View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Elsevier - Publisher Connector

ELSEVIER

Journal of Computational and Applied Mathematics 50 (1994) 509–521

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

An interval arithmetic domain decomposition method for a class of elliptic PDEs on nonrectangular domains

Hartmut Schwandt *

Department of Mathematics, Technical University of Berlin, Strasse des 17. Juni 136, D-10623 Berlin, Germany

Received 28 July 1992; revised 21 December 1992

Abstract

We introduce an interval arithmetic domain decomposition method for linear systems with interval coefficients resulting from the application of difference methods for a class of elliptic boundary value problems on domains with irregular geometry. The efficient treatment of such systems is crucial for the efficiency of globally convergent Newton-like interval methods for the corresponding nonlinear problems.

Key words: Domain decomposition; Interval arithmetic; Elliptic boundary value problems; Nonrectangular domains

1. Introduction

During the last years, domain decomposition methods have been widely used in the numerical solution of partial differential equations. By choosing individually adapted methods on subdomains, we introduce a domain decomposition method for the efficient inclusion of the set of solutions of a class of systems of linear equations with interval coefficients. As these systems occur in each step of Newton-like interval methods for nonlinear systems resulting from the discretization of elliptic boundary value problems by difference methods, this linear method is essentially motivated by nonlinear problems. Newton-like interval methods are attractive because of their almost global convergence under appropriate conditions. Their efficiency strongly depends on the linear solver used in each step. Domain decomposition enables us to introduce an efficient solver on nonrectangular domains. We assume an almost linear boundary value problem

$$\begin{aligned} &Lu = h(u), && \text{on a nonrectangular domain } \Omega \subset \mathbb{R}^2, \\ &r\left(u, \frac{\partial u}{\partial n}\right) = 0, && \text{on } \Gamma_\Omega, \end{aligned} \tag{1}$$

* e-mail: schwandt@math.tu-berlin.de.

0377-0427/94/$07.00 © 1994 Elsevier Science B.V. All rights reserved
SSDI 0377-0427(93)E0150-K

where $u \equiv u(x, y)$, $h(u) \equiv h(x, y, u(x, y))$, L is a second-order linear elliptic operator, r is linear (Dirichlet, Neumann or mixed). We do not prescribe a specific discretization; in particular we do not specify the approximation near the irregular boundaries. We only require a nonlinear system

$$\boldsymbol{F}\boldsymbol{u} + \boldsymbol{\phi}(\boldsymbol{u}) = 0, \text{ in } \mathbb{R}^N, \quad \boldsymbol{F} = (-\boldsymbol{S}_j, \boldsymbol{A}_j, -\boldsymbol{T}_j), \ \boldsymbol{\phi}(\boldsymbol{u}) = (\phi_i(u_i))_{i=1}^N, \tag{2}$$

with block-tridiagonal $\boldsymbol{F}$ and a diagonal function $\boldsymbol{\phi}$. We mainly assume that $\boldsymbol{F}$ is an M-matrix ($f_{j,i} \leqslant 0$ for $i \neq j$, $\boldsymbol{F}^{-1} \geqslant \boldsymbol{0}$) and that $\boldsymbol{\phi}'(\boldsymbol{u})$ is a diagonal matrix with $\boldsymbol{\phi}'(\boldsymbol{u}) \geqslant \boldsymbol{0}$, hence $\boldsymbol{f}'(\boldsymbol{u}) = \boldsymbol{F} + \boldsymbol{\phi}'(\boldsymbol{u})$ is also an M-matrix.

As a typical simple example, we mention the almost linear Dirichlet problem

$$\begin{aligned} &a(x, y)u_{xx} + c(x, y)u_{yy} + d(x, y)u_x + e(x, y)u_y = f(x, y, u), \\ &\quad \text{on a domain } \Omega \subset \mathbb{R}^2, \\ &u(x, y) = g(x, y), \quad \text{on } \Gamma_\Omega, \qquad a, c > 0, \ f_u \geqslant 0. \end{aligned} \tag{3}$$

The standard five-point discretization with central difference quotients and with the usual restriction of the mesh size

$$h \leqslant 2 \min\left\{ \frac{\min\{a(x, y)\}}{\max\{|d(x, y)|\}}, \frac{\min\{c(x, y)\}}{\max\{|e(x, y)|\}} \right\}$$

yields a system of the above form with

$$\boldsymbol{A}_j = \left(-\left(a_{i,j} - \tfrac{1}{2}hd_{i,j}\right), 2(a_{i,j} + c_{i,j}), -\left(a_{i,j} + \tfrac{1}{2}hd_{i,j}\right)\right), \qquad \boldsymbol{S}_j = -\operatorname{diag}\left(c_{i,j} - \tfrac{1}{2}he_{i,j}\right),$$

$$\boldsymbol{T}_j = -\operatorname{diag}\left(b_{i,j} + \tfrac{1}{2}he_{i,j}\right),$$

for $a_{i,j} = a(x_i, y_j)$, $c_{i,j} = c(x_i, y_j)$, $d_{i,j} = d(x_i, y_j)$, $e_{i,j} = e(x_i, y_j)$ on grid points (x_i, y_i) (see, e.g., [12]).

The derivation of the interval domain decomposition method (abbreviated by IDD in the sequel) will show that its properties do not depend on a specific discretization, but more generally on algebraic properties like the above-mentioned.

A starting interval vector $\mathbf{u}^0$ including the (unique under the above conditions [9, Theorem 4.4.1]) solution y of (2) for nonlinear interval methods into which IDD has to be integrated can be computed from the relation

$$\boldsymbol{y} \in \left[-\boldsymbol{F}^{-1}|\boldsymbol{\phi}(0)|, \boldsymbol{F}^{-1}|\boldsymbol{\phi}(0)|\right] \subseteq \mathbf{u}^0 \tag{4}$$

(see [9, Theorem 13.4.6c]). A typical class of Newton-like interval methods has the following form:

$$\mathbf{u}^{n+1} := \{\boldsymbol{m}^n - \mathrm{LES}(\mathbf{M}(\mathbf{u}^n), \mathbf{N}(\mathbf{u}^n)(\boldsymbol{m}^n - \mathbf{u}^n) + \boldsymbol{\phi}(\boldsymbol{m}^n))\} \cap \mathbf{u}^n. \tag{5}$$

