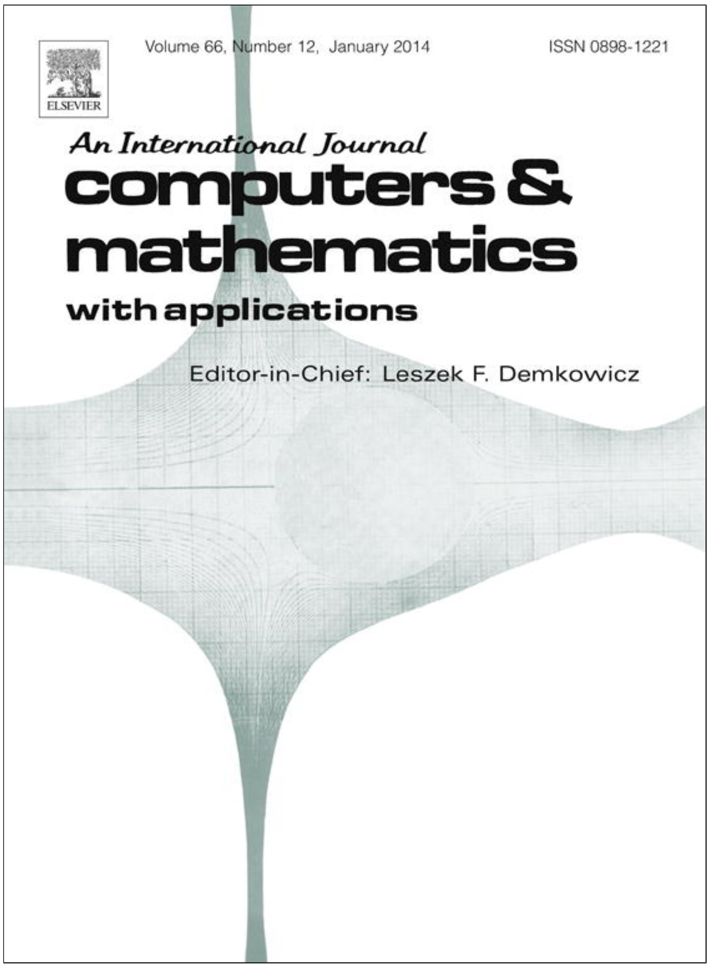


Volume 66, Number 12, January 2014 ISSN 0898-1221

ELSEVIER

An International Journal
computers & mathematics with applications

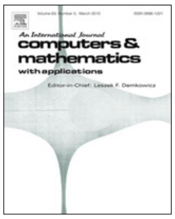
Editor-in-Chief: Leszek F. Demkowicz

Computers and Mathematics with Applications 66 (2014) 2477–2488

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa



Integral equation formulation of an unsteady diffusion–convection equation with variable coefficient and velocity

J. Ravnik [*], L. Škerget

Faculty of Mechanical Engineering, University of Maribor, Smetanova 17, SI-2000 Maribor, Slovenia

ARTICLE INFO

Article history:
Received 10 June 2013
Received in revised form 20 September 2013
Accepted 29 September 2013

Keywords:
Boundary element method
Diffusion–convection equation
Diffusion–advection equation
Variable coefficient
Domain decomposition

ABSTRACT

In this paper we present an integral equation formulation for the time dependent diffusion–convection equation with variable coefficient and velocity with sources. The formulation is based on usage of the steady fundamental solution of the convection–diffusion equation. For a known velocity and coefficient fields, which may change with location and time, the formulation avoids the usage of the gradient of the unknown field function and thus avoids making the problem nonlinear. Two discretization approaches are proposed and compared: a standard single domain boundary–domain element technique and a domain decomposition approach. The validity of the formulation and comparison of discretization approaches is preformed on several challenging test cases. Mesh convergence is reported and the advantages and disadvantages of both approaches are examined.

1. Introduction

Convective and diffusive transport processes occur frequently in nature and engineering. Since fluid flow is in general an unsteady three-dimensional phenomenon, the transport processes taking place in such flows are governed by a velocity field, which changes with time and location. Furthermore, the transport coefficient (i.e. diffusivity in the case of mass transfer, heat conductivity in the case of heat transfer or viscosity in the case of momentum transfer) may also change with time and location. The change may be due to physical processes, such as a change in fluid temperature or pressure. Changes in diffusivity may also be due to the use of modelling of different physical phenomena as diffusion type processes with the introduction of a model based diffusivity. A prominent example of such modelling is the Reynolds averaged turbulence model, which introduces time and spatially varying turbulent viscosity. The turbulent viscosity is added to molecular viscosity in the convection–diffusion momentum transfer equation. In many disciplines, like environmental flows, soil physics, petroleum engineering, chemical engineering and biosciences, diffusive and convective processes with variable coefficients occur. There are many other application areas, where variable diffusion coefficients are used, such as, for example, in lithium ion battery electrodes (Renganathan and White [1]).

