19) we readily have

$$(18) \iff m'_{3p-2,p-1} = \frac{a \cosh \phi (\sinh(p-2)\phi - 4 \sinh(p-1)\phi)}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

$$(19) \iff m'_{3p-2,p} = \frac{(1-a) \cosh \phi (4 \sinh(p-1)\phi - \sinh(p-2)\phi)}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

If we now use the equations (20) and (21) we have

$$(20) \iff m'_{3p-1,p-1} = \frac{(1-a) \cosh \phi (4 \sinh(p-1)\phi - \sinh(p-2)\phi)}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

$$(21) \iff m'_{3p-1,p} = \frac{a \cosh \phi (\sinh(p-2)\phi - 4 \sinh(p-1)\phi)}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

4.4.4 Elements of the Matrix T : First Approach

If we use equations (22)–(25) we have

$$(22) \iff t_{3p-2,3p-2} = \frac{(a+b-1) \cosh \phi (\sinh(p-2)\phi - 4 \sinh(p-1)\phi) + 3(a+b-1) \cosh \phi \sinh p\phi}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

$$(23) \iff t_{3p-2,3p-1} = \frac{(a-b) \cosh \phi (\sinh(p-2)\phi - 4 \sinh(p-1)\phi) - 3(b-a) \cosh \phi \sinh p\phi}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

$$(24) \iff t_{3p-1,3p-2} = \frac{(a-b) \cosh \phi (\sinh(p-2)\phi - 4 \sinh(p-1)\phi) - 3(b-a) \cosh \phi \sinh p\phi}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

$$(25) \iff t_{3p-1,3p-1} = \frac{(a+b-1) \cosh \phi (\sinh(p-2)\phi - 4 \sinh(p-1)\phi) + 3(a+b-1) \cosh \phi \sinh p\phi}{\sinh(p+1)\phi + \sinh(p-1)\phi (2 \cosh^2 \phi - 4 \cosh \phi + 1)}$$

4.4.5 Optimum Values of a and b

Now we have all the elements that will allow us to use equations (10) and (11). These equations give successively

$$(10) \iff (a+b-1)(\sinh(p-2)\phi - 4 \sinh(p-1)\phi + 3 \sinh p\phi) = 0$$

$$\iff a+b-1 = 0 \tag{34}$$

$$\begin{aligned}
(11) \quad &\iff [(a+b-1)((\sinh(p-2)\phi - 4\sinh(p-1)\phi + 3\sinh p\phi)]^2 \\
&\quad - [(a-b)(\sinh(p-2)\phi - 4\sinh(p-1)\phi + 3\sinh p\phi)]^2 = 0 \\
&\iff (a+b-1)^2 - (a-b)^2 = 0 \\
&\iff (2a-1)(2b-1) = 0 \\
&\iff a = \frac{1}{2} \quad \text{or} \quad b = \frac{1}{2} \tag{35}
\end{aligned}$$

Equation (34) was obtained since it can be proved that the second factor in the product preceding it is different from zero. Now, in view of (34) either solution of equation (35) gives as the optimum values for a and b the following

$$a = \frac{1}{2} \quad \text{and} \quad b = \frac{1}{2}$$

REMARK

For the optimum values of a and b just determined we observe that the four elements $t_{3p-2,3p-2}$, $t_{3p-2,3p-1}$, $t_{3p-1,3p-2}$, $t_{3p-1,3p-1}$ of T become zero. As a result of this, it can be checked out that $T^2 = 0$ (!) implying that in the absence of round-off errors the exact solution of the linear system will be obtained after **two** iterations.

4.5 Determination of a and b : Second Approach

A similar analysis to the one before can be done. Specifically, we substitute the elements of the matrix N in the equations (14), (15), (16) and (17) and obtain equivalently

$$t_{3p-2,3p-2} = bm'_{3p-2,p-1} + (1-b)m'_{3p-2,p} + 3am'_{3p-2,3p-2} \quad (36)$$

$$t_{3p-2,3p-1} = (1-b)m'_{3p-2,p-1} + bm'_{3p-2,p} - 3(1-a)m'_{3p-2,3p-2} \quad (37)$$

$$t_{3p-1,3p-2} = bm'_{3p-1,p-1} + (1-b)m'_{3p-1,p} - 3(1-a)m'_{3p-1,3p-1} \quad (38)$$

$$t_{3p-1,3p-1} = (1-b)m'_{3p-1,p-1} + bm'_{3p-1,p} + 3am'_{3p-1,3p-1} \quad (39)$$

4.5.1 Determination of $m'_{p-2,p-1}$ $m'_{p-1,p-1}$ $m'_{p,p}$ and $m'_{p+1,p}$

Here we follow exactly the same steps as in section 4.4.1. Consequently,

$$m'_{p-2,p-1} = \frac{\sinh(p-2)\phi}{\sinh p\phi} \quad m'_{p-1,p-1} = \frac{\sinh(p-1)\phi}{\sinh p\phi}$$

$$m'_{p,p} = \frac{\sinh(p-1)\phi}{\sinh p\phi} \quad m'_{p+1,p} = \frac{\sinh(p-2)\phi}{\sinh p\phi}$$

4.5.2 Determination of $m'_{3p-2,3p-2}$ and $m'_{3p-1,3p-1}$

The element $m'_{3p-2,3p-2}$ belongs to the block M'_{33} and specifically it is the last element of its last column. Since we know that $M_{33}M'_{33} = I$ we denote the elements of the last column of M'_{33} by x_i , $i = 1, 2, \dots, p$, and multiplying out we can write down the equations that arise from the last column. Thus we have

$$\begin{aligned} 2 \cosh \phi x_1 - x_2 &= 0 \\ -x_1 + 2 \cosh \phi x_2 - x_3 &= 0 \\ &\vdots \\ -x_{p-2} + 2 \cosh \phi x_{p-1} - x_p &= 0 \\ -2x_{p-1} + 2 \cosh \phi x_p &= 1 \end{aligned}$$

or, equivalently,

$$x_0 = 0 \quad (40)$$

$$-x_{i-1} + 2 \cosh \phi x_i - x_{i+1} = 0 \quad i = 1, \dots, p \quad (41)$$

$$-x_{p-1} + x_{p+1} = 1 \quad (42)$$

Equation (41) is the general difference equation, whereas equations (40) and (42) can be regarded as boundary conditions which are obtained by applying the general equation for $i = 1$ and $i = p$ and requiring that x_0 and x_{p+1} satisfy them.

The characteristic equation of (41) is

$$\rho^2 - (2 \cosh \phi) \rho + 1 = 0$$

and as we have already seen in section 4.4.1 the general solution of (41) is given by

$$x_i = c_1 e^{i\phi} + c_2 e^{-i\phi}$$

Applying now the boundary condition (40) we have

$$x_0 = 0 \iff c_1 + c_2 = 0 \iff c_2 = -c_1$$

which makes the general solution of (41) be as follows

$$x_i = c_1 e^{i\phi} - c_1 e^{-i\phi} = 2c_1 \sinh i\phi \quad (43)$$

If we apply (42) we have

$$\begin{aligned} -x_{p-1} + x_{p+1} &= 1 && \iff \\ -2c_1 \sinh(p-1)\phi + 2c_1 \sinh(p+1)\phi &= 1 && \iff \\ c_1 &= \frac{1}{2 \sinh(p+1)\phi - 2 \sinh(p-1)\phi} \end{aligned}$$

From (43) we can obtain all the elements of the last column of M'_{33} from the expression

$$x_i = \frac{\sinh i\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

So,

$$m'_{3p-2,3p-2} = \frac{\sinh p\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

