

Purdue University  
**Purdue e-Pubs**

---

Department of Computer Science Technical  
Reports

Department of Computer Science

---

1997

## Interface Relaxation Methods for Elliptic Differential Equations

John R. Rice  
*Purdue University*, [jrr@cs.purdue.edu](mailto:jrr@cs.purdue.edu)

P. Tsompanopoulou

E. Vavalis

Report Number:  
97-004

---

Rice, John R.; Tsompanopoulou, P.; and Vavalis, E., "Interface Relaxation Methods for Elliptic Differential Equations" (1997). *Department of Computer Science Technical Reports*. Paper 1344.  
<https://docs.lib.purdue.edu/cstech/1344>

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries.  
Please contact [epubs@purdue.edu](mailto:epubs@purdue.edu) for additional information.

**INTERFACE RELAXATION METHODS FOR  
ELLIPTIC DIFFERENTIAL EQUATIONS**

**John R. Rice  
E. Vavalis  
P. Tsompanopoulou**

**Department of Computer Sciences  
Purdue University  
West Lafayette, IN 47907**

**CSD-TR #97-004  
January 1997**

# Interface Relaxation Methods for Elliptic Differential Equations <sup>1</sup>

J.R. Rice, P. Tsompanopoulou<sup>2</sup> and E. Vavalis<sup>2</sup>

*Purdue University, Computer Science Department, West Lafayette, IN 47907*

---

## Abstract

A population of eight non-overlapping domain decomposition methods for solving elliptic differential equations are viewed and formulated as iterated interface relaxation procedures. A comprehensive review of the underlying mathematical ideas and the computational characteristics is given. The existing theoretical results are also reviewed and high level descriptions of the various algorithms are presented. The effectiveness of these methods on 1-dimensional differential problems is investigated by presenting and discussing preliminary performance evaluation data.

---

## 1 Introduction

The various domain decomposition methods that have been recently developed for the efficient solution of elliptic differential equations can be easily classified into two categories –overlapping and non–overlapping. Both approaches already have been used to effectively model large scale, industrial, ill-conditioned problems. Nevertheless it is believed that further theoretical and experimental analysis is required before such methods will become practical and useful tools for non-experts.

Overlapping (Schwartz) schemes have received in the past a great deal of attention. Articles that review and compare various such schemes [19] and survey the associated preconditioning strategies [7,4] have already appeared

---

<sup>1</sup> Work supported in part by PENED grants 95-602 and 95-107, NSF grants CCR-9202536 and CDA-9123502, and AFOSR FM 49620-92-J-0069. An early version of this paper was presented at the Ninth International Domain Decomposition Conference.

<sup>2</sup> Authors permanent address: University of Crete, Mathematics Department, 714 09 Heraklion, GREECE and IACM, FORTH, 711 10 Heraklion, GREECE.

in the literature. It is relatively recent that a number of studies have shown that non-overlapping schemes can compete well and can possibly free the user from certain complications in their formulation and implementation. The comparison of the main characteristics of these two classes of methods and the existence of equivalence relations between them have already received a great deal of study [3,2,5].

Interface relaxation methods are taking us a step beyond non-overlapping domain decomposition. In an effort to mimic the physics in the real world, they split a complicated partial differential equation (PDE) that acts on a large and/or complex domain into a set of PDE problems with different but simple, operators acting on different smaller and "easy" subdomains. This Multi-PDE, Multi-domain system is properly coupled using smoothing operators on the inter-domain boundaries. The present work reviews and evaluates a class of interface relaxation methods for solving elliptic PDE problems. Although these methods can be considered from the preconditioning viewpoint, here we follow Southwell's relaxation of the 1930's – but at the PDE level instead of the linear algebra level – to formulate them as iterated interface smoothing procedures. We believe that such a formalism has certain theoretical and algorithmic advantages. From the interface relaxation viewpoint these methods consist of partitioning the domain on a set of non-overlapping subdomains and of imposing some boundary conditions on the interface boundaries defined by this partition. Then, using initial guesses on the interfaces, the set of the resulting PDE problems is solved. The solutions obtained do not satisfy the interface boundary conditions and interface relaxation is applied to obtain new interface boundary values, which satisfy the conditions better, and we solve the PDEs with these new values. We repeat the above steps until convergence.

For our study we have collected most of the known interface relaxation methods and proposed three new ones. Specifically we consider the methods listed below in alphabetical order with respect to their acronyms. These acronyms are used in the sequel to refer to associated methods.

**AVE** A simple method of averaging the solution and its normal derivative along the interfaces.

**GEO** A method based on a simple geometric contraction.

**LSQ** A technique that uses overdetermined interface conditions and least squares.

**NEW** A scheme based on Newton's method to "correct" the interface values.

**ROB** An algorithm that uses Robin interface conditions for smoothing.

**SCO** A scheme that is based (but not formulated) on a Schur complement approach.

**SHO** A method based on the concept of the shooting method for solving Ordinary Differential Equations (ODEs).

**SPO** A method originated from the use of Steklov–Poincaré operator which involves alternating boundary condition types.

To the best of our knowledge **GEO LSQ** and **NEW** has not been considered in any previous studies. The analysis of these methods is beyond the scope of this paper. We should point out that in order to preserve some uniformity in our study we have not experimented with a class of interesting interdomain smoothing methods which use a few modes of the expansion [6] of certain interface operators (i.e., Lagrange multipliers [11,12] or Steklov–Poincaré operators [26,27]). We will only briefly describe these techniques.

The rest of the paper is organized as follows. In Section 2 we present the general framework for decomposing a multi–PDE problem into a collaborative pool of single–PDE problems and discuss the implications on simulating complicated physical problems. The interface relaxation methods we consider for this study are presented in Section 3, where we give their formulation and list the known theoretical results. In Section 4 we present our performance data and in Section 5 we summarize the contributions of our study.

## 2 Domain decomposition with iterated interface relaxation

Currently the domain decomposition world consists of two parts –overlapping and non–overlapping– both living in prosperity. Overlapping, known also as Schwartz, methods were the first considered and have already proved themselves as very efficient numerical procedures enjoying certain very desirable convergence properties. Nevertheless it has been also observed that they might have several serious drawbacks which will prohibit their use for certain applications. For example, almost all of the many proposed domain decomposition methods for solving wave propagation models (that consist of the Helmholtz equation coupled with various absorbing or reflecting boundary conditions) are non–overlapping and of interface relaxation type [1,9,20,29].

Non–overlapping methods exhibit certain advantages compared to overlapping ones. Specifically:

- They are not sensitive to jumps on the operator coefficients. Their convergence behavior and theoretical error estimates remain the same even if the differential operator includes discontinuous coefficients provided that the jumps occur along the interface lines.
- They have smaller communication overhead in a parallel implementation on distributed memory multiprocessor systems. Their communication overhead is propotional to the length of the interface lines while it is propotional to the overlapping area in the case of overlapping methods.

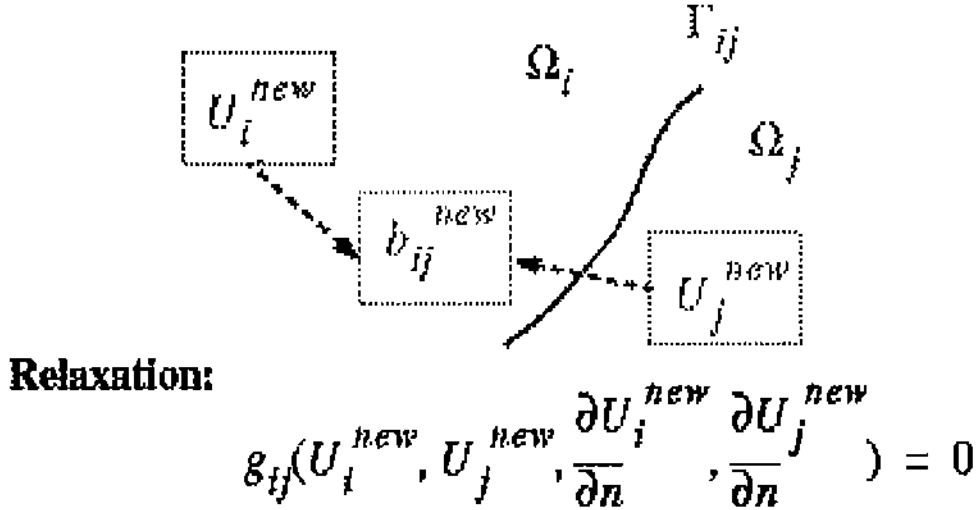


Fig. 1. The interface relaxation mechanism

- The bookkeeping is rather easy for the decomposition and manipulations of the associated data structures compared to the more complicated and costly bookkeeping of the overlapping methods.

There are two principal viewpoints of non-overlapping methods, preconditioning and interface relaxation. For an in depth and up-to-date survey of non-overlapping domain decomposition methods considered and analyzed from the preconditioning viewpoint the reader is referred to [32] and for a general formulation and analysis of interface relaxation methods to [24]. We give a brief presentation of the interface relaxation method philosophy and practice, in order to identify its main characteristics.

Interface relaxation is a step beyond non-overlapping domain decomposition; it follows Southwell's relaxation of the 1930's - but at the PDE instead of the linear algebra level - to formulate relaxation as iterated interface smoothing procedures. A complex physical phenomenon consists of a collection of simple parts with each one of them obeying a single physical law locally and adjusting its interface conditions with neighbors. Interface relaxation partitions the domain on a set of non-overlapping subdomains, imposes some boundary conditions on the interface among subdomains lines. Given an initial guess, it imitates the physics of the real world by solving the local problems exactly on each subdomain and relaxing boundary values to get better estimates of correct interface conditions. This is illustrated in Figure 1 where the generic relaxation formula  $g_{i,j}$  (based on the current solutions  $U_i^{New}$  and  $U_j^{New}$  of the two local to the neighboring subdomains  $\Omega_i$  and  $\Omega_j$ ) calculates successive approximations  $b_{i,j}^{New}$  to the solution on the interface  $\Gamma_{i,j}$  between them.