LES denotes a suitable interval method for the treatment of systems $(\mathbf{M}, \mathbf{y})$ of linear form with interval coefficients, i.e., $\mathbf{M}$ and $\mathbf{y}$ have real compact intervals as coefficients. The iterates $\mathbf{u}^n$

are interval vectors, $\boldsymbol{m}^n \in \mathbf{u}^n$ are arbitrarily chosen real-point vectors and $\mathbf{M}(\mathbf{u}^n)$, $\mathbf{N}(\mathbf{u}^n)$ are matrices with interval coefficients derived from a decomposition of the Jacobian $f'(\boldsymbol{u}) = \boldsymbol{M}(\boldsymbol{u}) - \boldsymbol{N}(\boldsymbol{u})$, in the present context typically $f'(\boldsymbol{u}) = \boldsymbol{M} - \boldsymbol{N} + \boldsymbol{\phi}'(\boldsymbol{u})$, $\boldsymbol{F} = \boldsymbol{M} - \boldsymbol{N}$, and a subsequent interval arithmetic evaluation, i.e., the real arithmetic or functional expressions defining the coefficients are replaced by corresponding interval expressions where necessary to enclose $\boldsymbol{y}$ or expressions depending on it. Several Newton-like interval methods for nonlinear systems (2) have been introduced (e.g., [1,4,10,11]). For methods of the so-called Krawczyk-type, for example, with $\mathbf{M}(\mathbf{u}^n) = \boldsymbol{M} + \boldsymbol{\Phi}_n$, $\mathbf{N}(\mathbf{u}^n) = \boldsymbol{\Phi}_n - \boldsymbol{\phi}'(\mathbf{u}^n) + \boldsymbol{N}$, where the iteration matrix degenerates to a real point matrix, convergence to the solution can be shown under rather weak conditions. In particular, it can be shown that $\forall n \in N_0$: $\boldsymbol{y} \in \mathbf{u}^{n+1} \subseteq \mathbf{u}^n$ and that $\mathbf{u}^n \to \boldsymbol{y}$, $n \to \infty$. The essential conditions depending on the given problem, which are satisfied in problems like (1), (2), are that $\boldsymbol{F}$, $\boldsymbol{M}$ have to be M-matrices, $\boldsymbol{N} \geqslant \boldsymbol{0}$ (componentwise) and $\boldsymbol{\phi}'(\boldsymbol{u}) \geqslant \boldsymbol{0}$ in $\mathbf{u}^0$. The major problem is to find an efficient method LES with satisfactory inclusion properties for the linear interval systems $(\mathbf{M}, \mathbf{y})$ occurring in each iteration. In contrast to real linear systems, an interval method LES yields an inclusion

$$\mathbf{x} := \mathrm{LES}(\mathbf{M}, \mathbf{y}) \supseteq \mathrm{SOL}(\mathbf{M}, \mathbf{y}) := \{\boldsymbol{x} \in \mathbb{R}^N \mid \boldsymbol{M}\boldsymbol{x} = \boldsymbol{y},\ \boldsymbol{M} \in \mathbf{M},\ \boldsymbol{y} \in \mathbf{y}\} \tag{6}$$

of the set of solutions of all linear systems $\boldsymbol{M}\boldsymbol{x} = \boldsymbol{y}$ which can be defined by choosing arbitrary point matrices $\boldsymbol{M} \in \mathbf{M}$ and point vectors $\boldsymbol{y} \in \mathbf{y}$. As the set $\mathrm{SOL}(\mathbf{M}, \mathbf{y})$ is not necessarily an interval vector, the interval vector $\mathbf{x}$ can only be an inclusion (see e.g., [1,2]). The best result one can expect is an *optimal inclusion*, the tightest possible interval vector which still includes the set of solutions.

This explains the motivation of a method of linear form in a nonlinear context. If Ω is rectangular or has a similar simple geometry such that the linear part of the Jacobian is block-tridiagonal $\boldsymbol{F} = (-\boldsymbol{S}, \boldsymbol{A}, -\boldsymbol{T})$ with identical block rows, the only known interval "fast solver", an interval version of stabilized block cyclic reduction ("interval Buneman") IBU, can be used [10,12,13]. For more general geometries leading to general block tridiagonal or related matrices, one only has the choice between block forms of interval Gaussian elimination or iterative methods based on regular splittings (e.g., [1,5,7,8]), which are not very efficient for systems resulting from problems like (1)–(3). Interval arithmetic equivalents of efficient point methods like conjugate gradient, multigrid methods or SOR (with optimal parameters) with satisfactory inclusion properties are not known. Here, we introduce an interval arithmetic domain decomposition method IDD for a class of block-tridiagonal systems (6) which have to be treated in Newton-like methods (5) for discretizations (2) of boundary value problems like (1)–(3) for not necessarily rectangular Ω. For the discussion of the linear method IDD one obviously does not need any nonlinear properties. We discuss its applicability, the most important inclusion properties and its efficiency, illustrated by numerical results.

Notation. We denote real numbers or indices by $a, \ldots, z$, real compact intervals by $A, \ldots, Z$ with bounds $X = [\boldsymbol{i}(X), \boldsymbol{s}(X)]$, real vectors and matrices by $\boldsymbol{a}, \ldots, \boldsymbol{z}$ and $\boldsymbol{A}, \ldots, \boldsymbol{Z}$, respectively, interval vectors by $\mathbf{a}, \ldots, \mathbf{z}$ with (componentwise defined) bounds $\mathbf{x} = [\boldsymbol{i}(\mathbf{x}), \boldsymbol{s}(\mathbf{x})]$ and interval matrices by $\mathbf{A}, \ldots, \mathbf{Z}$ with $\mathbf{X} = [\boldsymbol{i}(\mathbf{X}), \boldsymbol{s}(\mathbf{X})]$. For real matrices and vectors we use the partial ordering $\boldsymbol{A} \geqslant 0 \Leftrightarrow a_{i,j} \geqslant 0$, $\forall i, j \in \{1, \ldots, n\}$. We further define $X \geqslant 0 \Leftrightarrow \boldsymbol{i}(X) \geqslant 0$, $X \leqslant 0 \Leftrightarrow \boldsymbol{s}(X) \leqslant 0$.