The element $m'_{3p-1,3p-1}$ belongs to the block M'_{44} and specifically it is the first element of its first column. The matrices M_{33} and M_{44} are centro-symmetric to each other and as we saw in section 4.4.1 we can deduce that M'_{33} and M'_{44} are also centro-symmetric to each other. Therefore, the first element of the first column of M'_{44} is the same as the last element of the last column of M'_{33} . Consequently,

$$m'_{3p-1,3p-1} = \frac{\sinh p\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

4.5.3 Determination of $m'_{3p-2,p-1}$, $m'_{3p-2,p}$, $m'_{3p-1,p-1}$ and $m'_{3p-1,p}$

If we use the equations (18)–(21) we have

$$(18) \iff m'_{3p-2,p-1} = a \frac{\sinh(p-2)\phi - 4\sinh(p-1)\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

$$(19) \iff m'_{3p-2,p} = -(1-a) \frac{\sinh(p-2)\phi - 4\sinh(p-1)\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

$$(20) \iff m'_{3p-1,p-1} = -(1-a) \frac{\sinh(p-2)\phi - 4\sinh(p-1)\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

$$(21) \iff m'_{3p-1,p} = a \frac{\sinh(p-2)\phi - 4\sinh(p-1)\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

4.5.4 Elements of the Matrix T : Second Approach

If we use the equations (36)–(39) we have

$$(36) \iff t_{3p-2,3p-2} = \frac{(a+b-1)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) + 3a\sinh p\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

$$(37) \iff t_{3p-2,3p-1} = \frac{(a-b)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) - 3(1-a)\sinh p\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

$$(38) \iff t_{3p-1,3p-2} = \frac{(a-b)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) - 3(1-a)\sinh p\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

$$(39) \iff t_{3p-1,3p-1} = \frac{(a+b-1)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) + 3a\sinh p\phi}{\sinh(p+1)\phi - \sinh(p-1)\phi}$$

4.5.5 Optimum Values of a and b

Now we can try to find the solutions of the system of equations (10) and (11). For this we successively have

$$(10) \iff (a+b-1)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) + 3a\sinh p\phi = 0 \quad (44)$$

$$(11) \iff [(a+b-1)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) + 3a\sinh p\phi]^2 - [(a-b)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) - 3(1-a)\sinh p\phi]^2 = 0$$

$$\iff [(2a-1)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) + 3(2a-1)\sinh p\phi] [(2b-1)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) + 3\sinh p\phi] = 0$$

$$\iff [(2a-1)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) + 3\sinh p\phi] [(2b-1)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) + 3\sinh p\phi] = 0$$

$$\iff (2a-1)[(2b-1)(\sinh(p-2)\phi - 4\sinh(p-1)\phi) + 3\sinh p\phi] = 0$$

$$\iff a = \frac{1}{2}$$

or

$$(45)$$

$$b = \frac{\sinh(p-2)\phi - 4\sinh(p-1)\phi - 3\sinh p\phi}{2(\sinh(p-2)\phi - 4\sinh(p-1)\phi)}$$

Using either solution of equation (45) into (44) we find out that the optimum values for a and b are

$$a = \frac{1}{2} \quad \text{and} \quad b = \frac{\sinh(p-2)\phi - 4\sinh(p-1)\phi - 3\sinh p\phi}{2(\sinh(p-2)\phi - 4\sinh(p-1)\phi)}$$

REMARK

In this case it is also observed that for the optimum values of a and b above the four elements $t_{3p-2,3p-2}$, $t_{3p-2,3p-1}$, $t_{3p-1,3p-2}$, $t_{3p-1,3p-1}$ of T become zero and, therefore, $T^2 = 0$! Again, as in the first approach, in the absence of round-off errors the exact solution of the linear system will be obtained after **two** iterations.

5 The Poisson Equation

The Poisson equation is a particular case of the Helmholtz equation studied in the previous two chapters. The Poisson equation was the very first part of our study and so we present very briefly the corresponding results here. The PDE problem for the Poisson equation is given by

$$\begin{aligned} -u'' &= f & \text{in } \Omega &\equiv (0, 1) \\ u(0) &= \alpha & \text{and } u(1) &= \beta \end{aligned} \tag{46}$$

where α and β are the given boundary values.

Since, as was already mentioned, the Poisson equation can be obtained from the Helmholtz one (4) with $c = 0$, and therefore the diagonal element d in the matrices M will be 2, all the results in this present case can also be obtained from the ones in the the previous case adopting limiting processes letting $c \rightarrow 0^+$.

5.1 Determination of a and b : First Approach

Again, the elements $t_{3p-2,3p-2}$, $t_{3p-2,3p-1}$, $t_{3p-1,3p-2}$ and $t_{3p-1,3p-1}$ will be given by equations (22)–(25).

What we now need are the necessary elements of the matrix M' .

5.1.1 Determination of the Elements of the Matrix M'

The elements $m'_{p-2,p-1}$ and $m'_{p-1,p-1}$ belong to the block M'_{11} . To compute these elements we write down the equations that correspond to the last column of the product $M_{11}M'_{11} = I$, we have

$$\begin{aligned} 2x_1 - x_2 &= 0 \\ -x_1 + 2x_2 - x_3 &= 0 \\ &\vdots \\ -x_{p-3} + 2x_{p-2} - x_{p-1} &= 0 \\ -x_{p-2} + 2x_{p-1} &= 1 \end{aligned}$$

or, equivalently,

$$x_0 = 0 \tag{47}$$

$$-x_{i-1} + 2x_i - x_{i+1} = 0 \quad i = 1, \dots, p-1 \tag{48}$$

$$x_p = 1 \tag{49}$$

Again, the boundary conditions (47) and (49) are obtained by applying the general equation for $i = 1$ and $i = p - 1$ and demanding that x_0 and x_p satisfy them.

The characteristic equation of the general difference equation (48) is

$$\rho^2 - 2\rho + 1 = 0$$

whose solutions are

$$\rho_1 = \rho_2 = 1$$

Thus the general solution of (48) is

$$\begin{aligned} x_i &= c_1\rho_1^i + c_2i\rho_1^i = c_11^i + c_2i1^i \\ &= c_1 + c_2i \end{aligned}$$

Applying the boundary condition (47) we have

$$x_0 = 0 \iff c_1 + c_2 \cdot 0 = 0 \iff c_1 = 0$$

which makes the general solution of (48) be given by

$$x_i = c_2i \tag{50}$$

If we now apply (49) we have

$$x_p = 1 \iff c_2p = 1 \iff c_2 = \frac{1}{p}$$

From (50) we can obtain all the elements of the last column of M'_{11} from the expression

$$x_i = \frac{i}{p}$$

So

$$m'_{p-2,p-1} = \frac{p-2}{p} \quad m'_{p-1,p-1} = \frac{p-1}{p}$$

We could continue in the same way and show how we could find all the elements of M' we need. However, at this point we can comment that in our case we can find all the elements we need, by using the various expressions we have already found in section 4.4, and determining the limits of the corresponding expressions as $\phi \rightarrow 0^+$. Thus we have

$$\begin{aligned}
 m'_{3p-2,p-1} &= \frac{a}{2}(-3p+2) \\
 m'_{3p-2,p} &= -\frac{1-a}{2}(-3p+2) \\
 m'_{3p-2,3p-2} &= \frac{p}{2} \\
 m'_{3p-1,p-1} &= -\frac{1-a}{2}(-3p+2) \\
 m'_{3p-1,p} &= \frac{a}{2}(-3p+2) \\
 m'_{3p-1,3p-1} &= \frac{p}{2}
 \end{aligned}$$

5.1.2 Elements of the Matrix T : First Approach

If we use equations (22)–(25) we have

$$(22) \iff t_{3p-2,3p-2} = a + b - 1$$

$$(23) \iff t_{3p-2,3p-1} = a - b$$

$$(24) \iff t_{3p-1,3p-2} = a - b$$

$$(25) \iff t_{3p-1,3p-1} = a + b - 1$$

5.1.3 Optimum Values of a and b

We will try to see if we can make equations (10) and (11) be satisfied simultaneously.

$$(10) \iff a + b - 1 = 0 \tag{51}$$

$$\begin{aligned}
 (11) \iff (a + b - 1)^2 - (a - b)^2 &= 0 \iff (2a - 1)(2b - 1) = 0 \\
 \iff a = \frac{1}{2} \quad \text{or} \quad b = \frac{1}{2} & \tag{52}
 \end{aligned}$$

From the system of equations (52) and (51) we readily find that the optimum values for a and b are

$$a = \frac{1}{2} \quad \text{and} \quad b = \frac{1}{2}$$

We remark that this result was somehow expected since the values found for the optimal ones for a and b in the case of Helmholtz equation were independent of ϕ .

