Acceleration of computing domain solutions by the IFPIES

Andrzej Kużelewski[0000−0003−2247−2714] and Eugeniusz Zieniuk[0000−0002−6395−5096]

Faculty of Computer Science, University of Bialystok
Ciolkowskiego 1M, 15-245 Bialystok, Poland
{a.kuzelewski,e.zieniuk}@uwb.edu.pl

Abstract. The paper presents the interval fast parametric integral equations system (IFPIES) applied for searching domain solutions of potential 2D boundary value problems (BVPs). Previously, the IFPIES has been successfully applied in modelling uncertainly defined 2D potential BVPs and finding solutions on the boundary. The combination of the modified fast multipole technique and modified directed interval arithmetic with the PIES reduced the computational time and RAM usage in the interval PIES. To find solutions in the domain, similar techniques are applied. The method is demonstrated with the solution in the domain of uncertainly defined BVPs.

Keywords: Interval Fast PIES · uncertainty · BVP · Fast Multipole Method.

1 Introduction

The interval fast parametric integral equations system (IFPIES) [1] belongs to a family of computational methods that are derived from the classical parametric integral equation system (PIES [2]). All of these methods, beginning with PIES, are in a constant state of development and are employed to solve a variety of two- and three-dimensional boundary value problems (BVPs) described by different types of differential equation (e.g. Laplace, Helmholtz or Navier–Lame). Also, all methods in that family offers the same remarkable advantages compared to the FEM [3–5], the BEM [6–8] (or the still being developed the FEM-BEM hybrids [9], isogeometric analysis (IgA) [10] and the Virtual Element Method (VEM) [11]) such as the lack of discretization of the problem's boundary and domain (sometimes called as mesh-free) or possibility of improving the accuracy of solutions with only a slight modification of the input data (precisely, the number of collocation points). They use parametric functions widely used in computer graphics and include, among others, Bézier and B-spline curves, Coons and Bézier surface patches. Due to the direct inclusion of these functions into the mathematical formalism of the PIES family methods, the boundary is defined by a small number of interval control points. Also, the use of parametric

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

functions reduces the dimensionality of the problem by 1. Improving the accuracy of solutions requires only a slight adjustment to the number of collocation points, unlike in the BEM and the FEM, where the discretization process must be repeated. Such an approach differs from a finite difference scheme applied in a wide family of mesh-free methods (such as particle methods [12], Galerkin methods [13] or cloud methods [14]).

Our previous studies on the PIES family methods have focused on various aspects, including the use of NURBS curves [15] or inverse problems [16]. All these studies have confirmed their higher accuracy compared to BEM or FEM. However, in early implementations of classical and interval PIES, time-consuming calculations and RAM utilization increase with the problem size squared. Similar to the BEM, the PIES uses dense non-symmetric coefficient matrices. Therefore, to compute the PIES matrices, we need $O(N^2)$ operations and another $O(N^3)$ operations to solve the obtained system using direct solvers (where N is the number of equations of the algebraic equations system). To eliminate this main disadvantage, the authors of this paper previously applied the fast multipole method (FMM) [17–19], obtaining the fast PIES (FPIES) [20] and the IFPIES [1]. Application of the FMM and modified binary tree [21] results in obtaining the system of algebraic equations $\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{b}$ which is generated implicitly in the IFPIES and is solved by the iterative GMRES solver [22].

It means that only the result of the multiplication of the matrix $\boldsymbol{A}$ by the vector of unknowns $\boldsymbol{x}$ is stored in memory, in contrast to the conventional PIES, which has to store the full dense non-symmetric matrix $\boldsymbol{A}$ and the vector $\boldsymbol{b}$ in RAM. The application of the FMM allows for a significant reduction in computational time to $O(NlogN)$ and a decrease in RAM utilization to $O(N)$. Previous studies on the IFPIES have focused on efficient and accurate solutions to problems described by single- and multi-connected domains [20, 23], as well as on analyses of key parameters [24, 25]. Also, a comparison with a competing method, such as the fast multipole BEM (FMBEM) described in [26].

Up to now, the authors of this paper have used the IFPIES to find solutions of uncertainly defined BVPs on the boundary only. Solutions in the domain can be obtained using the classical technique applied in the conventional PIES [2]. The main goal of this paper is to present the IFPIES applied to find numerical solutions for uncertainly defined 2D potential BVPs in the domain using the fast multipole technique. The efficiency and accuracy of the IFPIES are tested on a potential problem described by linear boundary segments.

2 The interval fast parametric integral equations system

The IFPIES is a direct successor of the IPIES [27] and the FPIES [20] which are based on original PIES [2]. The original PIES was obtained as the result of the analytical modification of the boundary integral equations (BIE) by direct inclusion parametric curves describing the shape of the boundary of the problem into the PIES kernels. The directed interval arithmetic [28] and the method

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

of mapping arithmetic operators to the positive semi-axis [29] applied to the original PIES results in obtaining the following form of the IPIES:

$$\begin{gathered}\frac{1}{2}\boldsymbol{u}_l(\widehat{s}) = \sum_{j=1}^{n} \int_{s_{j-1}}^{s_j} \left\{ \widehat{\boldsymbol{U}}^*_{lj}(\widehat{s}, s)\boldsymbol{p}_j(s) - \widehat{\boldsymbol{P}}^*_{lj}(\widehat{s}, s)\boldsymbol{u}_j(s) \right\} \boldsymbol{J}_j(s)ds, \\ l = 1, 2, ..., n, \;\; s_{l-1} \leq \widehat{s} \leq s_l, \;\; s_{j-1} \leq s \leq s_j,\end{gathered} \tag{1}$$