2. Interval arithmetic domain decomposition

We assume a not necessarily rectangular, simply connected domain $\Omega \subset \mathbb{R}^2$ and an equidistant grid Ω_h on Ω. We decompose Ω into nonoverlapping subdomains: we define rectangles as inner subdomains which do not contain boundary points; the shape of the remaining boundary subdomains obviously depends on the boundary of Ω. This decomposition is motivated by the wish to "fill" a large part of Ω by rectangular subdomains on which interval block cyclic reduction IBU can be applied as "fast solver". On the boundary subdomains we apply a less efficient, more general method like block-tridiagonal Gaussian elimination IBG. IDD is a two-level Schur complement method. On the first level, we decompose Ω into $k+1$ strips Ω_i, $0 \leqslant i \leqslant k$, whose k artificial boundaries Γ_i, $1 \leqslant i \leqslant k$, are defined as grid lines of Ω_h which are parallel to the x-axis:

$$\Omega = \bigcup_{i=0}^{k} \Omega_i \cup \bigcup_{i=1}^{k} \Gamma_i. \tag{7}$$

We assume a block tridiagonal system $(\mathbf{M}, \mathbf{y})$ with $\mathbf{M} = (-S_j, \mathbf{A}_j, -T_j)$ with interval matrices $\mathbf{A}_j$ and point matrices S_j, T_j. As Ω may have a curved boundary, the sizes of S_j, $\mathbf{A}_j$, T_j usually differ. Rearranging the system according to the domain decomposition, we get the formally well-known form (see, e.g., [3])

$$\overline{\mathbf{M}} = \begin{pmatrix} \mathbf{M}_\Omega & K \\ H & \mathbf{M}_\Gamma \end{pmatrix} = \begin{bmatrix} \operatorname{diag}(\mathbf{L}_i \mid 0 \leqslant i \leqslant k) & \operatorname{bidiag}\begin{pmatrix} K_{i-1,i} \\ K_{i,i} \end{pmatrix} \mid 1 \leqslant i \leqslant k \end{pmatrix} \\ \operatorname{bidiag}(H_{i-1,i}, H_{i,i} \mid 1 \leqslant i \leqslant k) & \operatorname{diag}(\mathbf{G}_i \mid 1 \leqslant i \leqslant k) \end{bmatrix}, \tag{8}$$

with block tridiagonal $\mathbf{L}_i = (-S_j, \mathbf{A}_j, -T_j)$, and

$$K_{i-1,i} = \begin{bmatrix} \mathbf{0} \\ -T_{g_i} \end{bmatrix}, \qquad K_{i,i} = \begin{bmatrix} -S_{g_i} \\ \mathbf{0} \end{bmatrix}, \qquad H_{i,i} = \begin{bmatrix} -S_{g_i} & \mathbf{0} \end{bmatrix}, \qquad H_{i-1,i} = \begin{bmatrix} \mathbf{0} & -T_{g_i} \end{bmatrix}$$

(g_i: grid line index of Γ_i). According to this decomposition, we subdivide any interval vector $\mathbf{z}$ into

$$\bar{\mathbf{z}} = \begin{pmatrix} \mathbf{z}_\Omega \\ \mathbf{z}_\Gamma \end{pmatrix}, \quad \mathbf{z}_\Omega = (\mathbf{z}_{\Omega_i})_{i=0}^{k}, \quad \mathbf{z}_\Gamma = (\mathbf{z}_{\Gamma_i})_{i=1}^{k}.$$

In example (3), we assume for simplicity that

$$S_j = s_j I, \qquad T_j = t_j I, \qquad \mathbf{A}_j = (-b, \mathbf{d}_j, -g), \tag{9}$$

and get tridiagonal $\mathbf{G}_i = (-b, \mathbf{d}_i, -g)$ for the equations on the artificial boundaries.

We now formally follow the idea of Schur complement methods (e.g., [3,6]):

(1) compute capacitance system: $\mathbf{C}_\Gamma = \mathbf{M}_\Gamma - H$ LES$(\mathbf{M}_\Omega, K)$; $\mathbf{w}_\Gamma = \mathbf{y}_\Gamma - H$ LES$(\mathbf{M}_\Omega, \mathbf{y}_\Omega)$;

(2) solve the capacitance system: $\mathbf{x}_\Gamma = \text{LES}(\mathbf{C}_\Gamma, \mathbf{w}_\Gamma)$;

(3) solve on the domains: $\mathbf{x}_\Omega = \text{LES}(\mathbf{M}_\Omega, \mathbf{y}_\Omega - K\mathbf{x}_\Gamma)$.

For arbitrary interval matrices $\mathbf{A}$, $\mathbf{B}$ (possibly degenerated to point matrices), LES$(\mathbf{A}, \mathbf{B})$ denotes the treatment of simultaneous systems LES$(\mathbf{A}, \mathbf{b}^j)$ with all columns $\mathbf{b}^j$ of the matrix $\mathbf{B}$ as right-hand sides. Using the above definitions, we obtain the following algorithm.

Algorithm for IDD — level 1.

Step 1. Computation of the capacitance system: $(\mathbf{C}_\Gamma, \mathbf{w}_\Gamma)$, $\mathbf{C}_\Gamma = (\mathbf{B}_{i-1}, \mathbf{A}_i, \mathbf{C}_i,)_{i=1,\dots,k}$, where

$$\begin{aligned}
\mathbf{w}_{\Gamma_i} &= \mathbf{y}_{\Gamma_i} - (\boldsymbol{H}_{i-1,i})^{\mathrm{T}}\,\mathrm{LES}(\mathbf{L}_{i-1}, \mathbf{y}_{\Omega_{i-1}}) - (\boldsymbol{H}_{i,i})^{\mathrm{T}}\,\mathrm{LES}(\mathbf{L}_i, \mathbf{y}_{\Omega_i}), && 1 \leqslant i \leqslant k,\\
\mathbf{A}_i &= \mathbf{G}_i - (\boldsymbol{H}_{i-1,i})^{\mathrm{T}}\,\mathrm{LES}(\mathbf{L}_{i-1}, \boldsymbol{K}_{i-1,i}) - (\boldsymbol{H}_{i,i})^{\mathrm{T}}\,\mathrm{LES}(\mathbf{L}_i, \boldsymbol{K}_{i,i}), && 1 \leqslant i \leqslant k,\\
\mathbf{C}_i &= -(\boldsymbol{H}_{i,i})^{\mathrm{T}}\,\mathrm{LES}(\mathbf{L}_i, \boldsymbol{K}_{i,i+1}), && 1 \leqslant i \leqslant k-1,\\
\mathbf{B}_i &= -(\boldsymbol{H}_{i,i+1})^{\mathrm{T}}\,\mathrm{LES}(\mathbf{L}_i, \boldsymbol{K}_{i,i}), && 2 \leqslant i \leqslant k.
\end{aligned}$$

Step 2. Solution of the capacitance system, i.e., solution on the articifial boundaries:

$$\mathbf{x}_\Gamma = \mathrm{LES}(\mathbf{C}_\Gamma, \mathbf{w}_\Gamma).$$

Step 3. Solution on the subdomains:

$$\begin{aligned}
\mathbf{x}_{\Omega_0} &= \mathrm{LES}(\mathbf{L}_0, \mathbf{y}_{\Omega_0} - \boldsymbol{K}_{0,1}\mathbf{x}_{\Gamma_1}),\\
\mathbf{x}_{\Omega_i} &= \mathrm{LES}(\mathbf{L}_i, \mathbf{y}_{\Omega_i} - \boldsymbol{K}_{i,i}\mathbf{x}_{\Gamma_i} - \boldsymbol{K}_{i,i+1}\mathbf{x}_{\Gamma_{i+1}}), \quad 1 \leqslant i \leqslant k-1,\\
\mathbf{x}_{\Omega_k} &= \mathrm{LES}(\mathbf{L}_k, \mathbf{y}_{\Omega_k} - \boldsymbol{K}_{k,k}\mathbf{x}_{\Gamma_k}).
\end{aligned}$$