5.2 Determination of a and b : Second Approach

The elements $t_{3p-2,3p-2}$, $t_{3p-2,3p-1}$, $t_{3p-1,3p-2}$ and $t_{3p-1,3p-1}$ will be given again by equations (36)–(39).

5.2.1 Determination of the Elements of the Matrix M'

As the blocks M_{11} M_{22} M_{33} and M_{44} are identically the same with the ones of the previous approach, we do not need to find the necessary elements of their inverse matrices again. We can refer to the corresponding expressions in section 5.1.1 instead.

5.2.2 Elements of the Matrix T : Second Approach

If we use the equations (36)–(39) we have

$$(36) \iff t_{3p-2,3p-2} = \frac{(a+b-1)(-3p+2) + 3ap}{2}$$

$$(37) \iff t_{3p-2,3p-1} = \frac{(a-b)(-3p+2) - 3(1-a)p}{2}$$

$$(38) \iff t_{3p-1,3p-2} = \frac{(a-b)(-3p+2) - 3(1-a)p}{2}$$

$$(39) \iff t_{3p-1,3p-1} = \frac{(a+b-1)(-3p+2) + 3ap}{2}$$

5.2.3 Optimum Values of a and b

If we try to find the solution of (10) and (11) we have

$$(10) \iff (a + b - 1)(-3p + 2) + 3ap = 0 \tag{53}$$

$$\begin{aligned} (11) \iff & [(a + b - 1)(-3p + 2) + 3ap]^2 - [(a - b)(-3p + 2) - 3(1 - a)p]^2 = 0 \\ \iff & (4a - 2)(4b - 6bp + 6p - 2) = 0 \\ \iff & a = \frac{1}{2} \quad \text{or} \quad b = \frac{3p - 1}{3p - 2} \end{aligned} \tag{54}$$

The solution to the system of equations (54) and (53) gives that the optimum values for a and b are

$$a = \frac{1}{2} \quad \text{and} \quad b = \frac{3p - 1}{3p - 2}$$

6 Numerical Experiments

In this section numerical examples are presented to confirm the theoretical results given above. In all computations below, we apply second order finite difference discretizations on uniform grids in each of the two subdomains.

The resulting linear systems of algebraic equations are solved by banded Gaussian Elimination. Single precision is used for all calculations. The initial guesses are always taken to be zero.

The errors are evaluated in the L^∞ norm of the vector of the differences at all the points of the discretization of the values obtained after the second Neumann iteration from the theoretical values. For all the numerical results in the examples the interface is at $x = 0.5$.

Note that the subdomain problems at each iteration level in our method are completely independent and thus parallelizable.

We select the following model problems on $\Omega \equiv (0, 1)$.

$$\begin{aligned} -u'' + 0.5u &= f, & \text{in } \Omega, & u = g, & \text{in } \partial\Omega, \\ -u'' &= f, & \text{in } \Omega, & u = g, & \text{in } \partial\Omega \end{aligned}$$

Example 1: The functions f and g are chosen such that the exact solution is:

$$u(x) = \sin\left(\frac{\pi}{2}x\right).$$

Example 2: The functions f and g are chosen such that the exact solution is:

$$u(x) = x(1 - x).$$

Example 3: The functions f and g are chosen such that the exact solution is:

$$u(x) = -\frac{e^{2x}}{4}.$$

In tables (1)-(6) we present the results obtained for the Helmholtz equation whereas in tables (7)-(12) we present those for the Poisson equation.

6.1 Results from the First Approach–Helmholtz

Table 1: Example 1, Helmholtz, First Approach

Test Problem 1. First Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$1.1211E - 4$	$2.6643E - 5$	$5.8472E - 5$
2	$1.1211E - 4$	$2.7060E - 5$	$5.9902E - 5$

Table 2: Example 2, Helmholtz, First Approach

Test Problem 2. First Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$2.6822E - 7$	$5.2154E - 7$	$1.9356E - 5$
2	$2.9802E - 7$	$5.2154E - 7$	$1.9356E - 5$

Table 3: Example 3, Helmholtz, First Approach

Test Problem 3. First Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$4.0793E - 4$	$8.6307E - 5$	$5.7578E - 5$
2	$4.0793E - 4$	$8.6000E - 5$	$5.7578E - 5$

6.2 Results from the Second Approach–Helmholtz

Table 4: Example 1, Helmholtz, Second Approach

Test Problem 1. Second Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$5.5645E - 3$	$6.3955E - 4$	$1.2171E - 4$
2	$5.5646E - 3$	$6.3955E - 4$	$1.2272E - 4$

Table 5: Example 2, Helmholtz, Second Approach

Test Problem 2. Second Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$2.6822E - 7$	$7.4505E - 7$	$1.9356E - 5$
2	$2.9802E - 7$	$5.5134E - 7$	$1.9356E - 5$

Table 6: Example 3, Helmholtz, Second Approach

Test Problem 3. Second Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$1.1623E - 2$	$1.3067E - 3$	$1.6671E - 4$
2	$1.1623E - 2$	$1.3064E - 3$	$1.6820E - 4$

6.3 Results from the First Approach–Poisson

Table 7: Example 1, Poisson, First Approach

Test Problem 1. First Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$1.1062E - 4$	$2.9683E - 5$	$5.3644E - 6$
2	$1.1068E - 4$	$2.9683E - 5$	$6.1988E - 6$

Table 8: Example 2, Poisson, First Approach

Test Problem 2. First Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$5.9604E - 8$	$2.3841E - 7$	$3.1441E - 6$
2	$5.9604E - 8$	$2.3841E - 7$	$3.1441E - 6$

Table 9: Example 3, Poisson, First Approach

Test Problem 3. First Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$4.1782E - 4$	$9.2864E - 5$	$1.3947E - 5$
2	$4.1782E - 4$	$9.2864E - 5$	$1.3947E - 5$

6.4 Results from the Second Approach–Poisson

Table 10: Example 1, Poisson, Second Approach

Test Problem 1. Second Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$5.8453E - 3$	$6.6816E - 4$	$6.3419E - 5$
2	$5.8450E - 3$	$6.6846E - 4$	$6.6339E - 5$

Table 11: Example 2, Poisson, Second Approach

Test Problem 2. Second Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$5.9604E - 8$	$2.3841E - 7$	$3.1441E - 6$
2	$1.0430E - 7$	$4.9173E - 7$	$3.7252E - 6$