where n is the number of parametric segments that creates boundary of domain in 2D (mapped into parametric reference system), $\boldsymbol{J}_j(s)$ is the interval Jacobian, s_{j-1} correspond to the beginning of j-th segment, while s_j to its end. The interval kernels $\widehat{\boldsymbol{U}}^*_{lj}(\widehat{s}, s)$ and $\widehat{\boldsymbol{P}}^*_{lj}(\widehat{s}, s)$ have the following form:

$$\begin{aligned}\widehat{\boldsymbol{U}}^*_{lj}(\widehat{s}, s) &= \frac{1}{2\pi} \ln \frac{1}{\sqrt{\left(\boldsymbol{S}^{(1)}\right)^2 + \left(\boldsymbol{S}^{(2)}\right)^2}}, \\ \widehat{\boldsymbol{P}}^*_{lj}(\widehat{s}, s) &= \frac{1}{2\pi} \frac{\boldsymbol{S}^{(1)} \cdot n_j^{(1)}(s) + \boldsymbol{S}^{(2)} \cdot n_j^{(2)}(s)}{\left(\boldsymbol{S}^{(1)}\right)^2 + \left(\boldsymbol{S}^{(2)}\right)^2},\end{aligned} \tag{2}$$

where $\boldsymbol{S}^{(1)} = \boldsymbol{S}_l^{(1)}(\widehat{s}) - \boldsymbol{S}_j^{(1)}(s)$ and $\boldsymbol{S}^{(2)} = \boldsymbol{S}_l^{(2)}(\widehat{s}) - \boldsymbol{S}_j^{(2)}(s)$. Expressions $\boldsymbol{S}_k(s_n) = \{\boldsymbol{S}_k^{(1)}(s_n), \boldsymbol{S}_k^{(2)}(s_n)\}$ $(k = \{j, l\}$ $s_n = \{\widehat{s}, s\})$ are interval Bézier curves of the third degree (curvilinear segments) or the first degree (linear segments) mapped into parametric reference system.

Boundary functions $\boldsymbol{u}_j(s)$ and $\boldsymbol{p}_j(s)$ in (1) are approximated by the following series:

$$\begin{aligned}\boldsymbol{u}_j(s) &= \sum_{k=0}^{N} \boldsymbol{u}_j^{(k)} \boldsymbol{L}_j^{(k)}(s), \\ \boldsymbol{p}_j(s) &= \sum_{k=0}^{N} \boldsymbol{p}_j^{(k)} \boldsymbol{L}_j^{(k)}(s),\end{aligned} \tag{3}$$

where $\boldsymbol{u}_j^{(k)}$ and $\boldsymbol{p}_j^{(k)}$ are unknown or given interval values of boundary functions in defined points of the segment j, N - is the number of terms in the approximation series (3), which approximates boundary functions on the segment j and $\boldsymbol{L}_j^{(k)}(s)$ - the base functions (Lagrange polynomials) on segment j.

Similarly to the FPIES [20], the IFPIES required modification of the IPIES kernels (to allow for the Taylor series approximation used by the FMM), and modification of the tree used by the FMM (to include the way of defining the boundary in the PIES). Modified IPIES kernels have the following form:

$$\begin{aligned}\widehat{\boldsymbol{U}}^{*(c)}_{lj}(\widehat{\boldsymbol{\tau}}, \boldsymbol{\tau}) &= -\frac{1}{2\pi} \ln\left(\widehat{\boldsymbol{\tau}} - \boldsymbol{\tau}\right), \\ \widehat{\boldsymbol{P}}^{*(c)}_{lj}(\widehat{\boldsymbol{\tau}}, \boldsymbol{\tau}) &= \frac{1}{2\pi} \frac{\boldsymbol{n}_j^{(c)}}{\widehat{\boldsymbol{\tau}} - \boldsymbol{\tau}},\end{aligned} \tag{4}$$

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

where $\boldsymbol{n}_j^{(c)}$ is the complex notation of interval normal vector to the curve, which creates segment j. Expressions $\widehat{\boldsymbol{\tau}}, \boldsymbol{\tau}$ are the complex interval version of parametric functions describing the boundary: $\widehat{\boldsymbol{\tau}} = \boldsymbol{S}_l^{(c)}(\widehat{s}) = \boldsymbol{S}_l^{(1)}(\widehat{s}) + i\boldsymbol{S}_l^{(2)}(\widehat{s})$, $\boldsymbol{\tau} = \boldsymbol{S}_j^{(c)}(s) = \boldsymbol{S}_j^{(1)}(s) + i\boldsymbol{S}_j^{(2)}(s)$.

Modification of the FMM tree take into account that the boundary of the 2D problem is mapped into a 1D parametric reference system. Therefore, the first and the last segment in the parametric reference system should be considered as connected to each other. It required to close the tree levels as presented in Fig. 1 (more precisely described in [21]). The interval points $\boldsymbol{\tau}_c'$ and $\boldsymbol{\tau}_{el}'$ are obtained during applying the FMM technique (tracing the tree structure up and down).