We have to choose appropriate, probably different methods LES for all subsystems. Due to the lack of interval arithmetic equivalents of conjugate gradient or of other methods which are efficient in real (noninterval) domain decomposition, we are forced to establish and to treat the above capacitance system. Second, the subdomains Ω_i, $0 \leqslant i \leqslant k$, of the first decomposition level are horizontal strips with straight (grid) lines as artificial boundaries but which still have partially curved boundaries. In order to get the rectangles needed for IBU, we decompose on a second level each Ω_i, $1 \leqslant k \leqslant k-1$, i.e., with the exception of the boundary domains Ω_0 and Ω_k, into rectangles $\Omega_{j,i}$ and two boundary domains $\Omega_{0,i}$, $\Omega_{l(i),i}$ by introducing artificial boundaries $\Gamma_{j,i}$ on vertical grid lines:

$$\Omega_i = \bigcup_{j=0}^{l(i)} \Omega_{j,i} \cup \bigcup_{j=1}^{l(i)} \Gamma_{j,i}, \quad 1 \leqslant i \leqslant k.$$

On level 1, *all* systems with coefficient matrices $\mathbf{L}_0$, $\mathbf{L}_k$, corresponding to Ω_0, Ω_k, are treated directly by, e.g., IBG. For *all* systems with matrices $\mathbf{L}_i$, $1 \leqslant i \leqslant k-1$, corresponding to Ω_i, we apply again IDD, but now for the decomposition on level 2. In contrast to level 1, matrix blocks now correspond to vertical lines, i.e., to the first index, which we indicate by a hat:

$$\widetilde{\mathbf{L}}_i = \begin{pmatrix} \widehat{\mathbf{M}_\Omega^{(i)}} & \widehat{\boldsymbol{K}^{(i)}} \\ \widehat{\boldsymbol{H}^{(i)}} & \widehat{\mathbf{M}_\Gamma^{(i)}} \end{pmatrix} = \begin{bmatrix} \mathrm{diag}\left(\widehat{\mathbf{L}_{j,i}} \middle| 0 \leqslant j \leqslant l(i)\right) & \mathrm{bidiag}\left(\begin{matrix}\widehat{\boldsymbol{K}_{j-1,j}^{(i)}} \\ \widehat{\boldsymbol{K}_{j,j}^{(i)}}\end{matrix} \middle| 1 \leqslant j \leqslant l(i)\right) \\ \mathrm{bidiag}\left(\widehat{\boldsymbol{H}_{j-1,j}^{(i)}}, \widehat{\boldsymbol{H}_{j,j}^{(i)}} \middle| 1 \leqslant j \leqslant l(i)\right) & \mathrm{diag}\left(\widehat{\mathbf{G}_{j,i}} \middle| 1 \leqslant j \leqslant l(i)\right) \end{bmatrix}, \tag{10}$$

with block tridiagonal $\widehat{\mathbf{L}_{j,i}} = (S_{j,i}, \widehat{\mathbf{A}_{j,i}}, T_{j,i})$ and the coupling matrices $\widehat{K^{(i)}_{j-1,j}}$, $\widehat{K^{(i)}_{j,j}}$, $\widehat{H^{(i)}_{j,j}}$, $\widehat{H^{(i)}_{j-1,j}}$. Analogously, we define for any interval vector $\mathbf{z}$:

$$\tilde{\mathbf{z}} = \begin{pmatrix} \widehat{\mathbf{z}_{\Omega_i}} \\ \widehat{\mathbf{z}_{\Gamma_i}} \end{pmatrix}, \quad \widehat{\mathbf{z}_{\Omega_i}} = \left(\widehat{\mathbf{z}_{\Omega_{j,i}}}\right)_{j=0}^{l(i)}, \quad \widehat{\mathbf{z}_{\Gamma_i}} = \left(\widehat{\mathbf{z}_{\Gamma_{j,i}}}\right)_{j=1}^{l(i)}.$$

In example (3), we get, according to (9), $\widehat{\mathbf{A}_{j,i}} = (-s_n, \mathbf{d}_n, -t_n)$, $S_{j,i} = -bI$, $T_{j,i} = -gI$, tridiagonal $\mathbf{G}_{j,i} = (-s_n, \mathbf{d}_n, -t_n)$ and the coupling matrices

$$\widehat{K^{(i)}_{j-1,j}} = \begin{bmatrix} \mathbf{0} \\ -gI \end{bmatrix}, \quad \widehat{K^{(i)}_{jj}} = \begin{bmatrix} -bI \\ \mathbf{0} \end{bmatrix}, \quad \widehat{H^{(i)}_{jj}} = [-bI \quad \mathbf{0}], \quad \widehat{H^{(i)}_{j-1,j}} = [\mathbf{0} \quad -gI].$$

All systems with matrices $\widehat{\mathbf{L}_{j,i}}$, $j = 0, l(i)$, corresponding to the boundary domains $\Omega_{j,i}$, are treated by, e.g., IBG. All systems with matrices $\widehat{\mathbf{L}_{j,i}}$ on the inner rectangles $\Omega_{j,i}$, $1 \leqslant j \leqslant l(i) - 1$, $1 \leqslant i \leqslant k - 1$, are treated by IBU.

Algorithm for IDD — level 2. Apply the above algorithm on the second level on all systems $(\widetilde{\mathbf{L}_i}, \tilde{\mathbf{x}})$, $(\widetilde{\mathbf{L}_i}, \tilde{X})$, $1 \leqslant i \leqslant k - 1$, in steps 1 and 3 of level 1.

We have chosen a two-stage decomposition in order to avoid cross points. We omit a detailed formulation of the final, rather complex algorithm. The following remarks resume the most important aspects.

Remarks. (1) On the second level, the decomposition is vertically oriented. This induces a exchange of the two dimensions. The restriction (9) on $\mathbf{M}$ in example (3) then guarantees coefficient matrices $\widehat{\mathbf{L}_{j,i}}$ of the form $(-S, \mathbf{A}, -T)$ on Ω for the second level of IDD. This special form is only required for IBU on the inner rectangles, but for any system (2) to be treated by IDD.