Table 12: Example 3, Poisson, Second Approach

Test Problem 3. Second Approach			
iteration	Grid size $\frac{1}{10}$	Grid size $\frac{1}{32}$	Grid size $\frac{1}{100}$
1	$1.2203E - 2$	$1.3688E - 3$	$1.4823E - 4$
2	$1.2203E - 2$	$1.3688E - 3$	$1.4823E - 4$

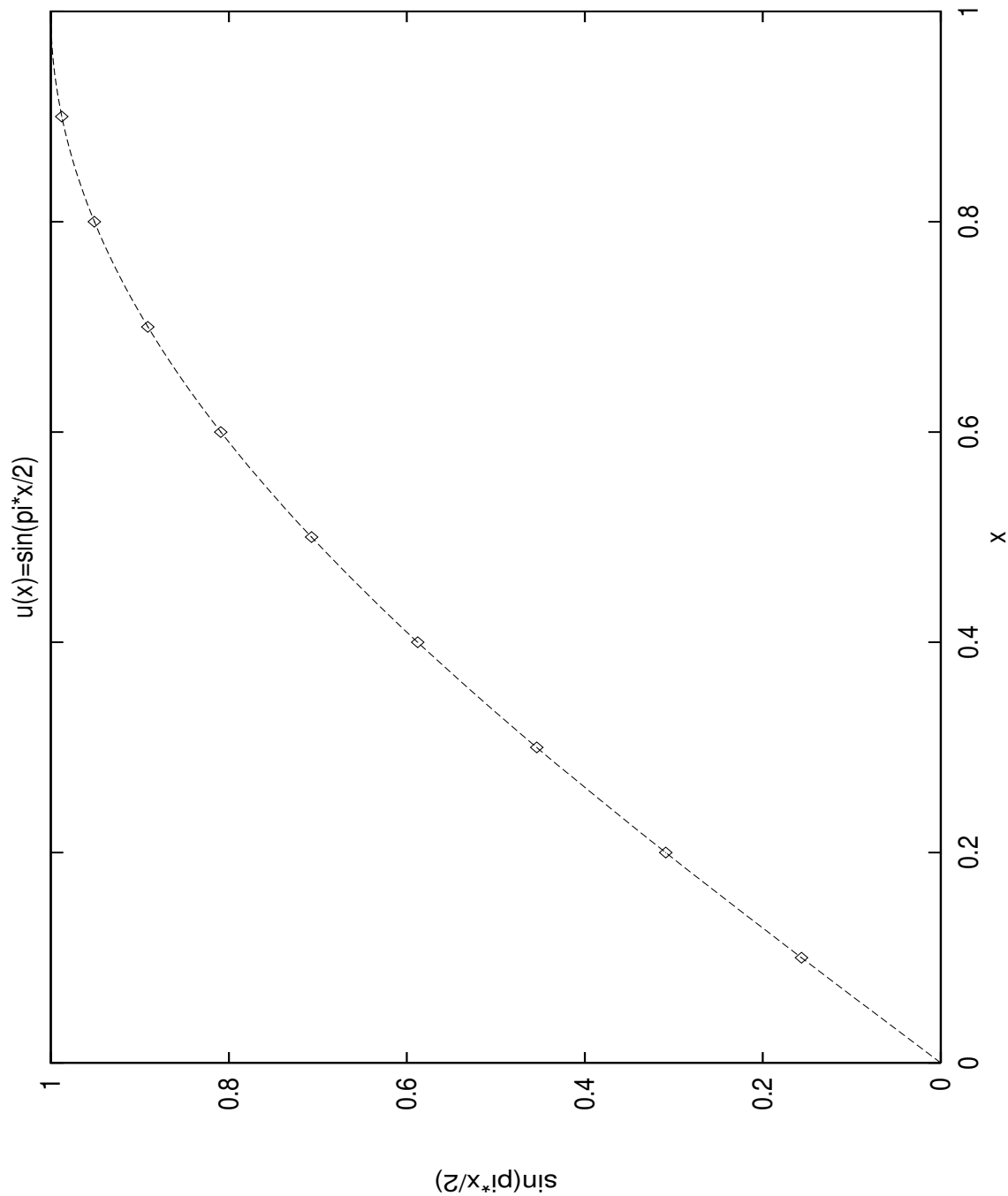


Figure 2: Example 1, Helmholtz equation

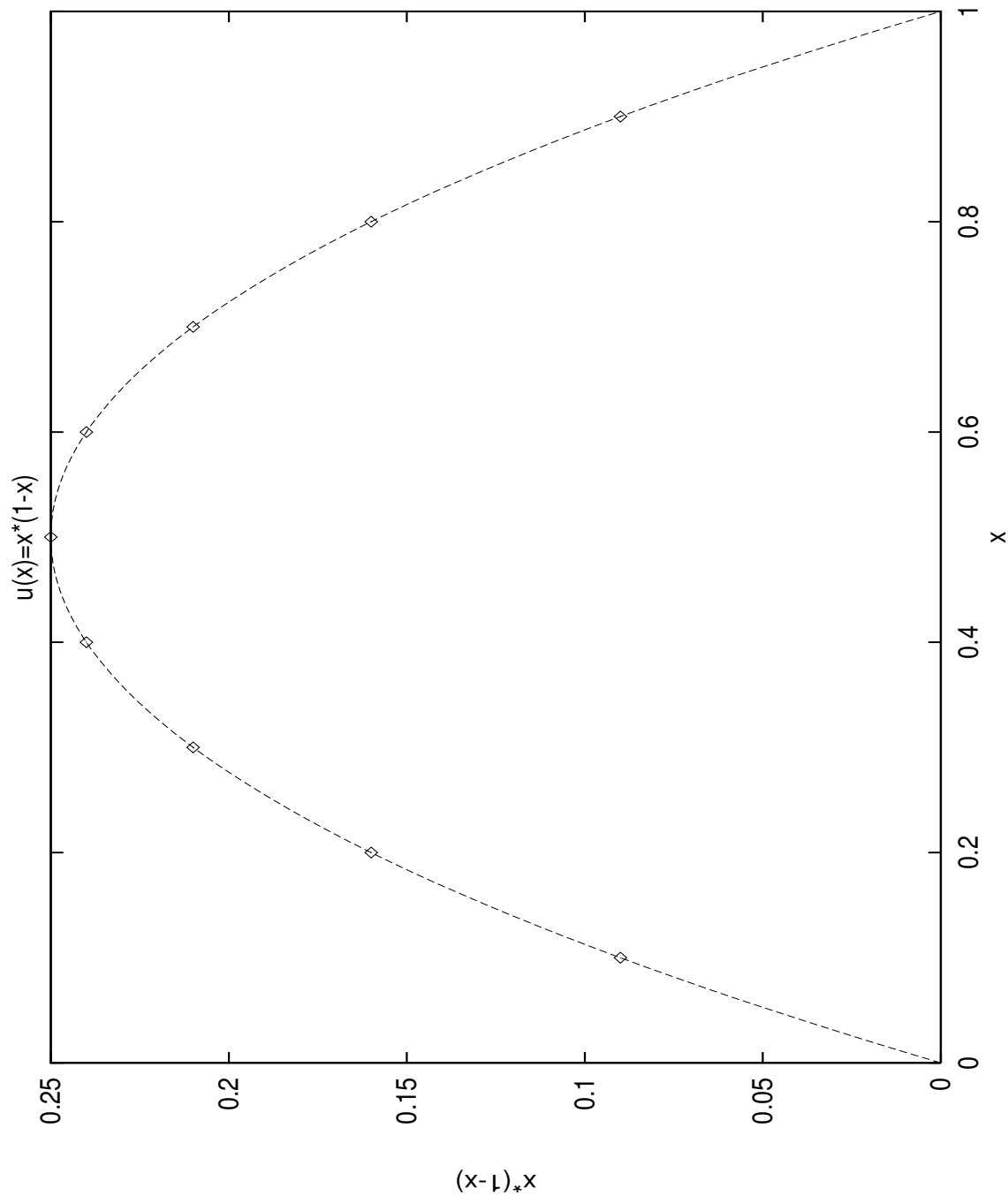


Figure 3: Example 2, Helmholtz equation

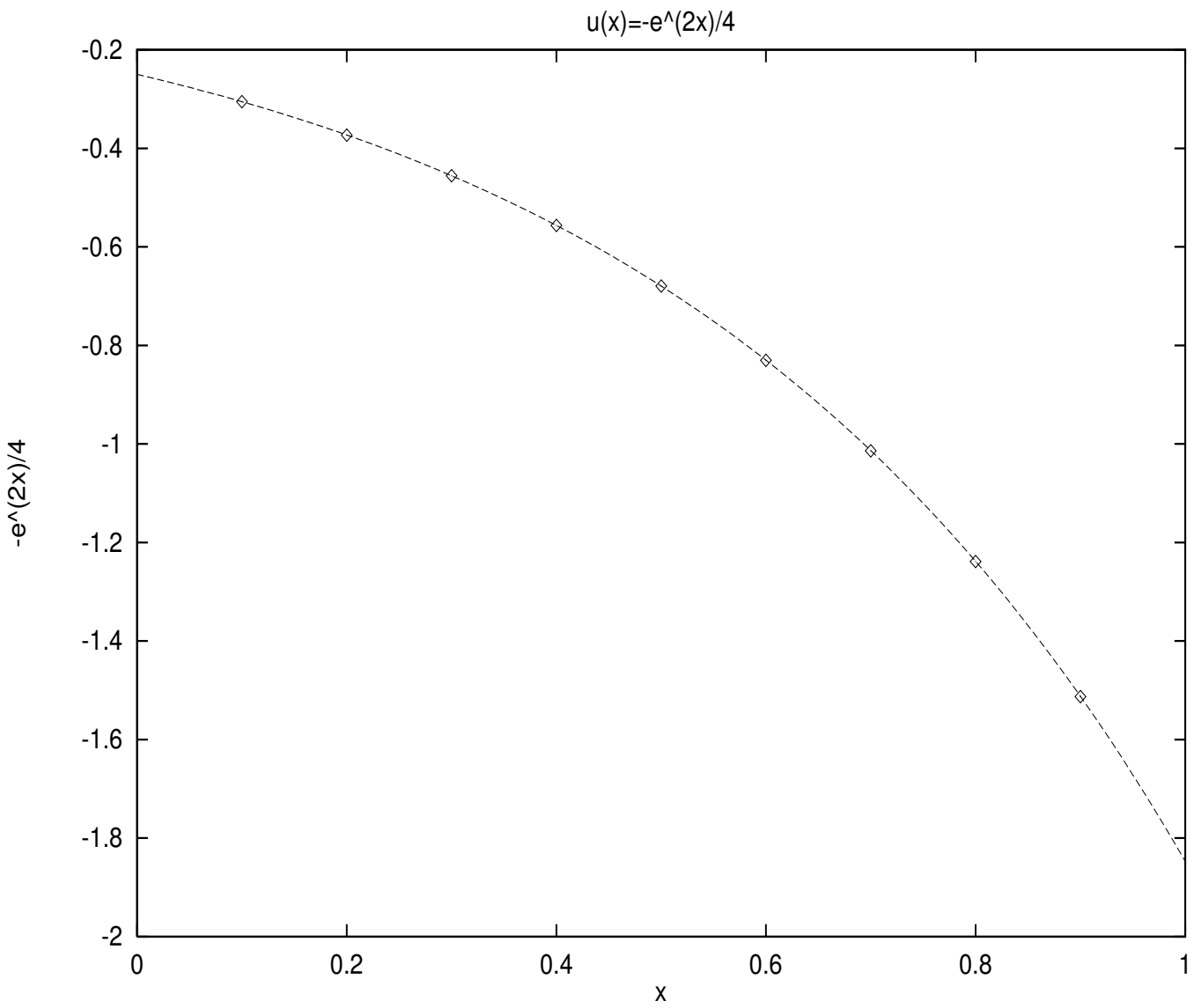


Figure 4: Example 3, Helmholtz equation
48

7 Conclusions

From the theoretical analysis done in this work it became clear that the best possible asymptotic convergence of the iterative schemes for both approaches was achieved. More specifically, as was remarked in sections 4.4, 4.5, in the absence of round-off errors, the exact solutions of the corresponding linear systems are obtained after two iterations.

However, in all of the experiments we tried and despite the presence of round-off errors and the single precision arithmetic used, we observe that the solutions obtained are much better than what one would expect. The errors in each one of them are even better than the order of accuracy (the order of the truncation errors) which for the grid sizes chosen should be 10^{-2} , 10^{-3} , 10^{-4} , respectively, for the three test examples.

The very good order of accuracy obtained in all the experiments tried seems to be achieved after the very first iteration. The second iteration gives results which are very close to the ones already obtained after the first one.

Out of the two approaches analyzed and studied in this work, the experiments show that the first one seems to give slightly better results than the second approach.

The way our theory was developed enables us to solve the discrete analog of the differential problem at the two half-iteration levels (Dirichlet and Neumann iterations) for the two subdomains independently. Thus this inherent parallelism can be fully exploited when one moves on to considering more than two subdomains and/or one dimension, an issue which is open for further research at the linear algebra level. The theoretical difficulties for the determination of the optimum parameters might have become obvious even at this "elementary" one-dimension, two-equally-uniformly-sized-subdomain-problem. So it is fully understandable that the theoretical issues one has to overcome will be of a much more difficult nature.

A Source Code developed in Fortran 77

The linear systems were solved using routines from the Linpack Package.