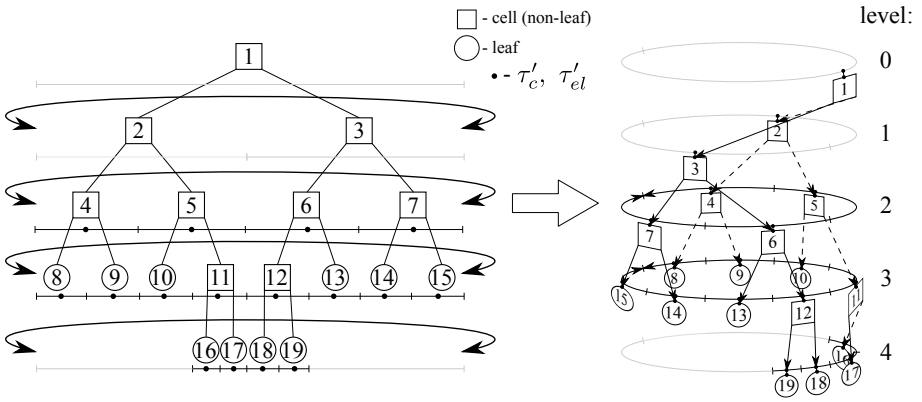

Fig. 1. The example of modified binary tree in the fast PIES for 2D problem

At last, the following formula presents the final form of the IFPIES for solving uncertainly defined BVPs on the boundary [1]:

$$\frac{1}{2}\boldsymbol{u}_l(\widehat{s}) = \sum_{j=1}^{n} \mathbb{R}\left\{ \frac{1}{2\pi} \sum_{l=0}^{N_T} (-1)^l \cdot \left[\sum_{k=0}^{N_T} \sum_{m=l}^{N_T} \frac{(k+m-1)! \cdot \boldsymbol{M}_k(\boldsymbol{\tau}_c')}{(\boldsymbol{\tau}_{el} - \boldsymbol{\tau}_c)^{k+m}} \right.\right.$$
$$\left.\left. \cdot \frac{(\boldsymbol{\tau}_{el}' - \boldsymbol{\tau}_{el})^{m-l}}{(m-l)!} \right] \cdot \frac{(\widehat{\boldsymbol{\tau}} - \boldsymbol{\tau}_{el}')^l}{l!} \right\} - \sum_{j=1}^{n} \mathbb{R}\left\{ \frac{1}{2\pi} \sum_{l=0}^{N_T} (-1)^l \cdot \right. \tag{5}$$
$$\left. \left[\sum_{k=1}^{N_T} \sum_{m=l}^{N_T} \frac{(k+m-1)! \cdot \boldsymbol{N}_k(\boldsymbol{\tau}_c')}{(\boldsymbol{\tau}_{el} - \boldsymbol{\tau}_c)^{k+m}} \cdot \frac{(\boldsymbol{\tau}_{el}' - \boldsymbol{\tau}_{el})^{m-l}}{(m-l)!} \right] \cdot \frac{(\widehat{\boldsymbol{\tau}} - \boldsymbol{\tau}_{el}')^l}{l!} \right\},$$

where interval moments $\boldsymbol{M}_k(\boldsymbol{\tau}_c')$ and $\boldsymbol{N}_k(\boldsymbol{\tau}_c')$ are computed twice only and have the form:

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

$$\begin{aligned} \boldsymbol{M}_k(\boldsymbol{\tau}_c') &= \sum_{l=0}^{k} \frac{(\boldsymbol{\tau}_c - \boldsymbol{\tau}_c')^{k-l}}{(k-l)!} \boldsymbol{M}_l(\boldsymbol{\tau}_c), \\ \boldsymbol{N}_k(\boldsymbol{\tau}_c') &= \sum_{l=1}^{k} \frac{(\boldsymbol{\tau}_c - \boldsymbol{\tau}_c')^{k-l}}{(k-l)!} \boldsymbol{N}_l(\tau_c), \end{aligned} \tag{6}$$

where:

$$\begin{aligned} \boldsymbol{M}_l(\boldsymbol{\tau}_c) &= \int_{\boldsymbol{s}_{j-1}}^{\boldsymbol{s}_j} \frac{(\boldsymbol{\tau} - \boldsymbol{\tau}_c)^l}{l!} \boldsymbol{p}_j(s)\, \boldsymbol{J}_j(s) ds, \\ \boldsymbol{N}_l(\boldsymbol{\tau}_c) &= \int_{\boldsymbol{s}_{j-1}}^{\boldsymbol{s}_j} \frac{(\boldsymbol{\tau} - \boldsymbol{\tau}_c)^{l-1}}{(l-1)!} \boldsymbol{n}^{(c)} \boldsymbol{u}_j(s)\, \boldsymbol{J}_j(s)\, ds, \end{aligned}$$

and N_T is the number of terms in Taylor series, $\boldsymbol{J}_j(s)$ is the interval Jacobian, $\boldsymbol{s}_{j-1}$ correspond to the beginning of j-th segment, while $\boldsymbol{s}_j$ to its end, $\boldsymbol{u}_j(s)$ and $\boldsymbol{p}_j(s)$ are interval boundary functions (3), $\boldsymbol{n}^{(c)} = \boldsymbol{n}^{(1)} + i\boldsymbol{n}^{(2)}$ is the complex interval notation of normal vector to the curve, which creates segment j, $\mathbb{R}$ is the real part of complex number and superscript (c) means the complex variable, while (1) and (2) - coordinates in the complex reference system.

3 Interval solutions in the domain

Solving the IFPIES (5) results in solutions on the boundary only. To find solutions in the domain, a modification of the IPIES integral identity is required. The interval identity uses the solutions of the IPIES, and it has the following form [27]:

$$\boldsymbol{u}(\boldsymbol{x}) = \sum_{j=1}^{n} \int_{\boldsymbol{s}_{j-1}}^{\boldsymbol{s}_j} \left\{ \widehat{\boldsymbol{U}}_j^*(\boldsymbol{x}, s) \boldsymbol{p}_j(s) - \widehat{\boldsymbol{P}}_j^*(\boldsymbol{x}, s) \boldsymbol{u}_j(s) \right\} \boldsymbol{J}_j(s) ds. \tag{7}$$