Since the convection–diffusion type equations govern many physical processes, many researchers worked on finding new solution methods. Most of the work was done with constant coefficients. Recently, Dehghan [2] proposed a numerical method for the solution of the three-dimensional advection–diffusion equation. Pudykiewicz [3] derived a finite volume algorithm for the solution of the reaction–advection–diffusion equation on the sphere. Sakai and Kimura [4] used a spectral method to solve a nonlinear two-dimensional unsteady advection–diffusion equation, which they transformed into a linear equation. Remešikova [5] proposed an operator splitting scheme for the numerical solution of two-dimensional convection–diffusion–adsorption problems.

[*] Corresponding author. Tel.: +386 22207745.
E-mail address: jure.ravnik@um.si (J. Ravnik).

0898-1221/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.camwa.2013.09.021

Kumar et al. [6] derived analytical solutions for the one-dimensional advection–diffusion equation with variable coefficients in a longitudinal finite initially solute free domain. Mikhailov [7] considered a solution of the heat equation with a time dependent coefficient. Ang et al. [8] used the boundary element method for a second order elliptic partial differential equation with variable coefficients. Similarly, Grzhibovskis et al. [9] considered a Dirichlet problem for the linear second order elliptic PDE in a bounded domain with variable coefficient. A boundary–domain integral equation was used, accelerated by the H-matrix/ACA technique. Several other authors also considered variable coefficient diffusion and Helmholtz equations in non-homogeneous media [10–16].

In this paper we propose an integral formulation, which leads to an efficient solution of unsteady diffusion–convection problems with variable velocity field and coefficient. This work is based on the formulation proposed by Ravnik and Škerget [17] for steady diffusion–convection problems. The gradient of the unknown field function is not included in the final integral formulation. Instead, the gradient of the coefficient is needed and thus, the final integral equation includes only the unknown function on the boundary and in the domain and its flux on the boundary. The proposed equation is linear and after discretization requires only a single solution of a system of linear equations to obtain the solution.

2. Governing equation

Let us consider a domain Ω in $\mathbb{R}^3$ with boundary Γ. The domain is filled with an incompressible fluid. Let $\vec{r}$ be a vector representing a point in the domain and let $\vec{v}$ be the fluid velocity. An unknown field function, u, which is subjected to convective and diffusive processes in the domain, is governed by the following PDE:

$$\frac{\partial u}{\partial t} + \vec{v}(\vec{r},t)\cdot\vec{\nabla}u = \vec{\nabla}\cdot\left(\alpha(\vec{r},t)\vec{\nabla}u\right) + f(\vec{r},t), \quad \vec{r}\in\Omega, \tag{1}$$

with the following Dirichlet and/or Neumann type boundary conditions

$$\begin{aligned} &u(\vec{r},t) = \bar{u}(\vec{r},t), \quad \vec{r}\in\Gamma_D, \\ &\vec{n}\cdot\vec{\nabla}u(\vec{r},t) = q(\vec{r}) = \bar{q}(\vec{r},t), \quad \vec{r}\in\Gamma_N, \end{aligned} \tag{2}$$

where Γ_D and Γ_N are the Dirichlet and Neumann parts of the boundary with $\Gamma = \Gamma_D \cup \Gamma_N$. Boundary conditions vary with time, while the initial conditions are

$$u(\vec{r},0) = u_0. \tag{3}$$

The fluid is incompressible, thus $\vec{\nabla}\cdot\vec{v} = 0$ and the fluid velocity varies in space and in time. The diffusion coefficient, α, in the domain is isotropic, time dependent and non-homogeneous, thus $\alpha(\vec{r},t)$ is a function of the location and time. Furthermore, there are sources $f(\vec{r},t)$ in the domain, which also vary in space and time.

3. Integral representation

The governing equation (1) may be recast into an integral form using the boundary–domain integral method (Škerget et al. [18]). The method relies on the fact that a fundamental solution of the steady diffusion–convection problem exists. It is defined by (Driessen [19])

$$\alpha_0\nabla^2 u^\star + \vec{v}_0\cdot\vec{\nabla}u^\star = -\delta(\vec{r},\vec{\xi}), \tag{4}$$

where α_0 and $\vec{v}_0$ are the constant parts of the transport coefficient and velocity and $\vec{\xi}$ is a source (collocation) point. Eq. (4) in 3D has the following fundamental solution:

$$u^\star(\vec{r},\vec{\xi}) = \frac{1}{4\pi|\vec{r}-\vec{\xi}|\alpha_0}\exp\left(\frac{\vec{v}_0\cdot(\vec{r}-\vec{\xi}) - v_0|\vec{r}-\vec{\xi}|}{2\alpha_0}\right), \tag{5}$$

where $v_0 = |\vec{v}_0|$ and its gradient is

$$\vec{\nabla}u^\star(\vec{r},\vec{\xi}) = \left[\left(\frac{1}{|\vec{r}-\vec{\xi}|} + \frac{v_0}{2\alpha_0}\right)\frac{\vec{r}-\vec{\xi}}{|\vec{r}-\vec{\xi}|} - \frac{\vec{v}_0}{2\alpha_0}\right]u^\star(\vec{r},\vec{\xi}). \tag{6}$$

The variable coefficient and the velocity field are decomposed into constant and variable parts as follows:

$$\alpha(\vec{r}) = \alpha_0 + \alpha', \qquad \vec{v}(\vec{r}) = \vec{v}_0 + \vec{v}', \tag{7}$$

where α' and $\vec{v}'$ are the variable parts.