To formally describe this method we consider the differential problem

$$Du = f \text{ in } \Omega, \quad Bu = c \text{ on } \partial\Omega \quad (1)$$

where  $D$  is an elliptic, non-linear in general, differential operator and  $B$  a condition operator defined on the boundary  $\partial\Omega$  of a domain  $\Omega \in R^d, d = 1, 2, \dots$ . This domain is partitioned into  $p$  subdomains  $\Omega_i, i = 1, \dots, p$  such that  $\Omega = \cup_{i=1}^p \Omega_i$  and  $\cap_{i=1}^p \overset{\circ}{\Omega}_i = \emptyset$ . For reasons related either to the physical characteristics of this problem or to the computing resources available, one would like to replace (1) with the following system of loosely coupled differential problems

$$D_i u = f_i \text{ in } \Omega_i, \quad G_i u = 0 \text{ on } \partial\Omega_i \setminus \partial\Omega, \quad B_i u = c_i \text{ on } \partial\Omega_i \cap \partial\Omega \quad (2)$$

where  $i = 1, \dots, p$ . These differential problems are coupled through the interface conditions  $G_i u = 0$  and involve the restrictions  $D_i$  and  $B_i$  of the global differential and boundary operators,  $D$  and  $B$ , respectively, on each subdomain with some of them possibly linear and some others nonlinear. The functions  $f_i$  and  $c_i$  are similar restrictions of functions  $f$  and  $c$ . The local interface operator  $G_i$  is associated with the interface relaxation method and different selections for the  $G_i$ 's lead to different relaxation schemes. In this study we consider several interface relaxation methods that have the following characteristics:

- They first decompose the problem (1) at differential level and then discretize the resulting differential subproblems (2).
- They have the versatility to use the most appropriate discretization scheme for each subproblem.
- They do not overlap the subdomains  $\Omega_i$ .
- Using good relaxation parameters in  $G_i$ , they are fast enough so no preconditioning is needed.
- They simplify the geometry and physics of the computation by considering the subproblems (2) instead of the global differential problem (1).
- They can utilize software parts technology by reusing existing "legacy" software parts for solving the individual subproblems (2).
- They are general and robust.

There are several challenging questions concerning practical applications of such methods (e.g. find the most suitable relaxer for a particular problem of application, determine what is the domain of applicability of each one of them, explain the interaction between the mathematical iteration and the numerical solving method, select "good" or "optimal" values for the relaxation parameters involved,  $\dots$ ). It is worth to point out that since we decompose and formulate all methods at continuum level the convergence analysis of these methods need to be carried out at PDE (continuum) level and therefore is a

mathematical and not a numerical analysis problem (see [24] for a discussion).

### 3 Interface relaxation methods

Due to the inherent abstraction, it is relatively easy to describe the various interface smoothing methods at both the conceptual and algorithmic level. Next we present the eight methods, give their high level algorithmic description and briefly present the known theoretical results. Detailed algorithms to define all schemes are given in the Appendix. For simplicity in the presentation of algorithms, we consider only one-way (along the x-axis) partition of the domain. Therefore each subdomain has two interface lines with the two neighboring subdomains. The basic building block for our algorithms is the procedure `u = solve_pde(ui, dui)` which calculates the solution  $u$  of the local to a subdomain PDE problem with Dirichlet, Neumann or Robin boundary conditions on the interface using as the interface values  $u_i$  and its gradient  $du_i$ . The subscripts  $R$  and  $L$  denote left and right subdomains or interfaces respectively and  $u_i$  denotes the solution of the problem associated with subdomain  $\Omega_i$ .

#### *The Dirichlet/Neumann Averaging (AVE) Method*

We start by presenting one of the simplest schemes which consists of two PDE solving sweeps coupled with two smoothing interface relaxation steps. In the first sweep, the Dirichlet problem is solved on all subdomains. Then the relaxation procedure smoothes the derivatives along all interfaces by estimating the normal derivative as a convex combination of the previously computed normal derivatives of the two adjacent subdomains. These estimates are then used as boundary conditions in the second PDE solving sweep where the Neumann problem is solved on all subdomains. The second relaxation step follows and computes estimates of the unknown function on the interfaces taking a convex combination of the previously computed solutions on the adjacent subdomains. These estimates are to be passed to the next iteration's Dirichlet sweep. This method, which we classify as a *two-step* method, can be algorithmically described by

- 
- for  $k = 0, 1, 2, \dots$ 
    - $u^{(k+\frac{1}{2})} = \text{solve\_pde}(u_i)$  in each subdomain
    - $du_i = \alpha \frac{\partial u_R^{(k+\frac{1}{2})}}{\partial x} + (1 - \alpha) \frac{\partial u_L^{(k+\frac{1}{2})}}{\partial x}$  on each interface
    - $u^{(k+1)} = \text{solve\_pde}(du_i)$  in each subdomain
    - $u_i = \beta u_R^{(k+1)} + (1 - \beta) u_L^{(k+1)}$  on each interface.



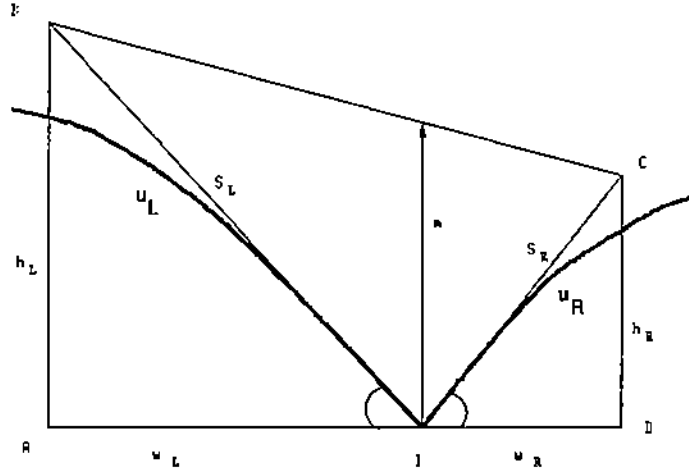


Fig. 2. Cross section perpendicular to the interface where  $u_L$  and  $u_R$  have slopes  $S_L$  and  $S_R$  at the interface point  $I$ . Changing the values of  $u_L$  and  $u_R$  by  $m$  makes these slopes equal.

where  $\alpha, \beta \in (0, 1)$  are relaxation parameters. There have been a few theoretical studies on the convergence of the above scheme which are discussed in [25]. In particular in [33] a convergence analysis of the method is carried out at a differential level using Hilbert space techniques. In [34] the Galerkin finite element method and the hybrid mixed finite element method are employed to give discrete versions of this method. Fourier analysis is used in [30] to obtain sharp convergence results and to estimate optimum values for the relaxation parameters involved for simple model problems.

### *The Geometric (GEO) Contraction Based Method*

**GEO** estimates the new solution for each subdomain by solving a Dirichlet problem and is classified as an *one-step* method. The values on the interfaces are obtained by adding to the old ones, a geometrically weighted combination of the normal boundary derivatives of the adjacent subdomains. Specifically, we assume in Figure 2 that  $u_L$  and  $u_R$  are the solutions of the PDE problems associated with the left and right subdomains, respectively, of the interface point  $I$ . They are equal along  $I$  and we denote by  $S_L$  and  $S_R$  their slopes at  $I$ . As it can be easily seen geometrically,  $m$  is the correction needed to be added to  $u_L$  and  $u_R$  so as to match the normal derivatives at  $I$ . To calculate  $m$  we consider the two right triangles  $IAB$  and  $CDI$  whose heights are given multiplying the corresponding tangent with the base of the triangle, or equiv-

alently multiplying the normal derivative with the base. The bases  $w_L$  and  $w_R$  are the widths assumed for the validity of the slope values; these can be arbitrarily selected and play the role of the relaxation parameters. The new interface values are now given by adding the weighted average of the heights to the old interface values  $u_L$  and  $u_R$ . In case that  $u_R \neq u_L$  on  $I$  we simply use their average. GEO is given algorithmically by

- 
- for  $k = 0, 1, 2, \dots$ 
    - $ui = \frac{u_L^{(k)} + u_R^{(k)}}{2} + \frac{w_L w_R}{w_L + w_R} \left( \frac{\partial u_L^{(k)}}{\partial x} + \frac{\partial u_R^{(k)}}{\partial x} \right)$  on each interface
    - $u^{(k+1)} = \text{solve\_pde}(ui)$  in each subdomain
- 

To the best of our knowledge this method has not been considered in any previous studies.

### *The Least Squares (LSQ) Method*

The next method has also not appeared in the literature before and does not require the determination of any relaxation parameters. The interface smoothing is achieved through least squares approximation by setting the value of the solution on a specific interface segment to be the one that best approximates the various interface conditions in the least squares sense. Thus the Dirichlet problem is solved first on all sub-domains and then a new value of the solution on the interfaces is obtained by solving the least squares problems on all interface segments. This value is used in the next iteration's Dirichlet step. This method is particularly useful in the case where more than the usual two interface conditions (for example any combination of complicated stress continuity, radiation or equilibrium conditions) are to be satisfied.

- 
- for  $k = 0, 1, 2, \dots$ 
    - Calculate  $ui_R^{(k+1)}, ui_L^{(k+1)}$  as the best least squares fit to the over-determined system  $A_L ui_L^{(k)} + A_R ui_R^{(k)} = 0$ , where  $A_R, A_L \in R^{n \times 2}$  on each interface
    - $u^{(k+1)} = \text{solve\_pde}(ui_R^{(k+1)}, ui_L^{(k+1)})$  in each subdomain
-

### *The Newton's (NEW) Method*

Another new, to our believe, idea is to use discrete Newton's method to update the values at the interface according to the following algorithm.

- 
- for  $k = 0, 1, 2, \dots$ 
    - Solve (for  $\delta_L$  and  $\delta_R$ ) the interface system:
$$\left(u_L^{(k)} + \delta_L\right) - \left(u_R^{(k)} + \delta_R\right) = 0$$
$$\frac{\partial u_L^{(k)}}{\partial x} \left(u_L^{(k)} + \delta_L\right) - \frac{\partial u_R^{(k)}}{\partial x} \left(u_R^{(k)} + \delta_R\right) = 0$$
    - $u_i^{(k+1)} = u_i^{(k)} + \delta$  on each interface
    - $u^{(k+1)} = \text{solve\_pde}(u_i^{(k+1)})$  in each subdomain
- 

There is no general convergence analysis for this new single step scheme which does not involve any relaxation parameters. Like most applications of Newton's method, it should converge very rapidly in some neighborhood of the true solution.

### *The Robin Relaxation (ROB) Method*

An even simpler interface relaxation is the one based on Robin interface conditions to transmit information across subdomain boundaries. It was first proposed in [23] and analyzed later in [10,18]. One solves the local PDE on the subdomains using Robin conditions on the interface lines by matching a convex combination of Dirichlet and Neumann data from the neighboring subdomains.