Integrands $\widehat{\boldsymbol{U}}_j^*(\boldsymbol{x}, s)$ and $\widehat{\boldsymbol{P}}_j^*(\boldsymbol{x}, s)$ are presented in the following form:

$$\begin{aligned} \widehat{\boldsymbol{U}}_j^*(\boldsymbol{x}, s) &= \frac{1}{2\pi} \ln \frac{1}{\sqrt{\boldsymbol{r}_1{}^2 + \boldsymbol{r}_2{}^2}} \\ \widehat{\boldsymbol{P}}_j^*(\boldsymbol{x}, s) &= \frac{1}{2\pi} \frac{\boldsymbol{r}_1 \boldsymbol{n}^{(1)}(s) + \boldsymbol{r}_2 \boldsymbol{n}^{(2)}(s)}{\boldsymbol{r}_1^2 + \boldsymbol{r}_2^2} \end{aligned} \tag{8}$$

where $\boldsymbol{r}_1 = \boldsymbol{x}^{(1)} - \boldsymbol{S}_j^{(1)}(s)$ and $\boldsymbol{r}_2 = \boldsymbol{x}^{(2)} - \boldsymbol{S}_j^{(2)}(s)$, $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ are the coordinates of the interval solution point in the domain, $\boldsymbol{n}^{(1)}$ and $\boldsymbol{n}^{(2)}$ are components of interval vector normal to the curve. The shape of the boundary is included in (8) by $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$, which contain interval parametric curves.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

Similarly to the concept presented in [31], we obtained the complex interval integrands $\widehat{\boldsymbol{U}}_j^{*(c)}(\boldsymbol{x}^{(c)}, \boldsymbol{\tau})$ and $\widehat{\boldsymbol{P}}_j^{*(c)}(\boldsymbol{x}^{(c)}, \boldsymbol{\tau})$ in the following form:

$$\begin{aligned} \widehat{\boldsymbol{U}}_j^{*(c)}(\boldsymbol{x}^{(c)}, \boldsymbol{\tau}) &= -\frac{1}{2\pi} \ln\left(\boldsymbol{x}^{(c)} - \boldsymbol{\tau}\right), \\ \widehat{\boldsymbol{P}}_j^{*(c)}(\boldsymbol{x}^{(c)}, \boldsymbol{\tau}) &= \frac{1}{2\pi} \frac{\boldsymbol{n}^{(c)}(s)}{\boldsymbol{x}^{(c)} - \boldsymbol{\tau}}, \end{aligned} \tag{9}$$

where $\boldsymbol{x}^{(c)} = \boldsymbol{x}^{(1)} + i\boldsymbol{x}^{(2)}$ and:

$$\widehat{\boldsymbol{U}}_j^{*}(\boldsymbol{x}, s) = \mathbb{R}\{\widehat{\boldsymbol{U}}_j^{*(c)}(\boldsymbol{x}^{(c)}, \boldsymbol{\tau})\},$$

$$\widehat{\boldsymbol{P}}_j^{*}(\boldsymbol{x}, s) = \mathbb{R}\{\widehat{\boldsymbol{P}}_j^{*(c)}(\boldsymbol{x}^{(c)}, \boldsymbol{\tau})\}.$$

We can use the same fast multipole tree and moments $\boldsymbol{M}_k(\boldsymbol{\tau}_c)$ and $\boldsymbol{N}_k(\boldsymbol{\tau}_c)$ calculated previously during solving the IFPIES. However, to find solutions in the domain, we are moving only to the highest level of the tree. Substituting kernels $\widehat{\boldsymbol{U}}_j^{*(c)}\left(\boldsymbol{x}^{(c)}, \boldsymbol{\tau}\right)$ and $\widehat{\boldsymbol{P}}_j^{*(c)}\left(\boldsymbol{x}^{(c)}, \boldsymbol{\tau}\right)$ into (7), we obtain the following expression:

$$\boldsymbol{u}(\boldsymbol{x}) = \frac{1}{2\pi} \sum_{j=1}^{n} \mathbb{R}\Big\{ \sum_{k=0}^{N_T} \boldsymbol{U}_k(\boldsymbol{x}^{(c)}, \boldsymbol{\tau}_c) \boldsymbol{M}_k(\boldsymbol{\tau}_c) - \sum_{k=1}^{N_T} \boldsymbol{P}_k(\boldsymbol{x}^{(c)}, \boldsymbol{\tau}_c) \boldsymbol{N}_k(\boldsymbol{\tau}_c) \Big\}, \tag{10}$$

where:

$$\boldsymbol{U}_k(\boldsymbol{x}^{(c)}, \boldsymbol{\tau}_c) = \begin{cases} -\ln\left(\boldsymbol{x}^{(c)} - \boldsymbol{\tau}_c\right) & \text{for } k = 0 \\ \frac{(k-1)!}{\left(\boldsymbol{x}^{(c)} - \boldsymbol{\tau}_c\right)^k} & \text{for } k \geq 1 \end{cases},$$

$$\boldsymbol{P}_k(\boldsymbol{x}^{(c)}, \boldsymbol{\tau}_c) = \frac{(k-1)!}{\left(\boldsymbol{x}^{(c)} - \boldsymbol{\tau}_c\right)^k} \quad \text{for} \quad k \geq 1.$$

We also expanded equation (10) using Taylor series about any point $\boldsymbol{x}_c \in \{\boldsymbol{x}_{c1}, \boldsymbol{x}_{c2}, ..., \boldsymbol{x}_{ck}\}$ close to some points of solutions (presented in Fig.2) to reduce the number of computations. At last, the integral identity in the IFPIES has the following form:

$$\boldsymbol{u}(\boldsymbol{x}) = \sum_{j=1}^{n} \mathbb{R} \sum_{l=0}^{N_T} \frac{(-1)^l}{2\pi} \left(\sum_{k=0}^{N_T} \frac{(k-1)! \cdot \left(\boldsymbol{M}_k(\boldsymbol{\tau}_c) - \boldsymbol{N}_k(\boldsymbol{\tau}_c)\right)}{\left(\boldsymbol{x}_c - \boldsymbol{\tau}_c\right)^k} \right) \frac{\left(\boldsymbol{x}^{(c)} - \boldsymbol{x}_c\right)^l}{l!}, \tag{11}$$

where $\boldsymbol{N}_0(\boldsymbol{\tau}_c) = 0$.