(2) On the irregular boundary domains, we also obtain block-tridiagonal, but possible less strictly structured systems. Therefore, we must apply a more general method like IBG on these subdomains. In particular, IBG does not require $\widehat{\mathbf{L}_{j,i}} = (-S, \mathbf{A}, -T)$. This also explains why it was not necessary to specify the discretization on the irregular boundary in the introductory example. One could go even further and apply any suitable direct method LES instead of IBG, satisfying the conditions of the theorem following below, on the irregular domains if this would be required by a particular discretization.

(3) Up to now, the restriction $\widehat{\mathbf{L}_{j,i}} = (-S, \mathbf{A}, -T)$ on inner rectangles forces a dependence of the given problem on the domain decomposition which is obviously not admissible. This problem can be solved as follows in the context of a Newton-like method like (5). If a more general coefficient matrix $F = (-S_j, \mathbf{A}_j, -T_j)$ resulting from (1) with variable $a \equiv a(x, y)$ and $c \equiv c(x, y)$ (defining b and g in the discretization of (3)) is given, one first prescribes the

domain decomposition. According to the latter a matrix decomposition $F = M - N$ is defined such that $\widehat{\mathbf{L}_{j,i}} = (-S, \mathbf{A}, -T)$ on each of the inner (rectangular) subdomains $\Omega_{j,i}$.

3. Inclusion properties

In the sequel, we discuss the most important properties of IDD: IDD is applicable if all interval arithmetic operations are well defined [1]; as an example for optimal inclusions, we note a condition which is useful for an application of IDD in a nonlinear method like (5). We first define some properties to be satisfied by any method LES in this context.

(a) LES is applicable to an interval M-matrix $\mathbf{M}$.
(b) For an interval M-matrix $\mathbf{M}$ and an interval vector $\mathbf{y}$ with $i(\mathbf{y}) \geqslant 0$, LES yields an optimal inclusion:

$$\begin{aligned} \mathrm{LES}(\mathbf{M}, \mathbf{y}) &= [\mathrm{LES}(i(\mathbf{M}), i(\mathbf{y})), \mathrm{LES}(s(\mathbf{M}), s(\mathbf{y}))] \\ &= \left[(i(\mathbf{M}))^{-1} i(\mathbf{y}), (s(\mathbf{M}))^{-1} s(\mathbf{y})\right] \geqslant 0. \end{aligned} \tag{11}$$

(c) For a real M-matrix $\mathbf{M} \equiv M$ and an arbitrary interval vector $\mathbf{y}$, LES yields an optimal inclusion:

$$\mathrm{LES}(\mathbf{M}, \mathbf{y}) = [\mathrm{LES}(M, i(\mathbf{y})), \mathrm{LES}(M, s(\mathbf{y}))] = \left[M^{-1} i(\mathbf{y}), M^{-1} s(\mathbf{y})\right].$$

The proof of the subsequent theorem relies on the following lemmas.

Lemma 1. *For an M-matrix M and a permutation P, PMP^{-1} is also an M-matrix.*

Proof. M is an M-matrix $\Rightarrow \exists u > 0$: $Mu > 0 \Rightarrow Pu > 0$ and $0 < PMu = (PMP^{-1})Pu \Rightarrow PMP^{-1}$ is an M-matrix. □

Lemma 2. *Let M be an L-matrix, i.e., $m_{i,i} \geqslant 0$ and $m_{i,j} \leqslant 0$ for $i \neq j$. If there exists an M-matrix $N \leqslant M$, then M is an M-matrix* [14].

Lemma 3. *Let $M = \left(\begin{smallmatrix} A & B \\ C & D \end{smallmatrix}\right)$ be an M-matrix, A, D quadratic. Then the Schur complement $\mathbb{S}(M) = D - CA^{-1}B$ is also an M-matrix.*

Proof. If M is an M-matrix, then A and D are M-matrices and B, $C \leqslant 0$. Hence,

$$\exists u = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} > 0\text{: } Mu > 0 \Leftrightarrow Au_1 + Bu_2 > 0 \text{ and } Cu_1 + Du_2 > 0 \Leftrightarrow (D - CA^{-1}B)u_2 > 0. \quad \square$$

Lemma 4. *Assume $\mathbf{M}$ to be an interval M-matrix and* LES *to be satisfying* (11). *Then $\overline{\mathbf{M}}$ is an interval M-matrix, its Schur complement $\mathbb{S}(\overline{\mathbf{M}}) = \mathbf{M}_\Gamma - H\ \mathrm{LES}(\mathbf{M}_\Omega, K)$ is also an interval M-matrix and $\mathbb{S}(\overline{\mathbf{M}}) = [\mathbb{S}(i(\overline{\mathbf{M}})), \mathbb{S}(s(\overline{\mathbf{M}}))]$.*

Proof. If $\mathbf{M}$ is an interval M-matrix, then $\hat{\mathbf{M}}$ and $\mathbf{M}_\Omega$ are interval M-matrices, H, $K \leqslant 0$, and from (11)(c) we get $\mathrm{LES}(\mathbf{M}_\Omega, K) = [(i(\mathbf{M}_\Omega))^{-1}K, (s(\mathbf{M}_\Omega))^{-1}K] \leqslant 0$, hence

$$
\begin{aligned}
&\mathbb{S}(\overline{\mathbf{M}}) \\
&= \left[i\left(\mathbb{S}(\overline{\mathbf{M}})\right), s\left(\mathbb{S}(\overline{\mathbf{M}})\right)\right] = \left[i(\mathbf{M}_\Gamma) - s(H\ \mathrm{LES}(\mathbf{M}_\Omega, K)), s(\mathbf{M}_\Gamma) - i(H\ \mathrm{LES}(\mathbf{M}_\Omega, K))\right] \\
&= \left[i(\mathbf{M}_\Gamma) - Hi(\mathrm{LES}(\mathbf{M}_\Omega, K)), s(\mathbf{M}_\Gamma) - Hs(\mathrm{LES}(\mathbf{M}_\Omega, K))\right] \\
&= \left[i(\mathbf{M}_\Gamma) - H\left((i(\mathbf{M}_\Omega))^{-1}\right)K, s(\mathbf{M}_\Gamma) - H\left((s(\mathbf{M}_\Omega))^{-1}\right)K\right] \\
&= \left[\mathbb{S}\left(i(\overline{\mathbf{M}})\right), \mathbb{S}\left(s(\overline{\mathbf{M}})\right)\right]. \qquad \square
\end{aligned}
$$

Lemma 5. *If A is a block-tridiagonal interval M-matrix, then the interval block-tridiagonal Gauss algorithm* IBG *can be applied to* $\mathbf{A}$. *In none of the subsystems, any pivoting is necessary. If $\mathbf{A} \equiv A$ degenerates to a point matrix, then* IBG *yields optimal inclusions for an arbitrary right-hand side* $\mathbf{y}$.