A.1 First Approach

```
C =====
C | main.f
C -----
C | Implementation of the first approach of the domain decomposition
C | method proposed. Solves numerically the PDE problem :
C |  $-u''|x + c*u = f(x)$ , defined in an interval (a,b), with  $c>0$  and
C | Dirichlet Conditions  $u(a)=\alpha$  and  $u(b)=\beta$  (Helmholtz)
C | In the particular case where  $c=0$ , the corresponding Poisson
C | equation is solved.
C =====

      program firstapproach
      integer lda, lda2
      parameter (lda=250, lda2=2*lda)
      integer dec, i, irun, it, n, p
      real a, b, c, alpha, beta, al, bl
      real h, norm, rp, phi
      real sol(lda2), grdx(lda)
      real rhs1(lda), rhs2(lda), rhs3(lda), rhs4(lda)
      real lhs1(lda,lda), lhs3(lda,lda),lhs4(lda,lda)

      write (*,*) 'Give the parameter c of the Helmholtz Eqn'
      read (*,*) c

      write (*,*) 'Give the interval (a,b)'
      write (*,*) 'a:='
      read (*,*) a
      write (*,*) 'b:='
      read (*,*) b
```

```
write (*,*) 'Give the Dirichlet boundary conditions'  
write (*,*) 'alpha=u(a):='  
read (*,*) alpha  
write (*,*) 'beta=u(b):='  
read (*,*) beta  
  
write (*,*) 'Which test problem to run?'  
write (*,*) 'u(x)=sin(pi*x/2) (1 to accept)'  
write (*,*) 'u(x)=x*(1-x) (2 to accept)'  
write (*,*) 'u(x)=-e^(2*x)/4 (3 to accept)'  
read (*,*) irun  
if ((irun.NE.1).AND.(irun.NE.2).AND.(irun.NE.3)) goto 1000  
  
10 write (*,*) 'Give number of discritisation points n, 0 to stop'  
write (*,*) 'n:='  
read (*,*) n  
p=(n+1)/2  
rp=real(p)  
h=(b-a)/(n+1)  
if (n .EQ. 0) goto 1000  
  
write (*,*) 'Let a:=1/2? (1 to accept)'  
read (*,*) dec  
if ( dec .EQ. 1 ) then  
    al=0.5  
else  
    write (*,*) 'al:='  
    read (*,*) al  
endif  
write (*,*) 'Let b:=(optimum value)? (1 to accept)'  
read (*,*) dec  
if ( dec .EQ. 1 ) then  
    tmp=1. + c*h*h/2.  
    phi=acosh(tmp)  
    snhpm2=sinh((rp-2)*phi)  
    snhpm1=sinh((rp-1)*phi)
```

```

        snhp=sinh(rp*phi)
        bl=(snhpm2-4.*snhpm1-3*snhp)/(2*(snhpm2-4.*snhpm1))
    else
        write (*,*) 'bl:= '
        read (*,*) bl
    endif
    if ((dec .EQ. 1) .AND. (c .EQ. 0)) bl=(3.*rp-1.)/(3.*rp-2.)

    do 20 i=1,n
        grdx(i)=a+i*h
20    continue

C Form the left hand side matrix
    call lhsgen(lda,p,al,bl,c,h,lhs1,lhs3,lhs4)

C Form the right hand side matrix
    call rhsgen(lda2,p,h,alpha,beta,irun,c,grdx,rhs1,rhs2,rhs3,rhs4)

C Iteration routine
    call = second(t1)
    call iterate(lda,p,al,bl,irun,grdx,lhs1,lhs3,lhs4,rhs1,rhs2,
&               rhs3, rhs4,sol,it,c)
    call = second(t2)

C Find L_infinity norm
    norm = findnorm(n,irun,c,sol,grdx)

C Print out results
    write (*,*) '-----',
    write (*,*) '|Iter',it,'Norm=',norm,'Time=',t2-t1
    do 30 i=1,n
        write (*,*) grdx(i),sol(i)
30    continue

    goto 10
1000 stop
    end

```

```

C =====
C | lhsgen.f
C -----
C | Form the left hand side lhs:= 4 block tridiag(-1,2+c*h^2,-1)
C | The first two blocks are (p-1)x(p-1)
C | The next two are pxp having (2+c*h^2,-2) in the last row of the
C | third and first row of the fourth
C =====
      subroutine lhsgen(lda,p,al,bl,c,h,lhs1,lhs3,lhs4)
      integer lda, p
      integer j
      real al, bl, c, h
      real lhs1(lda,1), lhs3(lda,1), lhs4(lda,1)
      real diag

      diag=2. + c*h*h

      do 10 j=1,p
         call vfill(p,lhs1(1,j),0.)
         call vfill(p,lhs3(1,j),0.)
         call vfill(p,lhs4(1,j),0.)
10      continue

      lhs1(1,1)=diag
      lhs1(1,2)=-1.
      do 20 i=2,p-2
         lhs1(i,i-1)=-1.
         lhs1(i,i)=diag
         lhs1(i,i+1)=-1.
20      continue
      lhs1(p-1,p-2)=-1.
      lhs1(p-1,p-1)=diag

      lhs3(1,1)=diag
      lhs3(1,2)=-1.
      do 30 i=2,p-1
         lhs3(i,i-1)=-1.

```

```
        lhs3(i,i)=diag
        lhs3(i,i+1)=-1.
30      continue
        lhs3(p,p-1)=-4.+diag
        lhs3(p,p)=2

        lhs4(1,1)=2
        lhs4(1,2)=-4.+diag
        do 40 i=2,p-1
            lhs4(i,i-1)=-1.
            lhs4(i,i)=diag
            lhs4(i,i+1)=-1.
40      continue
        lhs4(p,p-1)=-1.
        lhs4(p,p)=diag

        return
        end
```

```
C =====
C | rhsgen.f
C -----
C | Form the right hand side standard vector rhs:= h^2 * f(x)
C | f(x) given by function f, depending on the test problem to run
C =====
      subroutine rhsgen(lda2,p,h,alpha,beta,irun,c,grdx,
&                      rhs1,rhs2,rhs3,rhs4)
      integer lda2, p, irun
      integer i, n
      real alpha, beta, h
      real hs
      real grdx(1), rhs1(1), rhs2(1), rhs3(1), rhs4(1)
      real fc(lda2)

      hs=h*h
      n=2*p-1

      call vfill(p,rhs1,0.)
      call vfill(p,rhs2,0.)
      call vfill(p,rhs3,0.)
      call vfill(p,rhs4,0.)

      call f(n,irun,c,grdx,fc)

C First Dirichlet Iteration
      do 10 i=1, p-1
          rhs1(i)=hs*fc(i)
10      continue
      rhs1(1)=rhs1(1) + alpha

C Second Dirichlet Iteration
      do 20 i=1, p-1
          rhs2(i)=hs*fc(i+p)
20      continue
      rhs2(p-1)=rhs2(p-1) + beta
```

```
C First Neumann Iteration
  do 30 i=1, p-1
    rhs3(i)=rhs1(i)
30  continue
    rhs3(p)=rhs3(p-1)

C Second Neumann Iteration
  do 40 i=2,p
    rhs4(i)=rhs2(i-1)
40  continue
    rhs4(1)=rhs4(2)

  return
  end
```

```

C =====
C | iterate.f
C -----
C | Iterative Procedure as it comes from the Iterative schemes.
C | Only 2 full iterations are done.
C =====
      subroutine iterate(lda,p,al,bl,irun,grdx,lhs1,lhs3,lhs4,
&          rhs1,rhs2,rhs3,rhs4,un,it,c)
      parameter (itmax=2)
      integer lda, p, it, irun
      integer j, info, pm1
      real al, bl, c, grdx(1)
      real lhs1(lda,1), lhs3(lda,1), lhs4(lda,1)
      real rhs1(1), rhs2(1), rhs3(1), rhs4(1), un(1)
      real up1(lda), up2(lda), up3(lda), up4(lda)
      real uc1(lda), uc2(lda), uc3(lda), uc4(lda)
      real diag(lda), updiag(lda), botdiag(lda)
      real pnorm

      pm1=p-1
      n=2*p-1

      call vfill(pm1, up1, 0.)
      call vfill(pm1, up2, 0.)
      call vfill(p, up3, 0.)
      call vfill(p, up4, 0.)

C up: the previous solution, uc: the current one
      do 70 it=1, itmax

C First Dirichlet Iteration
      call scopy (pm1, rhs1, 1, un, 1)
      un(pm1)= un(pm1) + bl*up3(p) + (1.-bl)*up4(1)
      do 10 j=1,pm1
         diag(j)=lhs1(j,j)
         updiag(j)=lhs1(j,j+1)
10      continue