4 Numerical results

All tests of the presented algorithm were performed on a PC based on an Intel Core i7-14700KF processor with 128 GB of RAM. The program is compiled with g++ (with `-O3 -march=native` optimization) on a 64-bit Ubuntu Linux operating system (kernel 6.14.0). The IFPIES application was developed using pure

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

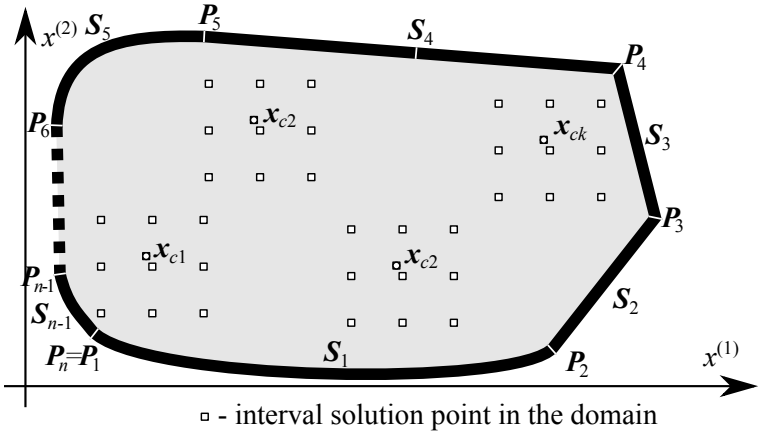

Fig. 2. Groups of interval solution points approximated by corresponding point x_{ci} $(i = 1, 2.., k)$

C/C++ to implement modified directed interval arithmetic, a method that has not yet been documented. As a result, it is not possible to utilize existing C/C++ frameworks for numerical computations, such as BLAS/LAPACK, Eigen, or Armadillo. Additionally, it is important to note that the IFPIES application runs on a single processor core and has not been parallelized, indicating that there is potential for further speed improvements.

The example is the current flow through the square plate presented in Fig. 3. Boundary conditions presented in the figure mean interval potential $\boldsymbol{V}$ (red electrodes) and interval flux $\frac{\partial \boldsymbol{V}}{\partial \boldsymbol{n}}$ on the rest of the boundary (n means normal vector to the boundary segment).

The research focused on the computation speed of solutions in the domain. The IFPIES approximation of the modified IPIES kernels uses 20 terms in the Taylor series, and the GMRES tolerance is 10^{-8}. These parameters were determined based on findings in [24] and [25] regarding the search for optimal IFPIES parameters. The number of collocation points is the same across all segments (2-8). Therefore, we should solve systems from 2 040 to 8 160 algebraic equations. The same number of terms in the Taylor series was also used to approximate the integral identity. Two different numbers of solution points were tested. The first was 65 025, uniformly distributed over the entire domain, divided into 289 groups (x_c points), and the second was 260 100, divided into 1 156 groups. It should be noted that each group in both examples consisted of 225 solution points. The distance between solution points was equal to 1 mm for 289 groups, while 0.5 mm for 1 156 groups, as shown in Fig. 4. As this is the first study utilizing IFPIES to calculate values in the domain of boundary problems, the number of points and groups was set arbitrarily to ensure that solutions could be explored at intervals of 0.5 mm and 1 mm.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

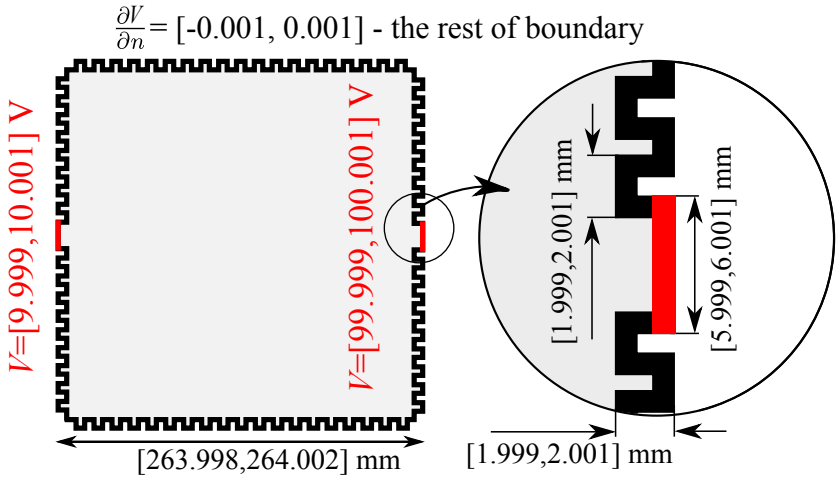

Fig. 3. The example of current flow through the square plate

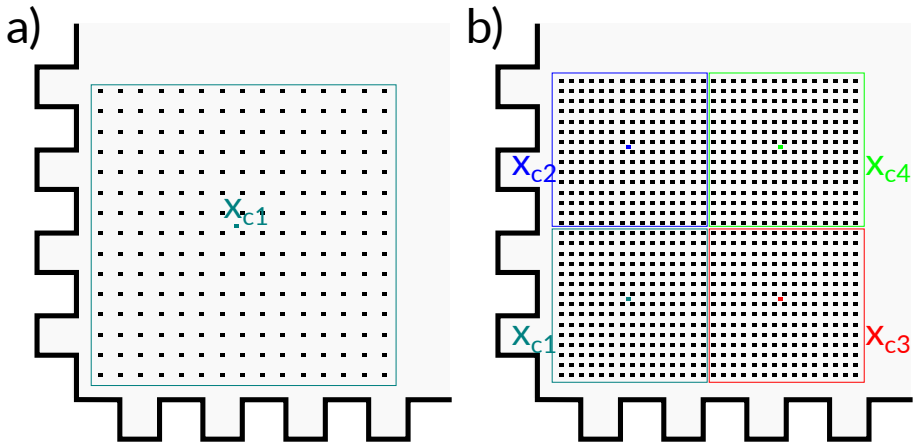

Fig. 4. Location of solution points approximated by x_c points for a) 289 b) 1 156 groups (coloured line mark the boundary of the group approximated by particular x_c)

Tab. 1 presents the obtained time of solving the problem (CPU time - solving), the computation of solutions in the domain (CPU time - domain) and the RAM usage of the applications.