Proof. The proof can be carried out using the technique of Lemma 4 and results on Gaussian elimination (e.g., from [1]). $\square$

Lemma 6. *If $\mathbf{M} = (-S, \mathbf{A}, -T)$ and $\mathbf{A} - 2(ST)^{1/2}$ are interval M-matrices, $AS = SA$, $AT = TA$, $\forall A \in \mathbf{A}$, then interval block cyclic reduction* IBU *can be applied to* $\mathbf{M}$. *If $\mathbf{M} \equiv M$ degenerates to a point matrix, then* IBU *yields optimal inclusions for an arbitrary right-hand side* $\mathbf{y}$ [12].

Theorem. *If* (1)

$$
\begin{aligned}
&\forall i \in \{1, \ldots, n_y\}, \forall j \in \{j_f(i) + 1, \ldots, j_l(i) - 1\}: \quad (j, i) \in \Omega_h, \\
&\forall j \in \{1, \ldots, m_x\}, \forall i \in \{i_f(j) + 1, \ldots, i_l(j) - 1\}: \quad (j, i) \in \Omega_h,
\end{aligned}
$$

where $(j_f(i), i)$ and $(j_l(i), i)$ $((j, i_f(j))$ and $(j, i_l(j)))$ denote the boundary points on the ith grid row (jth column);

(2) $\mathbf{M}$ *is an interval M-matrix;*

(3) $\forall i \in \{1, \ldots, k-1\}, \forall j \in \{1, \ldots, l(i) - 1\}$: $\mathbf{A}_{j,i} - 2\sqrt{S_{j,i}T_{j,i}}$ *are interval M-matrices and* $\forall A \in \mathbf{A}_{j,i}$: $AS_{j,i} = S_{j,i}A$, $AT_{j,i} = T_{j,i}A$;

(4) $m_{j,i} = (2^{p^{j,i}} + 1)2^{q^{j,i}} - 1$, $p^{j,i} \in \mathbb{N}$, $1 \leqslant j \leqslant l(i)$, $1 \leqslant i \leqslant k$, *where $m_{j,i}$ denotes the number of grid lines in $\Omega_{j,i}$;*

then (1) $\overline{\mathbf{M}}$, $\mathbf{L}_i$, $\widetilde{\mathbf{L}_i}$, $\widehat{\mathbf{L}_{j,i}}$, $\mathbf{C}_\Gamma$, $\widehat{\mathbf{C}_{\Gamma_i}}$ *are interval M-matrices;*

(2) *the following interval arithmetic methods* LES *are applicable:*

- *block tridiagonal Gaussian elimination* IBG *to* $\mathbf{L}_0$, $\mathbf{L}_k$, $\widehat{\mathbf{L}_{0,i}}$, $\widehat{\mathbf{L}_{l(i),i}}$, $\widehat{\mathbf{C}_{\Gamma_i}}$, $1 \leqslant i \leqslant k-1$, $\mathbf{C}_\Gamma$;
- *stabilized block cyclic reduction* IBU *to* $\widehat{\mathbf{L}_{j,i}}$, $1 \leqslant j \leqslant l(i) - 1$, $1 \leqslant i \leqslant k-1$;
- *domain decomposition* IDD *to* $\overline{\mathbf{M}}$;

(3) *if the respective coefficient matrix degenerates into a point matrix*, IBG, IBU, IDD *yield optimal inclusions for arbitrary right-hand sides* **y** *in* (2), *in particular*:

$$\mathrm{IDD}(\overline{\boldsymbol{M}}, \bar{\mathbf{y}}) = \left[\mathrm{IDD}(\overline{\boldsymbol{M}}, i(\mathbf{y})), \mathrm{IDD}(\overline{\boldsymbol{M}}, s(\bar{\mathbf{y}}))\right].$$

Proof. The technical condition (1) ensures contiguous index sets in both dimensions which are necessary for a reasonable index management. Condition (4) indicates the admissible number of blocks in $\widehat{\mathbf{L}_{j,i}}$ for the application of IBU which guarantees optimal inclusions (discussed in [13]). We outline the proof of the remaining assertions. For assertion (1), we note that with **M**, its permutation $\overline{\mathbf{M}}$ is also an interval M-matrix (Lemmas 1 and 2) as well as all submatrices $\mathbf{L}_i$. Because of $\mathrm{LES}(\mathbf{M}_\Omega, \cdot) = (\mathrm{LES}(\mathbf{L}_i, \cdot) \mid 0 \leqslant i \leqslant k) = (\mathrm{IBG}(\mathbf{L}_0, \cdot),\ (\mathrm{IDD}(\mathbf{L}_i, \cdot) \mid 1 \leqslant i \leqslant k-1),\ \mathrm{IBG}(\mathbf{L}_k, \cdot))$ and with Lemmas 3 and 5 it suffices to show (11)(a) and (11)(b) for IDD on the second level. Under this assumption, Lemma 3 implies that $\mathbf{C}_\Gamma$ is an interval M-matrix.

With $\mathbf{L}_i$, its permutations $\widetilde{\mathbf{L}_i}$ and $\widehat{\mathbf{L}_{j,i}}$ are also interval M-matrices. We repeat the above proof for the second level with LES = IBG for $j = 0$, $l(i)$, $1 \leqslant i \leqslant k-1$, and LES = IBU for $1 \leqslant j \leqslant l(i)-1$, $1 \leqslant i \leqslant k-1$. According to Lemma 6 and to condition (3), IBU satisfies (11) for $\widehat{\mathbf{L}_{j,i}}$, hence the proof of assertion (1) is complete. For the proof of assertion (2), we repeatedly apply (11)(a) for LES = IBG, IBU (under the additional condition (3)), IDD. For assertion (3), we apply (11)(c) for LES = IBG, IBU, IDD and we have to prove that the lower (upper) bound of $\mathrm{IDD}(\boldsymbol{M}, \mathbf{y})$ only depends on the lower (upper) bound of **y**. This follows from the relations $-a\mathbf{x} = (-a)\mathbf{x}$ for an arbitrary interval vector **x** and $a \in \mathbb{R}$, and $a\mathbf{x} = [ai(\mathbf{x}), as(\mathbf{x})]$ if $a \geqslant 0$. □

Remark. Condition (1) for Ω is weaker than convexity. It only prescribes that all grid points between two boundary points on the same vertical or horizontal grid line belong to ω_h. Domains which do not satisfy this condition can only be treated by an additional domain decomposition level.