- 
- for  $k = 0, 1, 2, \dots$ 
    - On each sub-domain solve:
$$Lu^{(k+1)} = f \in \Omega \text{ with}$$
$$\frac{\partial u^{(k+1)}}{\partial x} + \rho u^{(k+1)} = \frac{\partial u_L^{(k)}}{\partial x} + \rho u_L^{(k)} \text{ on subdomain's left interface.}$$
$$\frac{\partial u^{(k+1)}}{\partial x} + \rho u^{(k+1)} = \frac{\partial u_R^{(k)}}{\partial x} + \rho u_R^{(k)} \text{ on subdomain's right interface.}$$
- 

Here  $\rho$  is a relaxation parameter. The convergence of this method was analyzed in [23] at the differential level assuming arbitrary decompositions and using "energy" estimates. The determination of effective choices for  $\lambda$  was marked as

“by large an open problem”. Variations of the above described method have appeared in the literature lately. Specifically in [17] an ADI-based modification for accelerating the convergence of the **ROB** scheme is proposed and analyzed. A modification of **ROB** that extends its applicability and frees it from the cross-point trouble is formulated and analyzed in [8]. Another variation that uses the tangential derivatives in addition to the normal derivative for smoothing is given in [31] where optimal values for the relaxation parameters are obtained for a model problem.

### *The Schur complement (SCO) Method*

Among the first interface relaxation procedures that captured the attention of researchers is the one analyzed in [14] (see also the references therein). It alternates Dirichlet and Neumann interface conditions in space and can be described by

- 
- for  $k = 0, 1, 2, \dots$
  - $u_{1L} = u, u_{1R} = \theta_1 u_2^{(k)} + (1 - \theta_1) u_1^{(k)}$
  - $u_1^{(k+1)} = \text{solve\_pde}(u_{1L}, u_{1R})$
  - for  $i = 2, \dots, p-1$ 
    - $du_{iL} = \frac{\partial u_{i-1}^{(k+1)}}{\partial x}$
    - $u_{iR} = \theta_i u_{i+1}^{(k)} + (1 - \theta_i) u_i^{(k)}$
    - $u_i^{(k+1)} = \text{solve\_pde}(u_{iR}, du_{iL})$
  - $u_{pR} = u, du_{pL} = \frac{\partial u_{p-1}^{(k+1)}}{\partial x}$
  - $u_p^{(k+1)} = \text{solve\_pde}(u_{pR}, du_{pL})$
- 

Here  $\theta \in (0, 1)$  is a relaxation parameter. The convergence analysis at the differential level for the case of Helmholtz equation in two variables and 1-dimensional decompositions at differential level is given in [14] together with expressions that lead to optimum values for  $\theta$ . A method for dynamically determine, at each iteration, values for  $\theta$  for the spectral collocation approximation of the differential problems is also given. To the best of our knowledge, **SCO** is the only interface relaxation technique that has so far been successfully extended and applied to fourth order elliptic problems [15].

### *The Shooting (SHO) Method*

This method is proposed in [21] where it is formulated primarily for 1-dimensional boundary value problems. A convergence analysis was carried out and optimum values for the relaxation parameters were obtained for model problems. The basic idea is to couple the problems on the subdomains by solving the defect equation  $D(ui) \equiv \frac{\partial u_L^{(k)}}{\partial x} - \frac{\partial u_R^{(k)}}{\partial x} = 0$  on the interfaces using a fixed point (Picard) iteration scheme to obtain new values.

- 
- for  $k = 0, 1, 2, \dots$ 
    - $\alpha^{(k+1)} = \frac{\alpha^{(k)} |D(ui^{(k+1)})|}{|D(ui^{(k)}) - D(ui^{(k+1)})|}$  on each interface
    - $u_i^{(k+1)} = u_i^{(k)} - \alpha^{(k+1)} D(u_i^{(k)})$  on each interface
    - $u^{(k+1)} = \text{solve\_pde}(u_i^{(k+1)})$  in each subdomain
- 

### *The Steklov-Poincaré operator (SPO) Method*

This method was first mentioned in [22] but analyzed from the preconditioning viewpoint only. It uses the Steklov-Poincaré operator to carry the procedure of smoothing the normal derivatives at the interfaces, it is a *two-step* method described by the following algorithm

- 
- for  $k = 0, 1, 2, \dots$ 
    - $u^{(k+\frac{1}{2})} = \text{solve\_pde}(ui)$  in each subdomain
    - $dui = \frac{1}{2} \left( \frac{\partial u_R^{(k+\frac{1}{2})}}{\partial x} + \frac{\partial u_L^{(k+\frac{1}{2})}}{\partial x} \right)$  on each interface
    - $u^{(k+1)} = \text{solve\_pde0}(dui)$  ( $Lu = 0$ ) in each subdomain
    - $ui = ui - \frac{\rho}{2} (u_R^{(k+1)} + u_L^{(k+1)})$  on each interface
- 

No theoretical results, from the interface relaxation viewpoint, are available for SPO.

### 3.1 Methods not considered

As mentioned in the introduction, powerful interface relaxation methods can be constructed using spectral expansions of a trace operator. In this approach operators like Lagrange multipliers or Steklov–Poincaré operator, which can be interpreted as the interface flux, are solved to determine an improved value of the unknown function on the interface. Specifically, in [26] and [27] an independent low dimensional set of interfacial basis functions are used to meet interdomain continuity requirements on the solution. These functions are derived locally in each subdomain by solving an eigenvalue problem of the Steklov–Poincaré operator on the complementary region. An idea similar to the above approach is used in [11] and [12] where a different set of basis functions is used to smooth across interfaces. It is shown that a relatively small basis set for the Lagrange multiplier has certain significant advantages. In particular trigonometric functions, orthogonal polynomials, and one-dimensional Lagrange finite elements have been suggested as approximating basis set on the interface.

## 4 Numerical Experiments

An extensive and systematic performance evaluation study of all the interface relaxation schemes presented above is under way for general 2-dimensional decompositions using the SciAgents [13] system. In this section we present and discuss numerical performance data for 1-dimensional problems. These problems might be too simple to be of practical importance, but these experiments can be very illuminating for understanding the nature of the interface relaxation method. They might be useful to show the physical meaning and importance of the various characteristics and parameters involved in the relaxers. We should add that we have implemented most of the relaxation schemes presented above for 2-dimensional problems using Ellpack [28] assuming "skyline" domains (a string of rectangles of different heights and widths) This leads to 1-dimensional decompositions and we performed some selective experiments, all of these were in good agreement with both the quantitative and qualitative conclusions we draw from the 1-dimensional experiments presented next.

We consider the differential equation  $u'' - \gamma u = f$  in  $[0, 1]$ , where  $f$  is selected such that  $u(x) = e^{x+4}x(x-1)(x-.7)$  and we assume Dirichlet boundary conditions. All interface relaxation schemes are implemented in a unified way using MATLAB on a SUN workstation. The MATLAB code for the algorithms given in the Appendix can be obtained from our web page<sup>3</sup>. Central differ-

---

<sup>3</sup>[http://www.cs.purdue.edu/homes/mav/projects/m\\_code.html](http://www.cs.purdue.edu/homes/mav/projects/m_code.html)

ences are used to discretize the differential equation. We have verified that the convergence rate of all methods is independent of the local grid size and we use 160 equally distributed grid points to discretize the domain  $\Omega \equiv [0, 1]$ . The interval  $[0, 1]$  is partitioned into subdomains with interface conditions taken to be continuous value and derivative. Unless otherwise stated, we start all iterations from a zero initial guess and we select the values for the various parameters involved in the relaxation schemes in a straight forward and naive way. In particular we set  $\alpha = \beta = 1/2$  in **AVE**,  $w_L = w_R =$  half the length of the associated subdomain in **GEO**,  $\rho = 1$  in **ROB** and **SPO**, and  $\theta = 1/2$  in **SCO**. We also set  $\gamma = 20$  for all data except Figure 6. We select this, not very common, value of  $\gamma$  in order to increase the experimental data that can be fitted into the plots (in particular in Figures 3 and 6) so a clear qualitative comparison picture can be easily drawn.

We start with Figure 3 where the convergence rate of all relaxers is presented for 2, 4, 5 and 8 subdomains. We plot the logarithm of the max-norm of the error (on the  $y$ -axis) of the computed solution at the first 20 iterations (first 200 for **LSQ**) versus the iteration number. It is immediately seen that **LSQ** is by far the slowest. For 8 subdomains **SPO** is the fastest and **AVE** the second slowest (not seen in Figure 3). Nevertheless **AVE** is the fastest for 2 and 4 subdomains. **NEW** and **SHO** behave in a similar and rather erratic way. The number of subdomains does not seem to affect the rate of convergence of **LSQ**, **ROB** and **SPO** as much as the rest of the relaxers.

We believe that the specific convergence pattern might give important information about the convergence characteristics. To explore this, Figure 4 shows, for all relaxers and for a uniform decomposition of  $\Omega$  into 4 subdomains, the exact solution and the the computed solutions associated with the first three (first, tenth and twentieth for **LSQ**) iterations. Half of the schemes (**GEO**, **LSQ**, **SHO** and **SPO**) approach the exact solution in a monotonic (or nearly so) and smooth way while the rest do not seem to exhibit a specific pattern.

We next examine the convergence history of the *two step schemes* in more detail. The plots associated with the two-step methods (**AVE** and **SPO**) in Figure 4 correspond to their Dirichlet sweeps. In Figure 5 we present, in the same way, the history for their Neumann steps as well. We therefore see that some methods (**GEO**, **LSQ**, **SPO**) converge in a monotonic and systematic way. This suggests that their convergence could be accelerated by some extrapolation procedure. Other methods exhibit oscillatory convergence so averaging might improve the convergence. Still others show no obvious patterns of convergence.