```

```

        updiag(pm1)=0.
        call spts1(pm1, diag, updiag, un)
        call scopy (pm1, un, 1, uc1, 1)

C Second Dirichlet Iteration
        call scopy (pm1, rhs2, 1, un, 1)
        un(1)= un(1) + (1.-bl)*up3(p) + bl*up4(1)
        do 20 j=1,pm1
            diag(j)=lhs1(j,j)
            updiag(j)=lhs1(j,j+1)
20        continue
        updiag(pm1)=0.
        call spts1(pm1, diag, updiag, un)
        call scopy (pm1, un, 1, uc2, 1)

C First Neumann Iteration
        call scopy (p, rhs3, 1, un, 1)
        un(p)= un(p) + 3.*(al+bl-1.)*up3(p) - 3.*(bl-al)*up4(1)
&           + al*uc1(p-2) - 4.*al*uc1(pm1)
&           + 4.*(1.-al)*uc2(1) - (1.-al)*uc2(2)
        do 30 j=1,p
            botdiag(j)=lhs3(j,j-1)
            diag(j)=lhs3(j,j)
            updiag(j)=lhs3(j,j+1)
30        continue
        botdiag(1)=0.
        updiag(p)=0.
        call sgts1(p, botdiag, diag, updiag, un, info)
        call scopy (p, un, 1, uc3, 1)

C Second Neumann Iteration
        call scopy (p, rhs4, 1, un, 1)
        un(1)= un(1) - 3.*(bl-al)*up3(p) + 3.*(al+bl-1.)*up4(1)
&           - (1.-al)*uc1(p-2) + 4.*(1.-al)*uc1(pm1)
&           - 4.*al*uc2(1) + al*uc2(2)
        do 40 j=1,p
            botdiag(j)=lhs4(j,j-1)

```

```
        diag(j)=lhs4(j,j)
        updiag(j)=lhs4(j,j+1)
40      continue
        botdiag(1)=0.
        updiag(p)=0.
        call sgtsl(p, botdiag, diag, updiag, un, info)
        call scopy (p, un, 1, uc4, 1)

        do 50 j=1,pm1
            up1(j)=uc1(j)
            up2(j)=uc2(j)
50      continue
        do 60 j=1,p
            up3(j)=uc3(j)
            up4(j)=uc4(j)
60      continue

70      continue

C Find solution vector
        call scopy(pm1, uc3, 1, un, 1)
        call scopy(pm1, uc4(2), 1, un(p+1), 1)
        un(p)=(uc3(p)+uc4(1))/2.

        it=it-1

        return
        end
```

```

C =====
C | f.f
C -----
C | Form function f depending on the test problem.
C | irun:1 f(x)=(pi^2+4*c)*sin(pi*x/2)/4
C | irun:2 f(x)=2+c*x*(1-x)
C | irun:3 f(x)=(4-c)*e^(2*x)/4
C =====
      subroutine f(n,irun,c,grdx,fc)
      integer n, irun
      integer i
      real c, ttgrdx, pi
      real grdx(1), fc(1)

      if (irun .EQ. 1) then
        pi=4.*atan(1.)
        do 10 i=1,n
          ttgrdx=pi*grdx(i)/2.
          pis=(pi*pi+4.*c)/4.
          fc(i)=pis*sin(ttgrdx)
10      continue
      elseif (irun .EQ. 2) then
        do 20 i=1,n
          ttgrdx=c*grdx(i)*(1.-grdx(i))
          fc(i)=2. + ttgrdx
20      continue
      else
        do 30 i=1,n
          ttgrdx=2.*grdx(i)
          fc(i)=(4.-c)*exp(ttgrdx)/4.
30      continue
      endif

      return
      end

```

```

C =====
C | Compute the Infinite Norm of the real solution and the computed
C | solution depending on the test problem.
C | irun:1 f(x)=(pi^2+4.*c)*sin(pi*x/2)/4, real solution sin(pi*x/2)
C | irun:2 f(x)=2+c*x*(1-x), real solution is x*(1-x)
C | irun:3 f(x)=(4-c)*e^(2*x)/4, real solution is -e^(2*x)/4
C =====

      real function findnorm(n,irun,c,sol,grdx)
      integer n
      real normres, c
      real sol(1), grdx(1)
      real pi

      normres=0.

      if (irun .EQ. 1) then
        pi=4.*atan(1.)
        do 10 i=1, n
          tmp = pi*grdx(i)/2.
          tmp = sin(tmp)
          tmp = sol(i) - tmp
          normres = amax1(normres, abs(tmp))
10      continue
      elseif (irun .EQ. 2) then
        do 20 i=1, n
          tmp = grdx(i)*(1.-grdx(i))
          tmp = sol(i) - tmp
          normres = amax1(normres, abs(tmp))
20      continue
      else
        do 30 i=1, n
          tmp = grdx(i)
          tmp = -exp(2*tmp)/4.
          tmp = sol(i) - tmp
          normres = amax1(normres, abs(tmp))
30      continue
      endif

```

```

    findnorm=normres

    return
end

```

A.2 Second Approach

The subroutines that do not appear here, are the same with the corresponding ones in the first approach.

```

C =====
C | iterate.f
C -----
C | Iterative Procedure as it comes from the Iterative schemes.
C | Only 2 full iterations are done.
C =====
      subroutine iterate(lda,p,al,bl,irun,grdx,lhs1,lhs3,lhs4,
&                      rhs1,rhs2,rhs3,rhs4,un,it,c)
      parameter (itmax=2)
      integer lda, p, it, irun
      integer j, info, pm1
      real al, bl, c, grdx(1)
      real lhs1(lda,1), lhs3(lda,1), lhs4(lda,1)
      real rhs1(1), rhs2(1), rhs3(1), rhs4(1), un(1)
      real up1(lda), up2(lda), up3(lda), up4(lda)
      real uc1(lda), uc2(lda), uc3(lda), uc4(lda)
      real diag(lda), updiag(lda), botdiag(lda)
      real pnorm

      pm1=p-1
      n=2*p-1

      call vfill(pm1, up1, 0.)
      call vfill(pm1, up2, 0.)
      call vfill(p, up3, 0.)
      call vfill(p, up4, 0.)

```

```

C up: the previous solution, uc: the current one
  do 70 it=1, itmax

C First Dirichlet Iteration
  call scopy (pm1, rhs1, 1, un, 1)
  un(pm1)= un(pm1) + bl*up3(p) + (1.-bl)*up4(1)
  do 10 j=1,pm1
    diag(j)=lhs1(j,j)
    updiag(j)=lhs1(j,j+1)
10  continue
  updiag(pm1)=0.
  call spts1(pm1, diag, updiag, un)
  call scopy (pm1, un, 1, uc1, 1)

C Second Dirichlet Iteration
  call scopy (pm1, rhs2, 1, un, 1)
  un(1)= un(1) + (1.-bl)*up3(p) + bl*up4(1)
  do 20 j=1,pm1
    diag(j)=lhs1(j,j)
    updiag(j)=lhs1(j,j+1)
20  continue
  updiag(pm1)=0.
  call spts1(pm1, diag, updiag, un)
  call scopy (pm1, un, 1, uc2, 1)

C First Neumann Iteration
  call scopy (p, rhs3, 1, un, 1)
  un(p)= un(p) + 3.*al*up3(p) - 3.*(1.-al)*up4(1)
&          + al*uc1(p-2) - 4.*al*uc1(pm1)
&          + 4.*(1.-al)*uc2(1) - (1.-al)*uc2(2)
  do 30 j=1,p
    botdiag(j)=lhs3(j,j-1)
    diag(j)=lhs3(j,j)
    updiag(j)=lhs3(j,j+1)
30  continue
  botdiag(1)=0.
  updiag(p)=0.