Remark. The commutativity relation in condition (3) is essential for the derivation of IBU. It cannot be satisfied by interval matrices $\boldsymbol{S} \neq \mathbf{S}$ or $\boldsymbol{T} \neq \mathbf{T}$. On the other hand, the application of Newton-like interval methods (5) to systems (2) resulting from discretizations of almost linear boundary value problems like (1) or (3) with constant a and c leads to the application of IBU to interval systems with coefficient matrices $(-\boldsymbol{S}, \mathbf{A}, -\boldsymbol{T})$, $\mathbf{A} = \boldsymbol{A}' + \mathbf{D}$, where the diagonal interval matrix **D** results from an interval arithmetic evaluation [1] of the derivative $\phi'(\boldsymbol{u})$ of the (diagonal) nonlinear part of (2) and where the point matrix $(-\boldsymbol{S}, \boldsymbol{A}', -\boldsymbol{T})$ is defined from a suitable splitting $\boldsymbol{F} = \boldsymbol{M} - \boldsymbol{N}$ of the linear part of (2) (see, e.g., [10]).

4. Complexity

The *arithmetic complexity* of IDD mostly depends on the number and size of the domains. On the boundary domains, it is rather determined by the number of points than by the particular shape. We illustrate the arithmetic complexity for an example of type (3) on a rectangle with $N = m_x * n_y$ grid points, $k+1$ domains (horizontal strips) Ω_i, $0 \leqslant i \leqslant k$, on the first level. The boundary domains Ω_0, Ω_k each have r grid rows. Each of the Ω_i, $1 \leqslant i \leqslant k-1$, is divided into $l+1$, $l \equiv l(i)$, domains (vertical strips) on the second level with s grid columns in

all boundary domains $\Omega_{0,i}$, $\Omega_{l,i}$. We only note the high-order terms in the expressions for the number of *interval arithmetic operations*. In view of the remarks in the Introduction on the lack of an efficient interval method for problems on nonrectangular domains, we compare IDD only with block-tridiagonal Gaussian elimination, for which we get

IBG: $(n_y - 1)\frac{2}{3}m_x^4$.

While IBG does not depend on k, l, r and s, this dependence is crucial for IDD:

boundary domains first level:	$\mathbf{L}_0, \mathbf{L}_k$,	$(r-1)\frac{4}{3}m_x^4$,
boundary domains second level:	$\mathbf{L}_{0,i}, \mathbf{L}_{l(i),i}$,	$(s-1)\frac{4}{3}\frac{n_y^4}{k^3} + s\frac{8}{3}m_x\frac{n_y^3}{k^2}$,
rectangular domains second level:	$\mathbf{L}_{j,i}$,	$2ln_y m_x^2\left\{33\log_2\left(\frac{m_x}{l+1}\right)+95\right\}$,
capacitance system first level:	$\mathbf{C}_\Gamma$,	$(k-1)\frac{2}{3}m_x^4$,
capacitance systems second level:	$\mathbf{C}_{\Gamma_i}$,	$(l-1)\frac{2}{3}\frac{n_y^4}{k^3} + l\frac{4}{3}\frac{n_y^3}{k^2}$.

The obvious strategy is to reduce first r and s, i.e., to reduce the boundary domains on both levels by filling Ω optimally with rectangles. For, relative to n_y, large m_x, an increasing number k of subdomains may lead to a dominance of the main capacitance system, reducing thus the efficiency in comparison to IBG. For, relative to m_x, large n_y, an increasing k reduces the influence of the capacitance systems on the second level without affecting significantly the complexity of the main capacitance system. The solution on the inner rectangles by IBU does not depend on k, but on l. An increasing l increases in any case the total arithmetic complexity on both the inner rectangles and the capacitance systems on the second level. $l > 2$ can, however, be necessary for the reduction of the boundary domains on the second level under the restrictions for IBU of condition (3) from the Theorem.

With this strategy, the arithmetic complexity can be significantly reduced when compared to methods like IBG, particularly by the application of IBU on the inner rectangles. IDD cannot, however, be considered as a "fast solver" because of the complexity of the capacitance systems and the systems on the boundary domains. Methods like the interval Jacobi or Gauss–Seidel method with a small complexity per iteration step show the same extremely small convergence speed known from the corresponding point methods. They are not competitive at all with regard to arithmetic, i.e., time complexity, in particular as part of nonlinear methods, and not considered in this context.

The *memory complexity* of direct methods like IDD or IBG is obviously significantly higher than that of iterative methods ($O(N) = O(n_y * m_x)$ for Jacobi or Gauss–Seidel). The storage requirement of IDD strongly depends on the domain decomposition. The direct comparison yields

IBG: Gauss block-tridiagonal:	$6 \times n_y * m_x * m_x$,
IDD: interval arithmetic domain decomposition:	$\ll 2 \times n_y * m_x * m_x$.

5. Numerical results

Most of the following tests have been carried out on a CRAY Y-MP2E/164, except some tests on a workstation IBM RS/6000-550. For the comparison of computing times and precision, note that, due to the lack of an interval arithmetic on most machines, we use a simulation which is based on the available floating-point hardware. Interval operations are simulated by a subsequent multiplication by or addition of suitable constants such that computed intervals include all rounding errors occurring in the computation of their bounds [1]. As an example we choose the following system of equations:

$$([\boldsymbol{M}, \boldsymbol{M}], [\boldsymbol{y}, \boldsymbol{y}]), \qquad N = m_x * n_y, \qquad X_{i,j} = [j+i, j+i],$$

$$\boldsymbol{y} := [\boldsymbol{M} * \boldsymbol{x}, \boldsymbol{M} * \boldsymbol{x}], \qquad \boldsymbol{M} = \left(-s_j, \left(-b, d_{j,i}, -g\right), -t_j\right),$$

$$d_{j,i} = 8.3 + 2j + 0.01i, \qquad b = 3.1, \qquad g = 4.1, \qquad s_j = j, \qquad t_j = j + 1.$$

The computation times are not affected by the fact that this is a real point system. On the other hand, we know that the exact solution must be a real vector of width 0. The inspection of the computed width yields bounds for the enclosed rounding errors and a possible overestimate of the latter by the simulation.

The subsequent examples illustrate the timing behaviour in terms of CPU-seconds. They confirm the conclusions drawn from the above rough estimates of the interval arithmetic complexity. Note that IDD can be further improved, particularly by an optimization of IBU with respect to a specific computing environment. We have chosen rather small examples in order to show that IDD can yield improvements even under nonoptimal conditions. In particular, the inner rectangles (225 to 3969 points) are by far too small to enable IBU to really profit from its property to be a "fast solver".

General example

We choose a nonrectangular domain Ω discretized by a grid Ω_h with maximally $m_x = 131$ and $n_y = 149$ points in the x- and y-dimension, $k + 1 = 6$ domains on the first level, $r = 5$ grid lines each with between 1 and 10 grid points in both Ω_0 and Ω_k, between $l(i) + 1 = 3$ and $l(i) + 1 = 6$ subdomains on the second level and between $s \equiv s(i) = 1$ and $s \equiv s(i) = 17$ points per horizontal grid line on the boundary domains $\Omega_{0,i}$ and $\Omega_{l(i),i}$. Although the domain decomposition is not at all optimal because we define more subdomains and larger boundary domains than necessary on both levels we still observe a clear improvement:

IDD: domain decomposition: 95 sec.,
IBG: block-tridiagonal Gauss: 151 sec.