It is obviously expected that the convergence of the interface relaxation depends on the differential operator. To obtain a preliminary idea about this dependence we systematically vary the coefficient  $\gamma (= 1, 10, 20, 30)$  in the op-

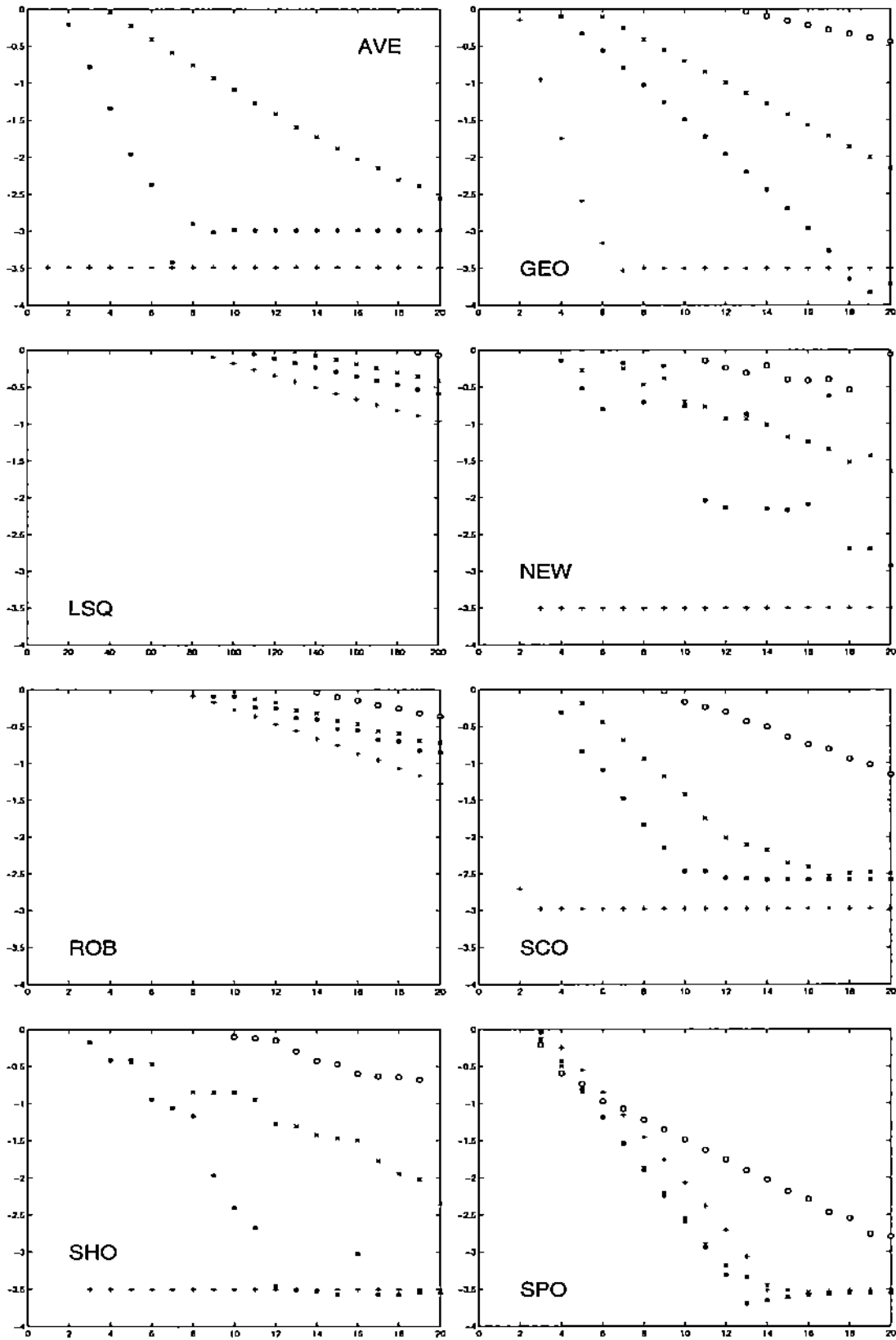


Fig. 3. Convergence rate for 2 (+), 4 (\*), 5 (x) and 8 (o) subdomains. On the  $x$ -axis we have the iteration number and on the  $y$ -axis the logarithm of the max-norm of the error.



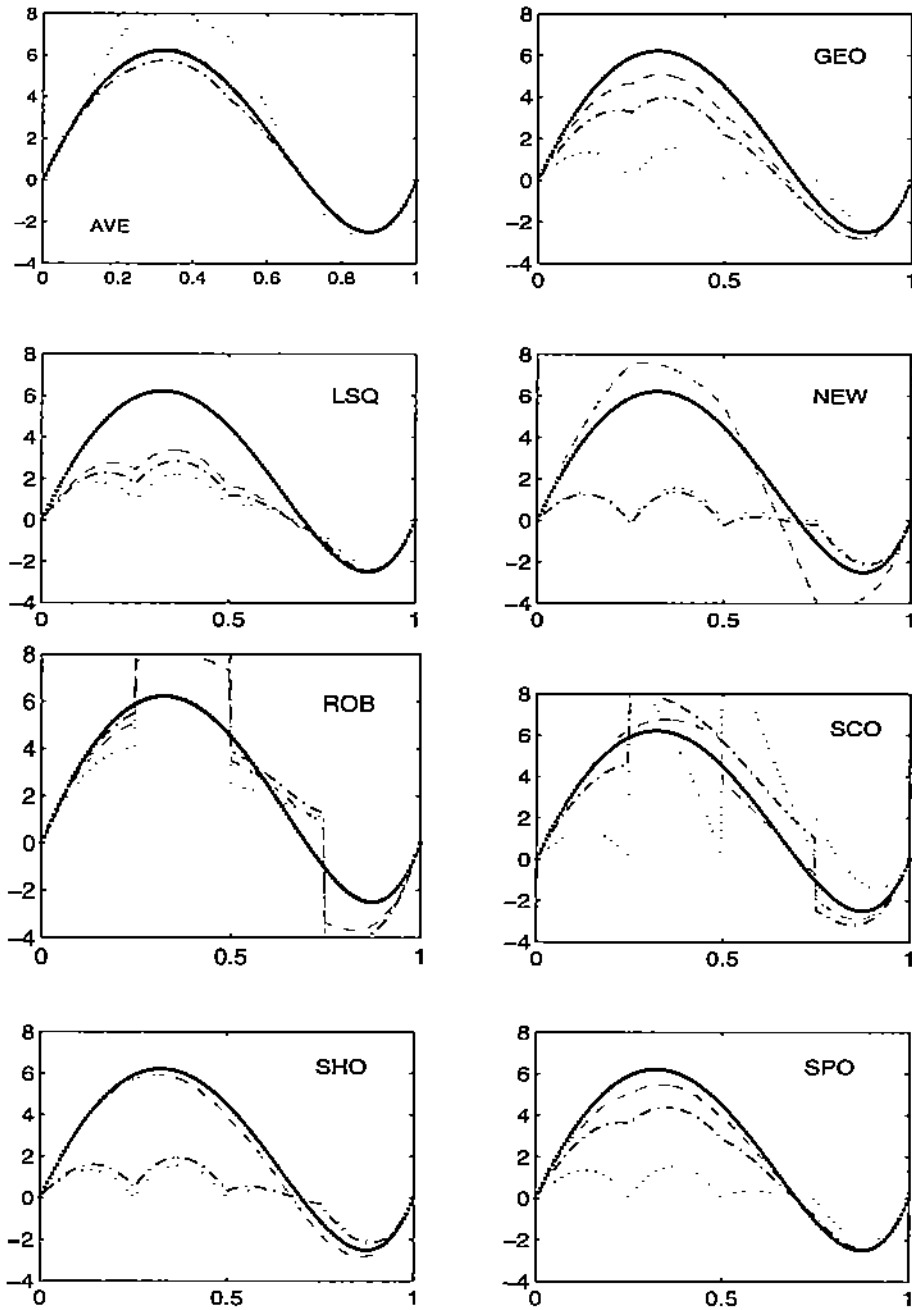


Fig. 4. Convergence history of the eight relaxers in a 4 subdomain decomposition. The true solution (solid line) is plotted along with the first (dotted line), second (dot-dashed line), and third (dashed line) computed solutions. For LSQ 1, 10 and 20 are shown instead of 1, 2 and 3.

erator and measure the convergence rate assuming a 4 subdomain uniform partition of  $\Omega$ . Our data are presented in Figure 6 which uses the same axes as in Figure 3. As  $\gamma$  becomes larger there is a general trend for the convergence rate to become faster (AVE, GEO, LSQ, NEW, SCO, SHO) or to

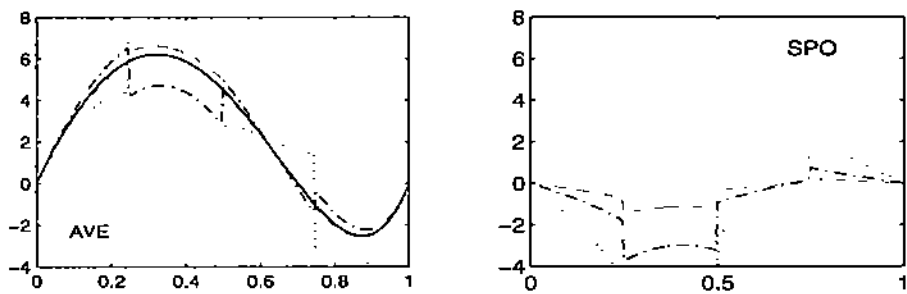


Fig. 5. Convergence history of the two-step relaxers, **AVE** and **SPO**, during the Neumann sweep.

be nearly unchanged (**ROB**, **SPO**). Increasing  $\gamma$  makes the operator more singular (i.e., divide through by  $\gamma$ ) but our choice of  $f$  to have  $u$  fixed ameliorates this effect and the problem actually becomes easier to solve (i.e., for  $\gamma = \infty$  the solution  $u$  is  $e^{x+4}x(x-1)(x-.7)$ ).

Recall that all the data presented so far correspond to uniform partitions of the interval  $[0, 1]$ . We next test the effect of non-uniform partitions by moving the interface point (denoted by  $ip$ ) from .2 to .4, .5 and .8. In Figure 7, where we consider the four different 2 subdomain partitions, we clearly see that only **LSQ** is sensitive to this change. Again the axes are as in Figure 3.

We note that three (**LSQ**, **NEW**, **SHO**) of the eight methods are parameter-free. The rest involve parameters of various kind whose values were selected in a naive and straight forward way for the experiments described above. In Figure 8 we systematically vary the values of these parameters and present convergence plots for the 2 subdomain case with the interface point at .8. In those two methods (**AVE**, **GEO**) with two parameters, their values are made equal in this experiment. We see that the parameter choices have a strong affect on the convergence behavior. The best parameter choices for Figure 8 are: **AVE** ( $\alpha = \beta = 0.3$ ), **GEO** ( $w_L = w_R = 0.7$ ), **ROB** ( $\rho = 0.9$ ), **SCO** ( $\theta = 0.5$ ) and **SPO** ( $\rho = 0.9$ ). These data show clearly that there is an important open question for these methods: **How does one choose optimal (or good) parameter values?**

It is worth to point out here that although the **LSQ** methods seems to be by far the slowest it enjoys certain very desirable properties (besides its capability of dealing with complicated interface conditions). Specifically its rate of convergence depends very little on the number of subdomains and the value of the coefficient  $\gamma$ . Furthermore it is parameter free and converges in a smooth, systematic and monotone way. It can possible be accelerated significantly using a parameter but this is beyond the scope of our study.