As shown in Tab. 1, the IPIES is much slower than the IFPIES. For the highest number of equations (8 collocation points), the IFPIES required about 42 seconds and 76 MB of RAM to solve the problem and find all solutions in the domain despite the number of solutions points, in comparison the IPIES required almost over 28 minutes and 1 586 MB of RAM for 65 025 and over 93 minutes and 1 586 MB of RAM for 260 100 solution points. It means that RAM utilization

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

Table 1. CPU time and RAM utilization between the IFPIES and FPIES

Number of col. pts.	CPU time - solving [s]		CPU time - domain [s]		RAM utilization [MB]	
	IFPIES	*IPIES*	*IFPIES*	*IPIES*	*IFPIES*	*IPIES*
289 x_c points						
2	0.840	10.622	6.807	297.145	14.388	103.824
3	1.618	24.621	10.279	452.618	20.732	227.484
4	2.720	44.390	13.842	639.618	28.976	400.180
5	4.156	70.502	17.673	834.692	38.264	621.484
6	6.023	103.811	21.655	1 029.510	49.492	895.900
7	8.329	144.483	26.040	1 271.600	62.180	1 216.448
8	11.282	193.757	30.752	1 518.860	76.156	1 586.676
1 156 x_c points						
2	0.849	10.737	7.937	1 118.380	14.296	103.328
3	1.639	24.918	10.939	1 783.180	20.912	228.192
4	2.698	45.567	14.193	2 386.600	28.964	399.972
5	4.147	72.314	17.673	3 082.080	38.772	621.480
6	6.014	105.702	21.357	3 794.660	49.460	895.436
7	8.311	147.307	25.438	4 669.470	62.204	1 216.440
8	11.238	197.708	29.798	5 416.680	76.240	1 586.212

in the IFPIES is strictly tied to solving the system, and finding solutions in the domain does not affect memory usage. Also, it can be seen that the time to find solutions in the domain by the IFPIES is almost the same regardless of the number of solution points, contrary to the IPIES. Fig. 5 presents the overview of the CPU time and RAM utilization between both applications for both numbers of solution points.

We also calculated the mean squared error (MSE) between the IFPIES and the FPIES to assess the method's accuracy. MSE was calculated separately for the infimum and the supremum of interval solutions.

Table 2. MSE of domain solutions between the IFPIES and the IPIES

Number of collocation points	MSE			
	289 x_c points		1 156 x_c points	
	inf	sup	inf	sup
2	$5.075 \cdot 10^{-12}$	$2.922 \cdot 10^{-12}$	$6.344 \cdot 10^{-12}$	$5.844 \cdot 10^{-12}$
3	$3.383 \cdot 10^{-12}$	$4.921 \cdot 10^{-12}$	$5.959 \cdot 10^{-12}$	$6.498 \cdot 10^{-12}$
4	$4.152 \cdot 10^{-12}$	$3.537 \cdot 10^{-12}$	$5.844 \cdot 10^{-12}$	$6.574 \cdot 10^{-12}$
5	$3.998 \cdot 10^{-12}$	$3.230 \cdot 10^{-12}$	$5.729 \cdot 10^{-12}$	$6.882 \cdot 10^{-12}$
6	$1.845 \cdot 10^{-12}$	$2.922 \cdot 10^{-12}$	$5.075 \cdot 10^{-12}$	$5.959 \cdot 10^{-12}$
7	$4.921 \cdot 10^{-12}$	$5.383 \cdot 10^{-12}$	$5.767 \cdot 10^{-12}$	$7.190 \cdot 10^{-12}$
8	$2.307 \cdot 10^{-12}$	$3.383 \cdot 10^{-12}$	$5.190 \cdot 10^{-12}$	$5.998 \cdot 10^{-12}$