Influence of an enlarged curved boundary

We simulate the effect of larger boundary domains by a rectangle with $63 + 2(r + 1) \times 63 + 2(s + 1)$ grid points with one inner rectangle of 63×63 grid points ($k = l = 2$) and rectangular boundary domains Ω_0 and Ω_2 with $r \times 63 + 2(s + 1)$ grid points and $\Omega_{0,0}$ and $\Omega_{0,2}$ with $63 \times s$ grid points. Due to the use of IBU on the boundary domains, the CPU-time significantly

increases with increasing r or s, as expected. In this example the effect of an increasing s, i.e., m_x, is more important than an increasing r, i.e., n_y:

r	s	m_x	n_y	sec.
1	1	67	67	12.5
10	1	85	67	28.4
1	10	67	85	52.8
10	10	85	85	83.4

Influence of a larger number of subdomains on the second level

We choose a rectangle with $m_x \times n_y$ points, $m_x = 259$, $n_y = 35$. The boundary domains are reduced each to one grid line ($r = s = 1$). As predicted, it should be avoided to increase l. In this case, we get an increasing number of subdomains of reduced size on the second level. This influences the applications of IBU which must also be used to compute more complex capacitance matrices on the second level:

$$\begin{cases} 93 \text{ sec}, & \text{if } l = 2, \\ 102 \text{ sec}, & \text{if } l = 3, \\ 189 \text{ sec}, & \text{if } l = 5, \\ 266 \text{ sec}, & \text{if } l = 9, \end{cases} \qquad \text{IBG: block tridiagonal Gauss: } 289 \text{ sec.}$$

Influence of a larger number of subdomains on the first level

We "rotate" the rectangle of the last example to $m_x = 35$, $n_y = 259$. Increasing the number k of subdomains on the first level does not affect the arithmetic complexity of IBU on the inner rectangles. The influence of the capacitance systems on the second level is reduced while that of the main capacitance system is increased. The optimal value depends on the relation of m_x and n_y. The qualitative difference between the results on a CRAY and on an IBM RISC workstation for different k is due to a machine-dependent code optimization. In this extreme example ($m_x \ll n_y$), the advantage of IDD over IBG is minimal as the arithmetic complexity is dominated by m_x. The main capacitance system has a complexity of the same order as IBG and the inner rectangles are too small for IBU:

$$\text{IDD:} \begin{cases} 38.8 \text{ sec.}, & 251.2 \text{ sec.}, & \text{if } k = 2, \\ 19.4 \text{ sec.}, & 115{,}2 \text{ sec.}, & \text{if } k = 3, \\ 15.0 \text{ sec.}, & 74.6 \text{ sec.}, & \text{if } k = 5, \\ 16.0 \text{ sec.}, & 59.2 \text{ sec.}, & \text{if } k = 9, \end{cases}$$

for CRAY Y-MP, respectively IBM RS/6000-550,

IBG: block-tridiagonal Gauss: 17.5 sec (CRAY Y-MP).

The precision of IDD can be indicated by means of the relative interval width:

$$\mathrm{rd}(\mathbf{x}) := \max\left\{ \frac{d(X_{j,i})}{\min\left(|\inf(X_{j,i})|, |\sup(X_{j,i})|\right)} \,\middle|\, 1 \leqslant i \leqslant n_y,\ j_f(i) + 1 \leqslant j \leqslant j_l(i) - 1 \right\}.$$

In this context we observe $10^{-9} \leqslant \mathrm{rd}(\mathbf{x}) \leqslant 10^{-13}$, depending on the example and the machine. The precision of the results by the IEEE-like arithmetic of the IBM (15 decimals) is better by 2–4 decimals in most cases than the CRAY arithmetic (14 decimals). This is essentially due to the necessary simulation of a missing rounding bit in additions on the CRAY.

6. Conclusion

IDD must be considered as a first example of an interval domain decomposition method which has to be subject of improvements under various aspects. IDD provides an efficient solver for interval systems of equations resulting from discretized elliptic partial boundary value problems on nonrectangular domains. Due to the specific nature of linear systems with interval coefficients, the importance of IDD consists in its application in Newton-like methods for nonlinear problems.

Acknowledgements

I want to express my gratitude to the referees for their helpful comments and suggestions.

References

[1] G. Alefeld and J. Herzberger, *Introduction to Interval Computations* (Academic Press, New York, 1983).
[2] W. Barth and E. Nuding, Optimale Lösungen von Intervallgleichungssystemen, *Computing* **12** (1974) 117–124.
[3] T.F. Chan, Analysis of preconditioners for domain decomposition, *SIAM J. Numer. Anal.* **24** (1987) 382–390.
[4] H. Cornelius, The acceleration of an interval arithmetic iterative method, *SIAM J. Numer. Anal.* **20** (1983) 1010–1022.
[5] A. Frommer and G. Mayer, Parallel interval multisplittings, *Numer. Math.* **56** (1989) 255–267.
[6] G.H. Golub and D. Mayers, The use of pre-conditioning over irregular regions, in: R. Vichnevetsky and R.S. Stepleman, Eds., *Advances in Computer Methods for Partial Differential Equations* (IMACS, New Brunswick, NJ, 1984) 360–368.
[7] R.E. Moore, *Interval Analysis* (Prentice-Hall, Englewood Cliffs, NJ, 1966).
[8] A. Neumaier, New techniques for the analysis of linear interval equations, *Linear Algebra Appl.* **58** (1984) 273–325.
[9] J.M. Ortega and W.C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables* (Academic Press, New York, 1970).
[10] H. Schwandt, The solution of nonlinear elliptic Dirichlet problems on rectangles by almost globally convergent interval methods, *SIAM J. Sci. Statist. Comput.* **6** (1985) 617–638.
[11] H. Schwandt, Krawczyk-like algorithms for the solution of systems of nonlinear equations, *SIAM J. Numer. Anal.* **22** (1985) 792–810.
[12] H. Schwandt, Interval arithmetic methods for systems of nonlinear equations arising from discretizations of quasilinear elliptic and parabolic partial differential equations, *Appl. Numer. Math.* **3** (3) (1987) 257–287.
[13] H. Schwandt, The interval Buneman algorithm for arbitrary matrix dimension, *Comput. Suppl.* **9** (1993) 213–231.
[14] R.S. Varga, *Matrix Iterative Analysis* (Prentice-Hall, Englewood Cliffs, NJ, 1962).