Among all the relaxation methods considered in this study only **SCO** appears

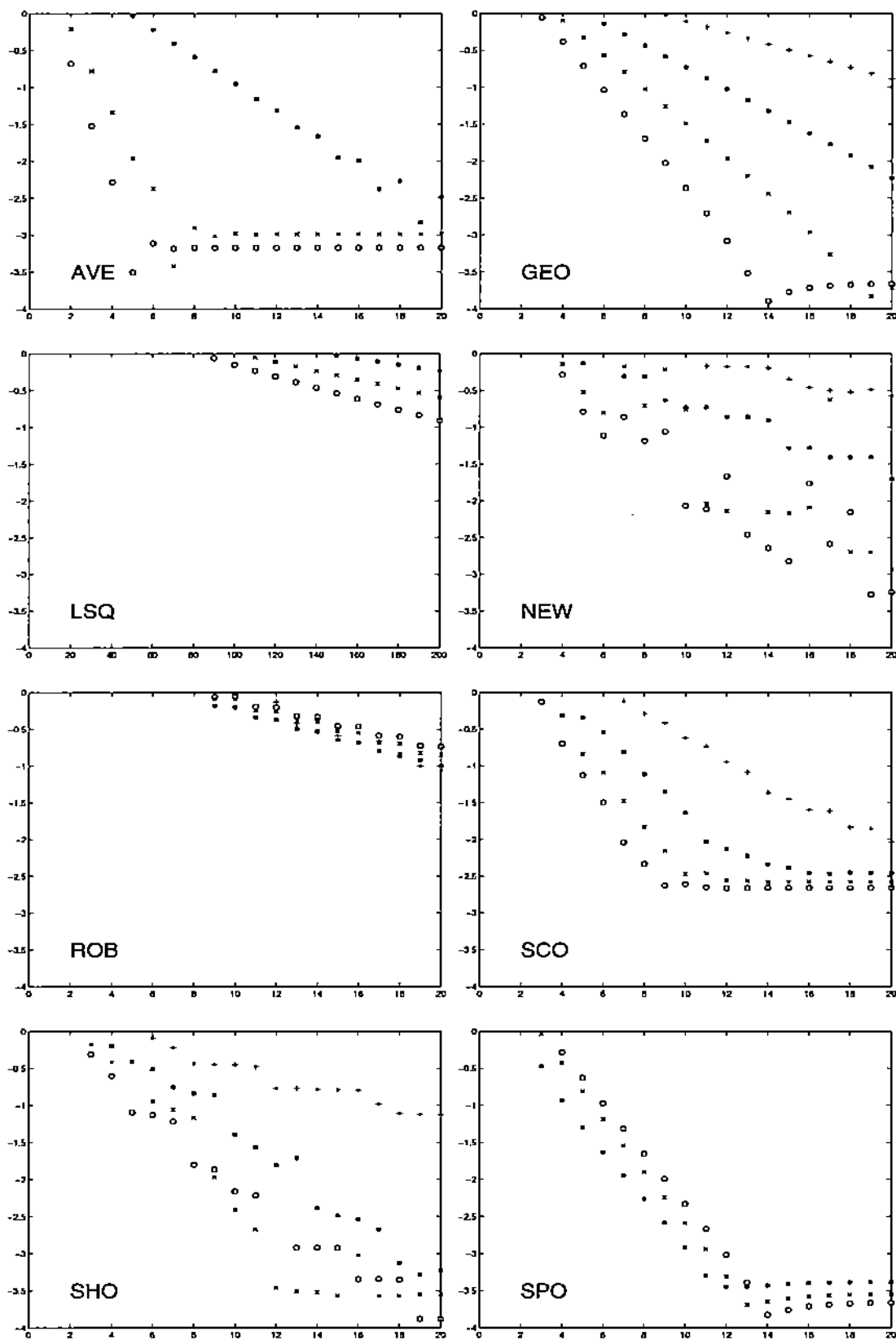


Fig. 6. The effect of the coefficient  $\gamma$  on the convergence rate of the relaxation schemes. The max-norm of the error is plotted for  $\gamma = 1$  (+), 10 (\*), 20 (x) and 30 (o) with a four subdomain partition of  $[0, 1]$ .

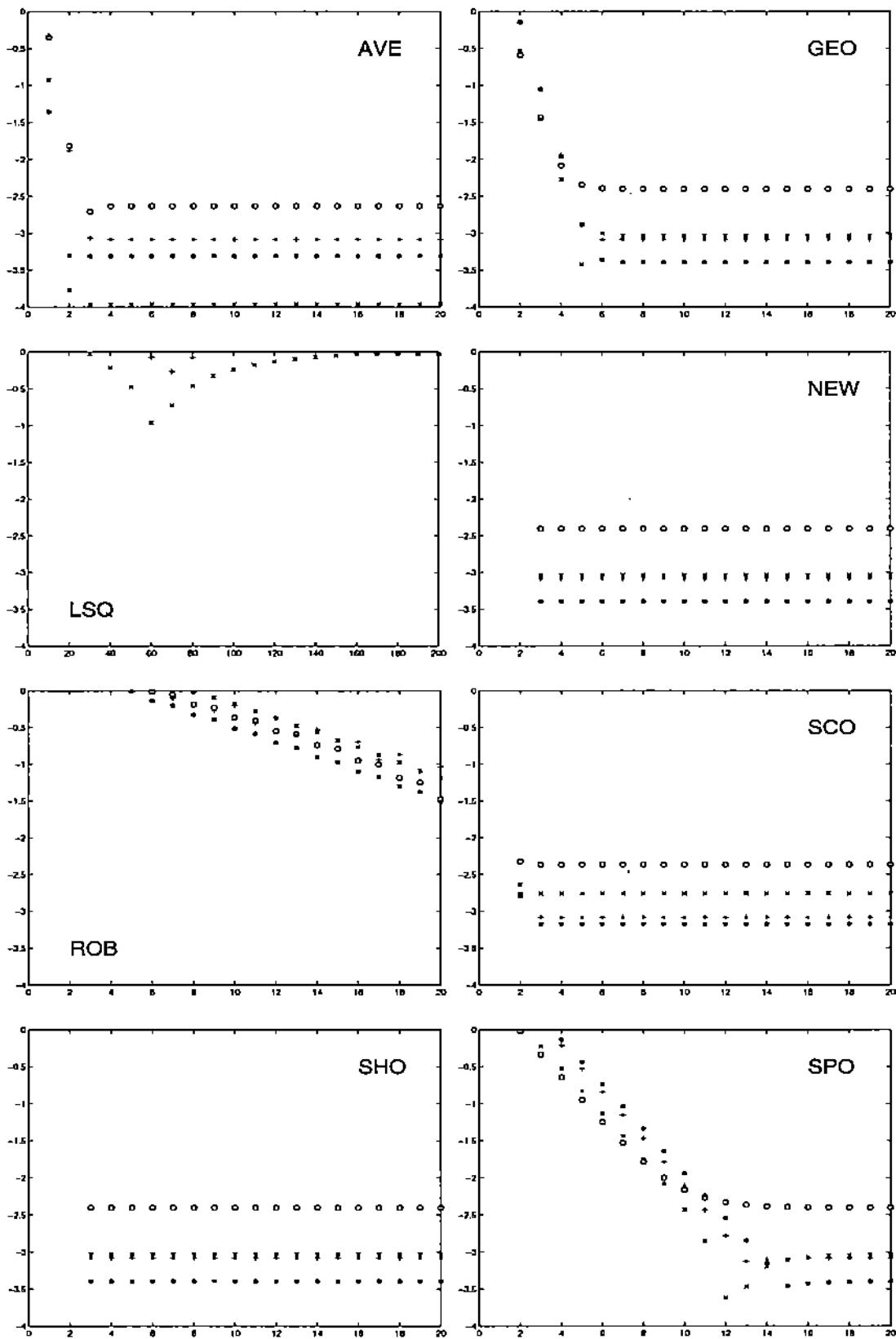


Fig. 7. The convergence of the relaxation schemes for non-uniform 2 subdomain decompositions. The interface point  $ip$  is placed at 0.2 (+), 0.4 (\*), 0.6 (x) and 0.8 (o).

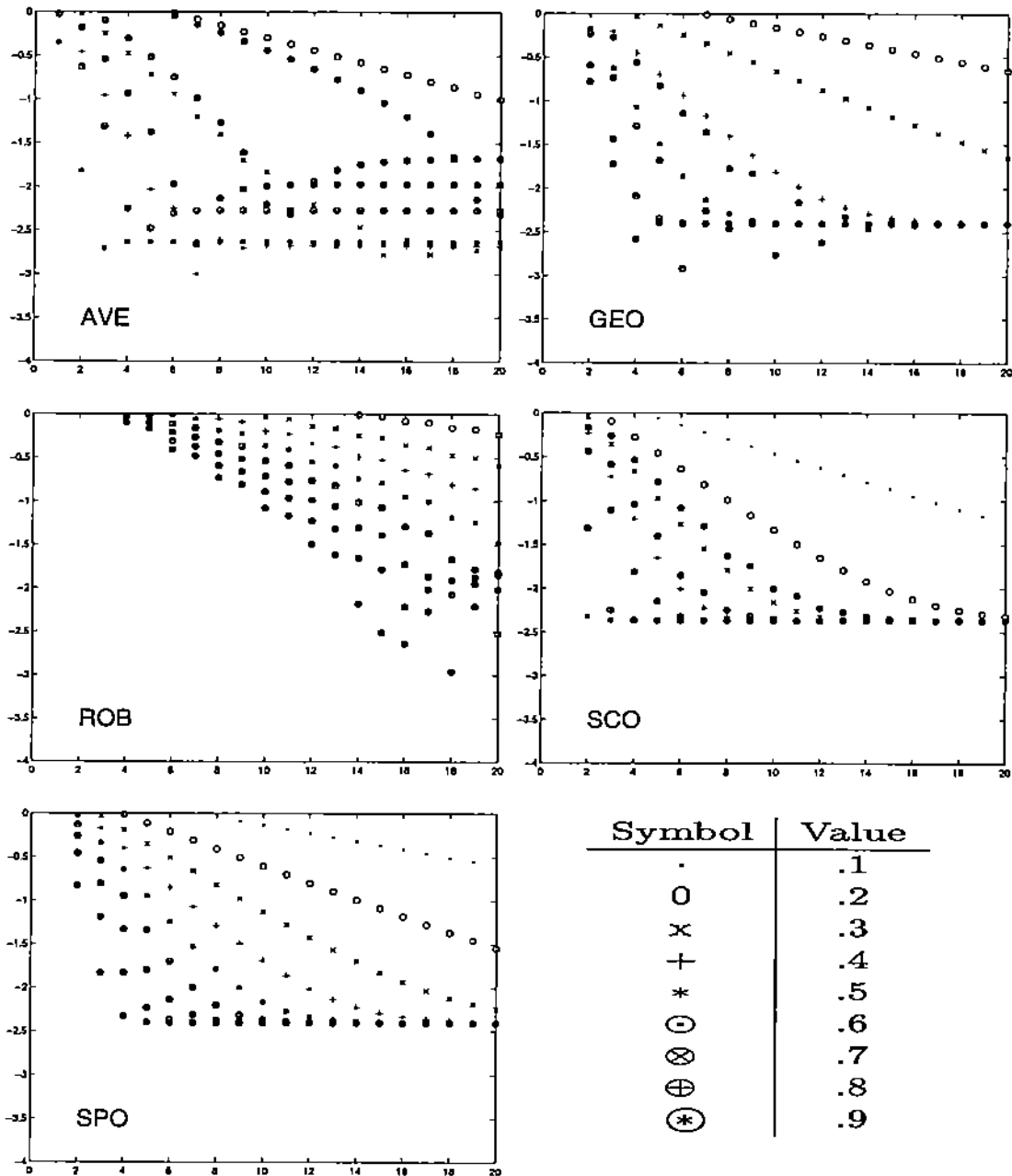


Fig. 8. The effect of the parameter selection on the convergence behavior of five relaxation methods. The legends and parameter values used are given in the lower right, the two parameters of **AVE** and **GEO** are both set equal to the value shown.