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

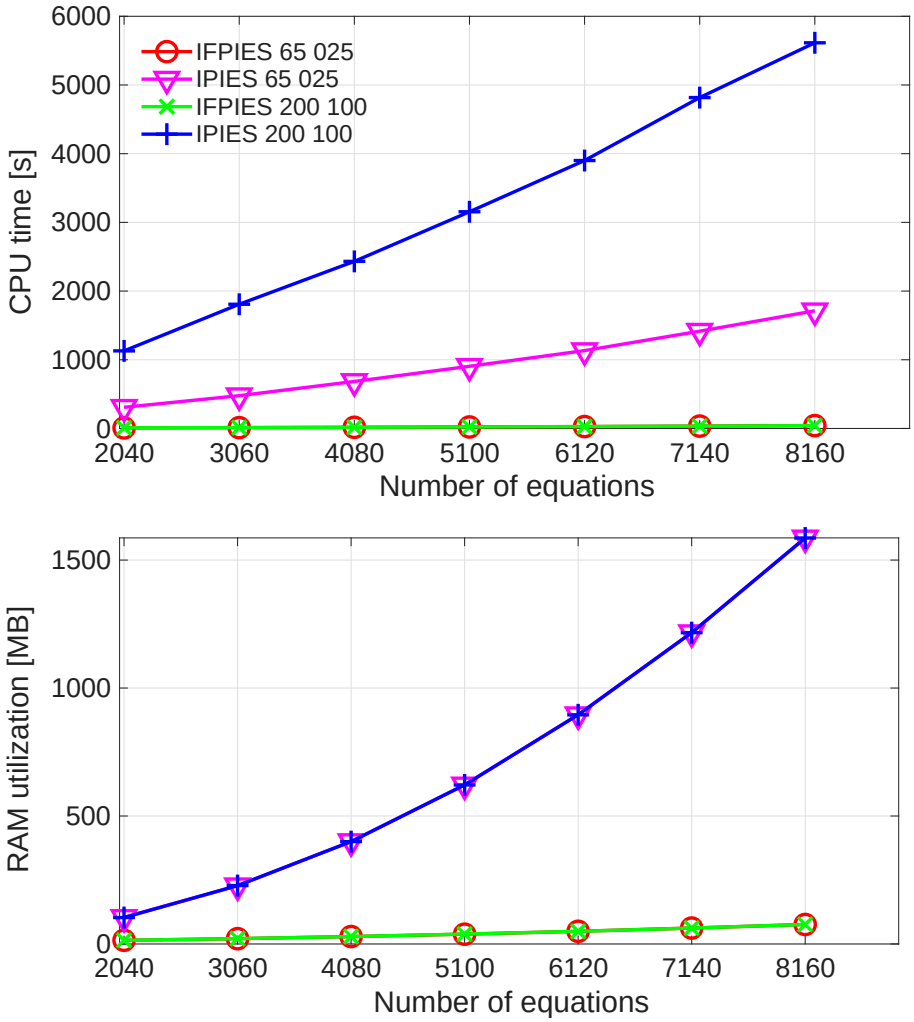

Fig. 5. CPU time and RAM utilization between the IFPIES and the IPIES

As shown in Tab. 2, solutions in the domain by the IFPIES are as accurate as those by the IPIES. The MSE for all examples does not exceed $1 \cdot 10^{-11}$. A graphical representation of MSE is presented in Fig. 6.

5 Conclusions

The paper presents the IFPIES for domain solutions of uncertainly defined 2D potential BVPs. The IFPIES has previously been successfully applied to the modelling of 2D potential BVPs and to the determination of boundary solutions. The fast multipole technique applied to the interval integral identity significantly

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

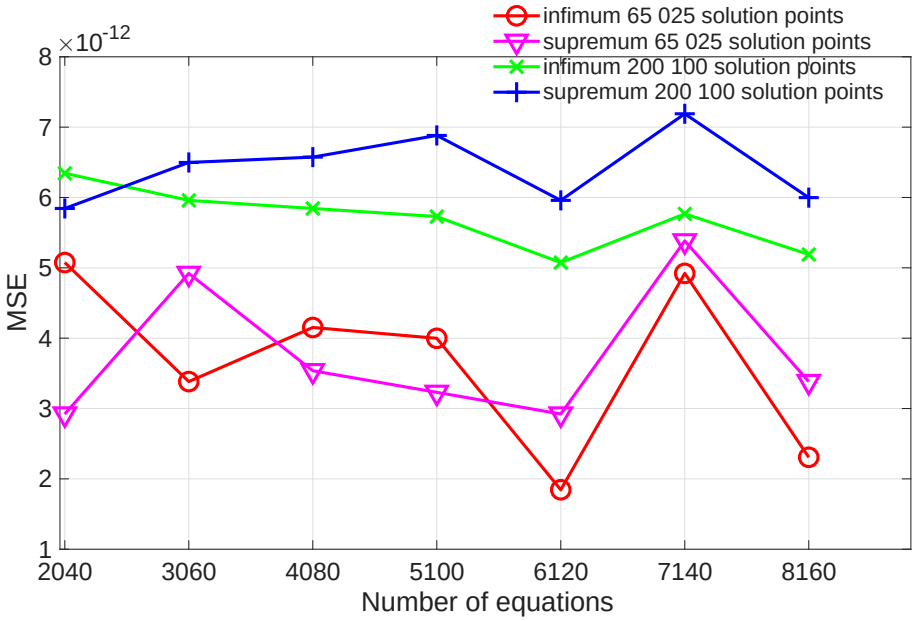

Fig. 6. MSE between the IFPIES and the IPIES

reduces the CPU time for computing domain solutions. The presented examples confirm the high efficiency of the IFPIES in solving complex, uncertainly defined engineering problems on a standard PC in a reasonable time. However, the real power of the IFPIES lies in the scale of the problems to be solved. The IPIES to solve an uncertainly defined problem with a system of 8 160 equations (the square plate, 8 collocation points on each segment) and finding 200 100 solutions in the domain uses almost 1.59 GB RAM over $1\frac{1}{2}$ h. At the same time, the IFPIES requires only 76.24 MB RAM in 41 s.

The obtained results suggest that this line of research should be continued. The authors intend to extend the IFPIES algorithm to problems modelled by other differential equations.

References

1. A. Kużelewski, E. Zieniuk, M. Czupryna, Interval modifications of the fast PIES in solving 2D potential BVPs with uncertainly defined polygonal boundary shape, in: International Conference on Computational Science ICCS 2022, LNCS 13351, Part II, Springer Cham, 18–25 (2022).
2. E. Zieniuk, Hermite curves in the modification of integral equations for potential boundary-value problems. Engineering Computations 20(1–2), 112–128 (2003).
3. O.C. Zienkiewicz, The Finite Element Method, McGraw-Hill, London (1977).
4. O. C. Zienkiewicz, R. L. Taylor, J. Z. Zhu, The Finite Element Method: Its Basis and Fundamentals (7 ed.), Butterworth-Heinemann, Oxford (2013).