```

```

        call sgtsl(p, botdiag, diag, updiag, un, info)
        call scopy (p, un, 1, uc3, 1)

C Second Neumann Iteration
        call scopy (p, rhs4, 1, un, 1)
        un(1)= un(1) - 3.*(1.-al)*up3(p) + 3.*al*up4(1)
&          - (1.-al)*uc1(p-2) + 4.*(1.-al)*uc1(pm1)
&          - 4.*al*uc2(1) + al*uc2(2)
        do 40 j=1,p
            botdiag(j)=lhs4(j,j-1)
            diag(j)=lhs4(j,j)
            updiag(j)=lhs4(j,j+1)
40        continue
        botdiag(1)=0.
        updiag(p)=0.
        call sgtsl(p, botdiag, diag, updiag, un, info)
        call scopy (p, un, 1, uc4, 1)

        do 50 j=1,pm1
            up1(j)=uc1(j)
            up2(j)=uc2(j)
50        continue
        do 60 j=1,p
            up3(j)=uc3(j)
            up4(j)=uc4(j)
60        continue

70    continue

C Find solution vector
        call scopy(pm1, uc3, 1, un, 1)
        call scopy(pm1, uc4(2), 1, un(p+1), 1)
        un(p)=(uc3(p)+uc4(1))/2.

        it=it-1
        return
        end

```

```

C =====
C | lhsgen.f
C -----
C | Form the left hand side lhs:= 4 block tridiag(-1,2+c*h^2,-1)
C | The first two blocks are (p-1)x(p-1)
C | The next two are pxp having (2+c*h^2,-2) in the last row of the
C | third and first row of the fourth
C =====
      subroutine lhsgen(lda,p,al,bl,c,h,lhs1,lhs3,lhs4)
      integer lda, p
      integer j
      real al, bl, c, h
      real lhs1(lda,1), lhs3(lda,1), lhs4(lda,1)
      real diag

      diag=2. + c*h*h

      do 10 j=1,p
         call vfill(p,lhs1(1,j),0.)
         call vfill(p,lhs3(1,j),0.)
         call vfill(p,lhs4(1,j),0.)
10      continue

      lhs1(1,1)=diag
      lhs1(1,2)=-1.
      do 20 i=2,p-2
         lhs1(i,i-1)=-1.
         lhs1(i,i)=diag
         lhs1(i,i+1)=-1.
20      continue
      lhs1(p-1,p-2)=-1.
      lhs1(p-1,p-1)=diag

      lhs3(1,1)=diag
      lhs3(1,2)=-1.
      do 30 i=2,p-1
         lhs3(i,i-1)=-1.

```

```
        lhs3(i,i)=diag
        lhs3(i,i+1)=-1.
30      continue
        lhs3(p,p-1)=-2.
        lhs3(p,p)=diag

        lhs4(1,1)=diag
        lhs4(1,2)=-2.
        do 40 i=2,p-1
            lhs4(i,i-1)=-1.
            lhs4(i,i)=diag
            lhs4(i,i+1)=-1.
40      continue
        lhs4(p,p-1)=-1.
        lhs4(p,p)=diag

        return
        end
```

```
C =====
C | rhsgen.f
C -----
C | Form the right hand side standard vector rhs:= h^2 * f(x)
C | f(x) given by function f, depending on the test problem to run
C =====
      subroutine rhsgen(lda2,p,h,alpha,beta,irun,c,grdx,
&                      rhs1,rhs2,rhs3,rhs4)
      integer lda2, p, irun
      integer i, n
      real alpha, beta, h
      real hs
      real grdx(1), rhs1(1), rhs2(1), rhs3(1), rhs4(1)
      real fc(lda2)

      hs=h*h
      n=2*p-1

      call vfill(p,rhs1,0.)
      call vfill(p,rhs2,0.)
      call vfill(p,rhs3,0.)
      call vfill(p,rhs4,0.)

      call f(n,irun,c,grdx,fc)

C First Dirichlet Iteration
      do 10 i=1, p-1
          rhs1(i)=hs*fc(i)
10      continue
      rhs1(1)=rhs1(1) + alpha

C Second Dirichlet Iteration
      do 20 i=1, p-1
          rhs2(i)=hs*fc(i+p)
20      continue
      rhs2(p-1)=rhs2(p-1) + beta
```

```
C First Neumann Iteration
  do 30 i=1, p-1
    rhs3(i)=rhs1(i)
30  continue
    rhs3(p)=hs*fc(p)

C Second Neumann Iteration
  do 40 i=2,p
    rhs4(i)=rhs2(i-1)
40  continue
    rhs4(1)=rhs3(p)

  return
  end
```

References

- [1] A. BERMAN AND R.J. PLEMMONS, *Nonnegative Matrices in the Mathematical Sciences*, SIAM, Philadelphia, PA, 1994.
- [2] P. E. BJORSTAD AND O. WIDLUND, *To overlap or not overlap: A note on a domain decomposition method for elliptic problems*, SIAM J. Sci. Stat. Comput., 10 (1989), pp. 1053–1061.
- [3] T. F. CHAN, T. Y. HOU, AND P. L. LIONS, *Geometry related convergence results for domain decomposition algorithms*, SIAM J. Numer. Anal., 28 (1991), pp. 378–391.
- [4] T. F. CHAN AND D. C. RESASCO, *A survey of preconditioners for domain decomposition*, SIAM J. Numer. Anal., (1988), pp. ??–??
- [5] A. HADJIDIMOS, D. NOUTSOS, AND M. TZOUMAS, *Research Notes*, 1997.
- [6] D. KEYES AND W. GROPP, *A comparison of domain decomposition techniques for elliptic partial differential equations and their parallel implementation*, SIAM J. Sci. Stat. Comput., 8 (1987), pp. s166–s202.
- [7] S.-B. KIM, A. HADJIDIMOS, E.N. HOUSTIS, AND J.R. RICE, *Multi-parameterized Schwarz alternating methods for elliptic boundary value problems*, Math. Comput. Simulation, 42 (1996), pp. 47–76.
- [8] Y.-L. LAI, A. HADJIDIMOS, AND E.N. HOUSTIS, *A generalized Schwarz splitting method based on Hermite collocation for elliptic boundary value problems* Appl. Numer. Math., 21 (1996), pp. 265–290.
- [9] P. LE-TALLEC, Y. DE-ROECL, AND M. VIDRASCU, *Domain decomposition methods for large linearly elliptic 3-dimensional problems*, J. Comput. Appl. Math., 34 (1991), pp. 93–117.
- [10] P. L. LIONS, *On the Schwarz alternating method ———: A variant for nonoverlapping subdomains*, in Domain Decomposition Methods for Partial Differential Equations, R. Glowinski, G. Golub, G. Meurant, and J. Periaux, eds., SIAM, 1990, pp. 202–223.

-
- [11] J. RICE, P. TSOMPANOPOULOU, AND E. VAVALIS, *Review and performance interface relaxation methods for elliptic pdes*, Tech. Report CSD-TR-96-113 (in preparation), Purdue University, W. Lafayette. IN, 1996.
- [12] J. RICE, E. VAVALIS, AND D. YANG, *Convergence analysis of a non-overlapping domain decomposition method for elliptic pdes*, Tech. Report CSD-TR-93-048, Purdue University, W. Lafayette. IN, 1993.
- [13] C. SCHNEIDESCH, , AND M. DEVILLE, *Chebyshev collocation methods and multi-domain decomposition for navier-stokes equations in complex curved geometries*, Journal of Comput. Physics, 106 (1993), pp. 234–257.
- [14] B. SMITH, *A parallel implementation of an iterative substructuring algorithm for problems in three dimensions*, SIAM J. Sci. Stat. Comput., 14 (1993).
- [15] W.P. TANG, *Schwarz splitting and template operators*, Ph.D. Thesis, Dept. of Computer Sciences, Stanford University, (1987)
- [16] W.P. TANG, *Generalized Schwarz splittings*, SIAM J. Statist. Comput., 13 (1992), pp. 573-595
- [17] R. VARGA, *Matrix Iterative Analysis*, Prentice-Hall, Inc., New Jersey, 1962.
- [18] D. YOUNG, *Iterative Solution of Large Linear Systems*, Academic Press, New York, NY, 1971.