to be sensitive on the order the various subdomains are processed during the interface relaxation process. We experimented for **SCO** as follows: Select a subdomain  $q$  as the first for an iteration and then process the others in sequence (left to right, wrapping around at the right end of the interval). In Table 1 we present the number of iterations required by this scheme to reduce the norm of the difference of two successive iterants below  $10^{-5}$  (i.e.  $\|u^{(k+1)} - u^{(k)}\|_{\infty} < 10^{-5}$ ) as a function of the starting subdomain  $q$ . Specifically, for

Starting Subdomain	1	2	3	4	5	6	7	8
Number of Iterations	48	40	32	24	25	34	40	42

Table 1

Number of iterations  $k$  to achieve  $\|u^{(k+1)} - u^{(k)}\|_\infty < 10^{-5}$  for various starting subdomains in the SCO method.

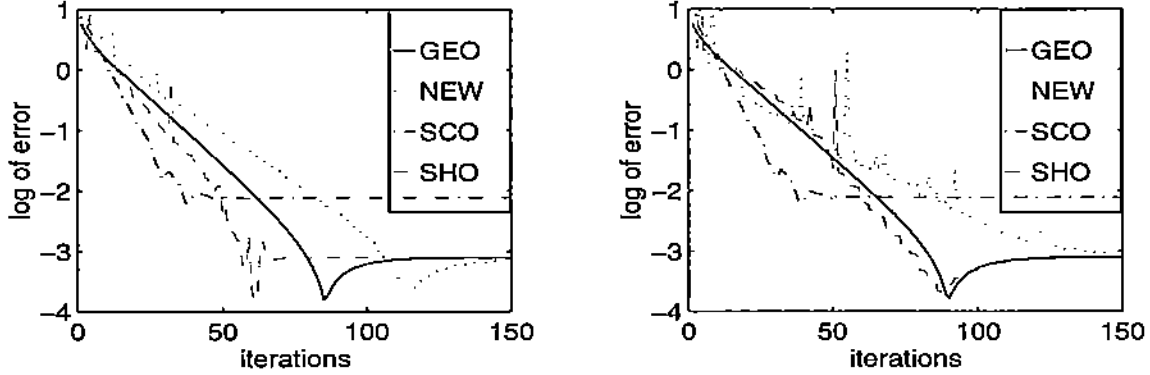


Fig. 9. The history of convergence of four relaxation methods for  $u'' + \sin(2\pi x)u = 0$  (on the left) and  $u'' + \cos(2\pi x)u = 0$  (on the right)

Table 1 we use a uniform decomposition of 8 subdomains and we start the domain decomposition scheme from subdomain  $q = 1, 2, \dots, 8$ . We see that the selection of the starting subdomain significantly affects the rate of convergence of the SCO interface relaxation method.

Finally we test if the convergence of the methods depends on the definiteness of the PDE operator. We consider the following two differential equations,  $u'' + \sin(2\pi x)u = 0$  and  $u'' + \cos(2\pi x)u = 0$ , that do not satisfy the ellipticity condition and appear (in a 2-dimensional form) in practical applications. Only four methods (GEO, NEW, SCO, SHO) converge for these indefinite problems. Figure 9 shows the convergence behavior of these methods for both problems. We see that the convergence rate is comparable to that seen in Figure 3. The rest either diverge or oscillate. Such behavior has been already noticed for some of the methods [10].

## 5 Conclusions

We present a wide class of non-overlapping domain decomposition, interface relaxation methods for elliptic differential equations. A set of experiments are described which explore the convergence properties of these methods in several directions. The qualitative conclusions are categorized in Figure 10. This figure also summarizes the mathematical and computational properties of the

	Effects of											
	Speed	Convergence Monotonicity	Parameters	Theory Available	History	Single Step	No Subdomains	Domain Discretization	PDE Discretization	PDE Operator	PDE Domain	Interface Position
AVE	—		2	<b>Some</b>		No						
GEO	—	<b>Yes</b>	2		<b>Yes</b>	<b>Yes</b>						
LSQ	Low	<b>Yes</b>	None			<b>Yes</b>				High	High	
NEW	—		None		<b>Yes</b>	<b>Yes</b>				High		
ROB	Low		1	<b>Some</b>		<b>Yes</b>	Low	Low	Low	Low	Low	Low
SCO	—		1	<b>Some</b>	<b>Yes</b>	No						
SHO	—	<b>Yes</b>	None	<b>Some</b>	<b>Yes</b>	<b>Yes</b>						
SPO	High	<b>Yes</b>	1		<b>Yes</b>	No			Low			

Fig. 10. Categorization of the properties of the eight interface relaxation methods. A blank entry indicates an “average” evaluation (for numerical properties) or absence of a property. Those evaluations considered to be positive are given in larger, bolder type.

methods. It is seen that the speed of convergence of the interface relaxation methods can be of high, moderate or low, that the iterates can approach the exact solution monotonically or not, and there can be two, one or no relaxation parameters to accelerate the convergence. Some single or two step interface relaxation methods use “history” (the new value on the interface is explicitly set to be the old one plus a correction term), some do not. It is natural to expect that the rate of convergence of all interface relaxation methods is affected, to some extent, by certain problem parameters. Some of the most important of these parameters are the fineness of the mesh or grid discretization of the domains (column “domain discretization” in Figure reftbl:conc), the particular method (finite element, finite difference . . .) used to discretize the PDE operators (column “PDE discretization”) and the geometric characteristics of the domains (column “PDE domain”). The theory available to explain all these cases and phenomena is very limited.

We start our iterations with a zero initial guess. Nevertheless, we expect that for many problems with discontinuities or for two-dimensional problems a more reasonable initial guess will be needed. Such a guess can be obtained by various approximation methods that extend the boundary conditions into the interior of the domain using either a blending technique [28] or a wavelet

approach [16]. In any case, better initial guesses provide faster solutions and more robust computations.

The principal conclusions of this study are: (1) There are many interface relaxation methods that work, (2) There is still much to be learned about their behavior and about how to choose among them or to choose their parameters.

## References

- [1] A. Bamberger, R. Glowinski, and Q.H. Tran. A domain decomposition method for the acoustic wave equation with discontinuous coefficients and grid change. *SIAM J. Numer. Anal.*, 34:??-??, 1997.
- [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(2):1053-1061, 1989.
- [3] T. F. Chan, T. Y. Hou, and P. L. Lions. Geometry related convergence results for domain decomposition algorithms. *SIAM J. Numer. Anal.*, 28(2):378-391, 1991.
- [4] T. F. Chan and D. C. Resasco. A survey of preconditioners for domain decomposition. Technical Report /DCS/RR-414, Yale University, 1985.
- [5] Tony F. Chan and Danny Goovaerts. On the relationship between overlapping and nonoverlapping domain decomposition methods. *SIAM J. Matrix Anal. Appl.*, 13:663-670, 1992.
- [6] Tony F. Chan and Thomas Y. Hou. Eigendecomposition of domain decomposition interface operators for constant coefficient elliptic problems. *SIAM J. Sci. Stat. Comput.*, 12:1471-1479, 1991.
- [7] Tony F. Chan and Tarek P. Mathew. Domain decomposition algorithms. In *Acta Numerica 1994*, pages 61-143. Cambridge University Press, 1994.
- [8] Q. Deng. An analysis for a nonoverlapping domain decomposition iterative procedure. *SIAM J. Sci. Comput.*, 18:1517-1525, 1997.
- [9] B. Despres. Domain decomposition method and the Helmholtz problem. In L. Halpern G. Cohen and P. Joly, editors, *Mathematical and Numerical Aspects of Wave Propagation Phenomena*, pages 44-52. SIAM, 1991.
- [10] B. Despres. *Methodes de decomposition de domaines pour les problemes de propagation d'ondes en regime harmonique*. PhD thesis, Universite Paris IX Dauphine, UER Mathematiques de la Decision, 1991.
- [11] M. R. Dorr. Domain decomposition via Lagrange multipliers. Technical Report 98532, Lawrence Livermore National Laboratory, Livermore. CA, 1988.