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

5. I. Babuska, U. Banerjee, J. E. Osborn, Generalized Finite Element Methods: Main Ideas, Results, and Perspective. International Journal of Computational Methods 1(1), 67–103 (2004).
6. C.A. Brebbia, J.C.F. Telles, L.C. Wrobel, Boundary element techniques, theory and applications in engineering, Springer-Verlag, New York (1984).
7. P. K. Banerjee, R. Butterfield, Boundary Element Methods in Engineering Science, McGraw-Hill, London (1981).
8. J. T. Katsikadelis, Boundary Elements Theory and Applications, Elsevier, Amsterdam (2002).
9. B. Nedjar, A coupled BEM-FEM method for finite strain magneto-elastic boundary-value problems. Computational Mechanics 59(5), 795–807 (2017).
10. T. J. R. Hughes, J. A. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Computer Methods in Applied Mechanics and Engineering 194(39–41), 4135–4195 (2005).
11. L. Beirao Da Veiga, A. Russo, G. Vacca, The Virtual Element Method with curved edges. ESAIM: Mathematical Modelling and Numerical Analysis 53(2), 375–404 (2019).
12. H.-D. Seo, H.-J. Park, J.-I. Kim, P. S. Lee, The particle-attached element interpolation for density correction in smoothed particle hydrodynamics. Advances in Engineering Software 154, 102972 (2021).
13. J. Jaśkowiec, P. Pluciński, Discontinuous Galerkin method in numerical simulation of two-dimensional thermoelasticity problem with single stabilization parameter. Advances in Engineering Software 122, 62-80 (2018).
14. C. A. Duarte, J. T. Oden, H-p clouds - an h-p meshless method. Numerical Methods for Partial Differential Equations 12, 673–705 (1996).
15. E. Zieniuk, M. Kapturczak, D. Sawicki, The NURBS curves in modelling the shape of the boundary in the parametric integral equations systems for solving the Laplace equation, in: International Conference of Numerical Analysis and Applied Mathematics 2015, ICNAAM 2015, AIP Conference Proceedings 1738, 480100 (2016).
16. E. Zieniuk, A. Bołtuć, K. Szerszeń, Shape identification in nonlinear boundary problems solved by PIES method, Acta Mechanica et Automatica 8(1), 16–21 (2014).
17. V. Rokhlin, Rapid solution of integral equations of classical potential theory, Journal of Computational Physics 60(2), 187–207 (1985).
18. L.F. Greengard, V. Rokhlin, A fast algorithm for particle simulations, Journal of Computational Physics 73(2), 325–348 (1987).
19. L.F. Greengard, The rapid evaluation of potential fields in particle systems, The MIT Press, Cambridge (1988).
20. A. Kużelewski, E. Zieniuk, The fast parametric integral equations system in an acceleration of solving polygonal potential boundary value problems. Advances in Engineering Software 141, 102770 (2020).
21. A. Kużelewski, E. Zieniuk, A. Bołtuć, K. Szerszeń, Modified binary tree in the fast PIES for 2D problems with complex shapes, in: International Conference on Computational Science ICCS 2020, LNCS 12138, Part II, Springer-Verlag, Berlin, 1–14 (2020).
22. Y. Saad, M. H. Schultz, A generalized minimal residual algorithm for solving non-symmetric linear systems. SIAM Journal on Scientific and Statistical Computing 7, 856–869 (1986).
23. A. Kużelewski, E. Zieniuk, Solving of multi-connected curvilinear boundary value problems by the fast PIES. Computer Methods in Applied Mechanics and Engineering 391, 114618 (2022).

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28

24. A. Kużelewski A., E. Zieniuk, Searching for the Best Tree Parameters in the IFPIES, in: The 37th Annual European Simulation and Modelling Conference ESM2023, Modelling and Simulation 2023, EUROSIS-ETI Publications, Ghent, Belgium, 48–52 (2023).
25. A. Kużelewski A., E. Zieniuk, Influence of selected IFPIES parameters on CPU time and RAM utilization, in: The 38th Annual European Simulation and Modelling Conference ESM2024, Modelling and Simulation 2024, EUROSIS-ETI Publications, Ghent, Belgium, 288–293 (2024).
26. Y.J. Liu, N. Nishimura, The fast multipole boundary element method for potential problems: A tutorial, Engineering Analysis with Boundary Elements 30(5), 371–381 (2006).
27. E. Zieniuk, A. Kużelewski, Concept of the Interval Modelling the Boundary Shape Using Interval Bezier Curves in Boundary Problems Solved by PIES, in: 12th International Conference of Numerical Analysis and Applied Mathematics ICNAAM 2014, AIP Conf. Proc. 1648, 590002 (2015).
28. S. M. Markov, On directed interval arithmetic and its applications, Journal of Universal Computer Science 1(7), 514–526 (1995).
29. E. Zieniuk, M. Kapturczak, A. Kużelewski, Modification of Interval Arithmetic for Modelling and Solving Uncertainly Defined Problems by Interval Parametric Integral Equations System, in: International Conference on Computational Science ICCS 2018, LNCS 10862, Part III, Springer, Cham, 231–240 (2018).
30. A. Kużelewski, E. Zieniuk, The FMM accelerated PIES with the modified binary tree in solving potential problems for the domains with curvilinear boundaries. Numerical Algorithms 88(3), 1025–1050 (2021).
31. A. Kużelewski, E. Zieniuk, Domain Solutions Obtained by the FPIES for Potential 2D BVPs, in: International Conference on Computational Science ICCS 2025, LNCS 15904, Part II, Springer Cham, 121–133 (2025).

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29921-5_28