- [12] M. R. Dorr. On the discretization of interdomain coupling in elliptic boundary-value problems. In Roland Glowinski, Gene H. Golub, Gérard A. Meurant, and Jacques Périaux, editors, *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 17–37, SIAM, Philadelphia, PA, 1988.
- [13] T.T. Drashansky. *An agent-based approach to building multidisciplinary problem solving environments*. PhD thesis, Purdue University, Computer Science Department, December 1996.
- [14] D. Funaro, A. Quarteroni, and P. Zanolli. An iterative procedure with interface relaxation for domain decomposition methods. *SIAM J. Numer. Anal.*, 25(6):1213–1236, 1988.
- [15] W. Heinrichs. Domain decomposition for fourth-order problems. *SIAM J. Numer. Anal.*, 30(2):435–453, 1993.
- [16] S. Jaffard. Wavelet methods for fast resolution of elliptic problems. *SIAM J. Numer. Anal.*, 29:965–986, 1992.
- [17] J. Douglas Jr. and C-S Huang. An accelerated domain decomposition procedure based on Robin transmission conditions. Technical report, Department of Mathematics, Purdue University, 1996.
- [18] J. Douglas Jr., P. J. Paes Leme, J. E. Roberts, and J. Wang. A parallel iterative procedure applicable to the approximate solution of the second order partial differential equations by mixed finite element methods. *Numer. Math.*, 65:95–108, 1993.
- [19] 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:s166–s202, 1987.
- [20] S. Kim. A parallelizable iterative procedure for the Helmholtz problem. *Appl. Numer. Math.*, 17:411–??, 1995.
- [21] C. H. Lai. On domain decomposition and shooting methods for two-point boundary value problem. In David E. Keyes and Jinchao Xu, editors, *Seventh International Conference of Domain Decomposition Methods in Scientific and Engineering Computing*, volume 180 of *Contemporary Mathematics*, pages 257–264. AMS, 1994.
- [22] P. Le-Tallec, Y. De-Roecl, and M. Vidrascu. Domain decomposition methods for large linearly elliptic 3-dimensional problems. *J. Comput. Appl. Math.*, 34(1):93–117, 1991.
- [23] P. L. Lions. On the Schwarz alternating method. III: a variant for nonoverlapping subdomains. In Tony F. Chan, Roland Glowinski, Jacques Périaux, and Olof Widlund, editors, *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, SIAM, Philadelphia, PA, 1990.

- [24] M. Mu. Solving composite problems with interface relaxation. *SIAM J. Sci. Comput.*, 1998 (to appear).
- [25] M. Mu and J.R. Rice. Modeling with collaborating PDE solvers - theory and practice. *Computing Systems in Engineering*, 6:87–95, 1995.
- [26] R. Natarajan. Domain decomposition using spectral expansions of Steklov–Poincaré operators. *SIAM J. Sci. Comput.*, 16(2):470–485, 1995.
- [27] R. Natarajan. Domain decomposition using spectral expansions of Steklov–Poincaré operators II: A matrix formulation. *SIAM J. Sci. Comput.*, 18:to appear, 1997.
- [28] J.R. Rice and R.F. Boisvert. *Solving Elliptic Problems Using ELLPACK*. Springer-Verlag, New York, NY, 1985.
- [29] J.R. Rice, P. Tsombanopoulou, E. Vavalis, and D. Yang. Domain decomposition methods for underwater acoustics problems. Technical report, Computer Science Department, Purdue University, W. Lafayette, IN, in preparation.
- [30] J.R. Rice, E. Vavalis, and D. Yang. Analysis of a non-overlapping domain decomposition method for elliptic PDEs. *J. Comput. Appl. Math.*, 87:11–19, 1998.
- [31] H.T.M. van der Maarel and A. Platschorre. Optimization of flexible computing in domain decomposition for a system of pdes. In *Proc. of the Ninth Intern. Conf. on Domain Decomposition Methods*. 1998 (to appear).
- [32] J. Xu and J. Zou. Non-overlapping domain decomposition methods. Technical report, Mathematics Department, Pennsylvania State University, University Park, PA, 1996.
- [33] D. Yang. A parallel iterative nonoverlapping domain decomposition procedure for elliptic problems. *IMA J. Numer. Anal.*, 16:75–91, 1996.
- [34] D. Yang. A parallel domain decomposition algorithm for elliptic problems. *J. Comp. Math.*, 16:141–151, 1998.

## Appendix

In this appendix we give the detailed algorithms for the eight relaxation methods in the 1-dimensional case.

**Algorithm 1.** The Dirichlet/Neumann Averaging (AVE) Method

$$g_i^i = \beta_i \frac{du_i^{(2k)}}{dx} \Big|_{x=x_i} + (1 - \beta_i) \frac{du_{i+1}^{(2k)}}{dx} \Big|_{x=x_i}, \quad i = 1, \dots, p-1.$$

$$Lu_1^{(2k+1)} = f \text{ in } \Omega_1 \quad \left| \quad \begin{array}{l} Lu_i^{(2k+1)} = f \text{ in } \Omega_i \\ \frac{du_i^{(2k+1)}}{dx} \Big|_{x=x_{i-1}} = g_{i-1}^{i-1}, \quad i = 2, \dots, p-1 \\ \frac{du_i^{(2k+1)}}{dx} \Big|_{x=x_i} = g_i^i \end{array} \right. \quad \left| \quad \begin{array}{l} Lu_p^{(2k+1)} = f \text{ in } \Omega_p \\ \frac{du_p^{(2k+1)}}{dx} \Big|_{x=x_{p-1}} = g_{p-1}^{p-1} \\ u_p^{(2k+1)} \Big|_{x=x_p} = 0 \end{array} \right.$$

$$h_i^i = \alpha_i u_i^{(2k+1)} \Big|_{x=x_i} + (1 - \alpha_i) u_{i+1}^{(2k+1)} \Big|_{x=x_i}, \quad i = 1, \dots, p-1.$$

$$Lu_1^{(2k+2)} = f \text{ in } \Omega_1 \quad \left| \quad \begin{array}{l} Lu_i^{(2k+2)} = f \text{ in } \Omega_i \\ u_i^{(2k+2)} \Big|_{x=x_{i-1}} = h_{i-1}^{i-1}, \quad i = 2, \dots, p-1 \\ u_i^{(2k+2)} \Big|_{x=x_i} = h_i^i \end{array} \right. \quad \left| \quad \begin{array}{l} Lu_p^{(2k+2)} = f \text{ in } \Omega_p \\ u_p^{(2k+2)} \Big|_{x=x_{p-1}} = h_{p-1}^{p-1} \\ u_p^{(2k+2)} \Big|_{x=x_p} = 0 \end{array} \right.$$

**Algorithm 2.** The Geometric (GEO) Contraction Based Method

$$g_i^i = \frac{u_i^{(k)} + u_{i+1}^{(k)}}{2} \Big|_{x=x_i} - \frac{w_i^i w_{i+1}^{i+1}}{w_i^i + w_{i+1}^{i+1}} \left( \frac{du_i^{(k)}}{dx} + \frac{du_{i+1}^{(k)}}{dx} \right) \Big|_{x=x_i} \quad \left. \right\} i = 1, \dots, p-1.$$

$$Lu_1^{(k+1)} = f \text{ in } \Omega_1 \quad \left| \quad \begin{array}{l} Lu_p^{(k+1)} = f \text{ in } \Omega_p \\ u_p^{(k+1)} \Big|_{x=x_{p-1}} = g_{p-1}^{p-1} \\ u_p^{(k+1)} \Big|_{x=x_p} = 0 \end{array} \right.$$

$$\left. \begin{array}{l} Lu_i^{(k+1)} = f \text{ in } \Omega_i \\ u_i^{(k+1)} \Big|_{x=x_{i-1}} = g_{i-1}^{i-1} \\ u_i^{(k+1)} \Big|_{x=x_i} = g_i^i \end{array} \right\} i = 2, \dots, p-1.$$

**Algorithm 3.** The Least Squares (LSQ) Method (with mesh parameter  $h$ )

$$\left. \begin{aligned}
 h_i^i &= \frac{u_i^{(k)}|_{x=x_i} + u_{i+1}^{(k)}|_{x=x_i}}{2} + \frac{u_i^{(k)}|_{x=x_i} - u_{i+1}^{(k)}|_{x=x_i}}{2(h^4+1)} \\
 &\quad \frac{h}{3} \left( \frac{du_i^{(k)}}{dx} \Big|_{x=x_i} - \frac{du_{i+1}^{(k)}}{dx} \Big|_{x=x_i} \right) - \frac{h^2}{2(h^4+1)} \left( \frac{d^2u_i^{(k)}}{dx^2} \Big|_{x=x_i} - \frac{d^2u_{i+1}^{(k)}}{dx^2} \Big|_{x=x_i} \right) \\
 h_i^{i+1} &= \frac{u_i^{(k)}|_{x=x_i} + u_{i+1}^{(k)}|_{x=x_i}}{2} - \frac{u_i^{(k)}|_{x=x_i} - u_{i+1}^{(k)}|_{x=x_i}}{2(h^4+1)} \\
 &\quad \frac{h}{3} \left( \frac{du_i^{(k)}}{dx} \Big|_{x=x_i} - \frac{du_{i+1}^{(k)}}{dx} \Big|_{x=x_i} \right) + \frac{h^2}{2(h^4+1)} \left( \frac{d^2u_i^{(k)}}{dx^2} \Big|_{x=x_i} - \frac{d^2u_{i+1}^{(k)}}{dx^2} \Big|_{x=x_i} \right)
 \end{aligned} \right\} i = 1, \dots, p-1.$$

$$\begin{array}{l}
 Lu_1^{(k+1)} = f \text{ in } \Omega_1 \\
 u_1^{(k+1)}|_{x=x_0} = 0 \\
 u_1^{(k+1)}|_{x=x_1} = h_1^1
 \end{array}
 \left| \begin{array}{l}
 Lu_i^{(k+1)} = f \text{ in } \Omega_i \\
 u_i^{(k+1)}|_{x=x_{i-1}} = h_{i-1}^i, \quad i = 2, \dots, p-1 \\
 u_i^{(k+1)}|_{x=x_i} = h_i^i
 \end{array} \right.
 \begin{array}{l}
 Lu_p^{(k+1)} = f \text{ in } \Omega_p \\
 u_p^{(k+1)}|_{x=x_{p-1}} = h_{p-1}^p \\
 u_p^{(k+1)}|_{x=x_p} = 0
 \end{array}$$

**Algorithm 4.** The Newton's (NEW) Method

$$\left. \begin{aligned}
 \alpha_i^i &= \frac{\frac{du_i^{(k)}}{dx} \Big|_{x=x_i} - \frac{du_i^{(k-1)}}{dx} \Big|_{x=x_i}}{u_i^{(k)} \Big|_{x=x_i} - u_i^{(k-1)} \Big|_{x=x_i}} \\
 \alpha_i^{i+1} &= \frac{\frac{du_{i+1}^{(k)}}{dx} \Big|_{x=x_i} - \frac{du_{i+1}^{(k-1)}}{dx} \Big|_{x=x_i}}{u_{i+1}^{(k)} \Big|_{x=x_i} - u_{i+1}^{(k-1)} \Big|_{x=x_i}}
 \end{aligned} \right\} i = 1, \dots, p-1$$

$$\left. \begin{aligned}
 h_i^i &= u_i^{(k)} \Big|_{x=x_i} + \frac{\alpha_i^{i+1} u_{i+1}^{(k)} \Big|_{x=x_i} - \alpha_i^{i+1} u_i^{(k)} \Big|_{x=x_i} - \frac{du_{i+1}^{(k)}}{dx} \Big|_{x=x_i} + \frac{du_i^{(k)}}{dx} \Big|_{x=x_i}}{\alpha_i^{i+1} - \alpha_i^i} \\
 h_i^{i+1} &= u_{i+1}^{(k)} \Big|_{x=x_i} + \frac{\alpha_i^i u_{i+1}^{(k)} \Big|_{x=x_i} - \alpha_i^i u_i^{(k)} \Big|_{x=x_i} - \frac{du_{i+1}^{(k)}}{dx} \Big|_{x=x_i} + \frac{du_i^{(k)}}{dx} \Big|_{x=x_i}}{\alpha_i^{i+1} - \alpha_i^i}
 \end{aligned} \right\} i = 1, \dots, p-1.$$

$$\begin{array}{l}
 Lu_1^{(k+1)} = f \text{ in } \Omega_1 \\
 u_1^{(k+1)} \Big|_{x=x_0} = 0 \\
 u_1^{(k+1)} \Big|_{x=x_1} = h_1^1
 \end{array}
 \left| \begin{array}{l}
 Lu_i^{(k+1)} = f \text{ in } \Omega_i \\
 u_i^{(k+1)} \Big|_{x=x_{i-1}} = h_{i-1}^i, \quad i = 2, \dots, p-1 \\
 u_i^{(k+1)} \Big|_{x=x_i} = h_i^i
 \end{array} \right.
 \left| \begin{array}{l}
 Lu_p^{(k+1)} = f \text{ in } \Omega_p \\
 u_p^{(k+1)} \Big|_{x=x_{p-1}} = h_{p-1}^p \\
 u_p^{(k+1)} \Big|_{x=x_p} = 0
 \end{array}$$

**Algorithm 5.** The Robin Relaxation (ROB) Method

$$\left. \begin{aligned}
 g_i^i &= \frac{du_{i+1}^{(k)}}{dx} \Big|_{x=x_i} + \lambda_i u_{i+1}^{(k)} \Big|_{x=x_i} \\
 g_i^{i+1} &= \frac{du_i^{(k)}}{dx} \Big|_{x=x_i} + \lambda_i \frac{du_i^{(k)}}{dx} \Big|_{x=x_i}
 \end{aligned} \right\} i = 1, \dots, p-1.$$

$$\begin{array}{l}
 Lu_1^{(k+1)} = f \text{ in } \Omega_1 \\
 u_1^{(k+1)} \Big|_{x=x_0} = 0 \\
 \frac{du_1^{(k+1)}}{dx} \Big|_{x=x_1} + \lambda_1 u_1^{(k+1)} \Big|_{x=x_1} = g_1^1
 \end{array}
 \left| \begin{array}{l}
 Lu_p^{(k+1)} = f \text{ in } \Omega_p \\
 \frac{du_p^{(k+1)}}{dx} \Big|_{x=x_{p-1}} + \lambda_{p-1} u_p^{(k+1)} \Big|_{x=x_{p-1}} = g_{p-1}^p \\
 u_p^{(k+1)} \Big|_{x=x_p} = 0
 \end{array} \right.$$

$$\left. \begin{aligned}
 Lu_i^{(k+1)} &= f \text{ in } \Omega_i \\
 \frac{du_i^{(k+1)}}{dx} \Big|_{x=x_{i-1}} + \lambda_{i-1} u_i^{(k+1)} \Big|_{x=x_{i-1}} &= g_{i-1}^i \\
 \frac{du_i^{(k+1)}}{dx} \Big|_{x=x_i} + \lambda_i u_i^{(k+1)} \Big|_{x=x_i} &= g_i^i
 \end{aligned} \right\} i = 2, \dots, p-1.$$

**Algorithm 6.** The Schur complement (SCO) Method

For a  $c \in \{1, \dots, p\}$

$$\left. \begin{aligned}
 h_{i-1}^i &= \begin{cases} 0 & , i = 1 \\
 \theta_{i-1} u_{i-1}^{(k+1)} \Big|_{x=x_{i-1}} + (1 - \theta_{i-1}) u_i^{(k)} \Big|_{x=x_{i-1}} & , i = 2, \dots, c-1 \end{cases} \\
 g_i^i &= \frac{du_{i+1}^{(k)}}{dx} \Big|_{x=x_i} \\
 Lu_i^{(k+1)} &= f \text{ in } \Omega_i \ , \ u_i^{(k+1)} \Big|_{x=x_{i-1}} = h_{i-1}^i \ , \ \frac{du_i^{(k+1)}}{dx} \Big|_{x=x_i} = g_i^i
 \end{aligned} \right\} i = 1, \dots, c-1$$

$$h_{c-1}^c = \theta_{c-1} u_{c-1}^{(k+1)} \Big|_{x=x_{c-1}} + (1 - \theta_{c-1}) u_c^{(k)} \Big|_{x=x_{c-1}}$$

$$h_c^c = \theta_c u_{c+1}^{(k)} \Big|_{x=x_c} + (1 - \theta_c) u_c^{(k)} \Big|_{x=x_c}$$

$$Lu_c^{(k+1)} = f \text{ in } \Omega_c \ , \ u_c^{(k+1)} \Big|_{x=x_{c-1}} = h_{c-1}^c \ , \ u_c^{(k+1)} \Big|_{x=x_c} = h_c^c$$

$$\left. \begin{aligned}
 h_i^i &= \begin{cases} \theta_i u_{i+1}^{(k)} \Big|_{x=x_i} + (1 - \theta_i) u_i^{(k)} \Big|_{x=x_i} & , i = c+1, \dots, p-1 \\
 0 & , i = p \end{cases} \\
 g_{i-1}^i &= \frac{du_{i-1}^{(k+1)}}{dx} \Big|_{x=x_{i-1}} \\
 Lu_i^{(k+1)} &= f \text{ in } \Omega_i \ , \ \frac{du_i^{(k+1)}}{dx} \Big|_{x=x_{i-1}} = g_{i-1}^i \ , \ u_i^{(k+1)} \Big|_{x=x_i} = h_i^i
 \end{aligned} \right\} i = c+1, \dots, p$$

**Algorithm 7. The Shooting (SHO) Method**

$$\left. \begin{aligned} g_i^i &= \left. \frac{du_{i+1}^{(k)}}{dx} \right|_{x=x_i} + \lambda_i u_{i+1}^{(k)} \Big|_{x=x_i} \\ g_i^{i+1} &= \left. \frac{du_i^{(k)}}{dx} \right|_{x=x_i} + \lambda_i du_i^{(k)} \Big|_{x=x_i} \end{aligned} \right\} i = 1, \dots, p-1.$$

$$\left. \begin{aligned} Lu_1^{(k+1)} &= f \text{ in } \Omega_1 \\ u_1^{(k+1)} \Big|_{x=x_0} &= 0 \\ \left. \frac{du_1^{(k+1)}}{dx} \right|_{x=x_1} + \lambda_1 u_1^{(k+1)} \Big|_{x=x_1} &= g_1^1 \end{aligned} \right\} \left. \begin{aligned} Lu_p^{(k+1)} &= f \text{ in } \Omega_p \\ \left. \frac{du_p^{(k+1)}}{dx} \right|_{x=x_{p-1}} + \lambda_{p-1} u_p^{(k+1)} \Big|_{x=x_{p-1}} &= g_{p-1}^p \\ u_p^{(k+1)} \Big|_{x=x_p} &= 0 \end{aligned} \right.$$

$$\left. \begin{aligned} Lu_i^{(k+1)} &= f \text{ in } \Omega_i \\ \left. \frac{du_i^{(k+1)}}{dx} \right|_{x=x_{i-1}} + \lambda_{i-1} u_i^{(k+1)} \Big|_{x=x_{i-1}} &= g_{i-1}^i \\ \left. \frac{du_i^{(k+1)}}{dx} \right|_{x=x_i} + \lambda_i u_i^{(k+1)} \Big|_{x=x_i} &= g_i^i \end{aligned} \right\} i = 2, \dots, p-1.$$

**Algorithm 8. The Steklov–Poincaré operator (SPO) Method**

$$\left. \begin{aligned} Lu_1^{(k+1)} &= f \text{ in } \Omega_1 \\ u_1^{(k+1)} \Big|_{x=x_0} &= 0 \\ u_1^{(k+1)} \Big|_{x=x_1} &= h_1^1 \end{aligned} \right\} \left. \begin{aligned} Lu_i^{(k+1)} &= f \text{ in } \Omega_i \\ u_i^{(k+1)} \Big|_{x=x_{i-1}} &= h_{i-1}^i, \quad i = 2, \dots, p-1 \\ u_i^{(k+1)} \Big|_{x=x_i} &= h_i^i \end{aligned} \right\} \left. \begin{aligned} Lu_p^{(k+1)} &= f \text{ in } \Omega_p \\ u_p^{(k+1)} \Big|_{x=x_{p-1}} &= h_{p-1}^p \\ u_p^{(k+1)} \Big|_{x=x_p} &= 0 \end{aligned} \right.$$

$$g_i^{i+1} = g_i^i = \frac{1}{2} \left( \left. \frac{du_i^{(k+1)}}{dx} \right|_{x=x_i} - \left. \frac{du_{i+1}^{(k+1)}}{dx} \right|_{x=x_i} \right), \quad i = 1, \dots, p-1.$$

$$\left. \begin{aligned} L\phi_1^{(k+1)} &= f \text{ in } \Omega_1 \\ \phi_1^{(k+1)} \Big|_{x=x_0} &= 0 \\ \left. \frac{d\phi_1^{(k+1)}}{dx} \right|_{x=x_1} &= g_1^1 \end{aligned} \right\} \left. \begin{aligned} L\phi_i^{(k+1)} &= f \text{ in } \Omega_i \\ \left. \frac{d\phi_i^{(k+1)}}{dx} \right|_{x=x_{i-1}} &= g_{i-1}^i, \quad i = 2, \dots, p-1 \\ \left. \frac{d\phi_i^{(k+1)}}{dx} \right|_{x=x_i} &= g_i^i \end{aligned} \right\} \left. \begin{aligned} L\phi_p^{(k+1)} &= f \text{ in } \Omega_p \\ \left. \frac{d\phi_p^{(k+1)}}{dx} \right|_{x=x_{p-1}} &= g_{p-1}^p \\ \phi_p^{(k+1)} \Big|_{x=x_p} &= 0 \end{aligned} \right.$$

$$h_i^{i+1} = h_i^i = u_i^i - \frac{\rho_i}{2} \left( \phi_i^{(k+1)} \Big|_{x=x_i} + \phi_{i+1}^{(k+1)} \Big|_{x=x_i} \right), \quad i = 1, \dots, p-1.$$