At time t for a time step Δt the backward Euler finite difference approximation is used to approximate the time derivative as

$$\frac{\partial u}{\partial t} = \beta_1 u + \beta_2 u^c \tag{8}$$

where $\beta_1 = \frac{1}{\Delta t}$ and $\beta_2 = -\frac{1}{\Delta t}$. The u is the function in the next time step and u^c is the function in the current time step.

Using this decomposition, we may rewrite Eq. (1) as

$$-\alpha_0\nabla^2 u + \vec{v}_0 \cdot \vec{\nabla} u = \vec{\nabla} \cdot \left(\alpha'\vec{\nabla} u\right) - \vec{v}' \cdot \vec{\nabla} u + f - (\beta_1 u + \beta_2 u^c). \tag{9}$$

Considering for a moment the terms on the right hand side of Eq. (9) as source terms, the standard BEM derivation (Wrobel [20]) yields the following integral representation for a source point $\vec{\xi} \in \Gamma$ located on the boundary:

$$\begin{aligned} c(\vec{\xi})u(\vec{\xi}) + \int_\Gamma \alpha_0 u \vec{\nabla} u^\star \cdot d\vec{\Gamma} = {} & \int_\Gamma u^\star \alpha_0 \vec{\nabla} u \cdot d\vec{\Gamma} - \int_\Gamma u^\star u \vec{v}_0 \cdot d\vec{\Gamma} + \underbrace{\int_\Omega u^\star \vec{\nabla} \cdot (\alpha' \vec{\nabla} u) d\Omega}_{\text{due to variable coef.}} - \underbrace{\int_\Omega u^\star \vec{v}' \cdot \vec{\nabla} u d\Omega}_{\text{due to var. velocity}} \\ & + \underbrace{\int_\Omega u^\star f d\Omega}_{\text{sources}} - \underbrace{\int_\Omega u^\star (\beta_1 u + \beta_2 u^c) d\Omega}_{\text{unsteady term}}, \end{aligned} \tag{10}$$

where $c(\vec{\xi})$ is the free coefficient given by the solid surface angle at $\vec{\xi}$. The domain integrals in Eq. (10) are due to the source terms of Eq. (9). These domain integrals include the variable parts of the coefficient α' and the variable part of velocity $\vec{v}'$, sources f and the unsteady accumulation term. The first two domain integrals have a gradient of the solution, $\vec{\nabla} u$, in their kernels. Integral equation (10) could be used as the basis for discretization, but since the gradient of the solution is needed to construct the source terms, numerical differentiation would be needed. Furthermore, an iterative scheme would be needed, where the solution of (10) and the numerical calculation of the gradient $\vec{\nabla} u$ would be alternatively calculated until convergence is achieved.

Thus, in the following, we rewrite Eq. (10) into an expression that avoids the gradient of the unknown function. Let us first focus on the domain integral due to the variable coefficient. Using rules for chain differentiation, we may write

$$\int_\Omega u^\star \vec{\nabla} \cdot (\alpha' \vec{\nabla} u) d\Omega = \int_\Omega \vec{\nabla} \cdot (u^\star \alpha' \vec{\nabla} u) d\Omega + \int_\Omega u \vec{\nabla} \alpha' \cdot \vec{\nabla} u^\star d\Omega - \int_\Omega \vec{\nabla} \cdot (\alpha' u \vec{\nabla} u^\star) d\Omega + \int_\Omega \alpha' u \nabla^2 u^\star d\Omega. \tag{11}$$

The two integrals that feature a divergence of the kernel can be written as boundary integrals using the Gauss clause, yielding

$$\int_\Omega u^\star \vec{\nabla} \cdot (\alpha' \vec{\nabla} u) d\Omega = \int_\Gamma u^\star \alpha' \vec{\nabla} u \cdot d\vec{\Gamma} + \int_\Omega u \vec{\nabla} \alpha' \cdot \vec{\nabla} u^\star d\Omega - \int_\Gamma \alpha' u \vec{\nabla} u^\star \cdot d\vec{\Gamma} + \int_\Omega \alpha' u \nabla^2 u^\star d\Omega. \tag{12}$$

The kernel of the last domain integral in Eq. (12) includes a Laplacian of the fundamental solution. This can be rewritten by using the definition in Eq. (4) as

$$\int_\Omega \alpha' u \nabla^2 u^\star d\Omega = -\frac{1}{\alpha_0} \int_\Omega \alpha' u \delta(\vec{r}, \vec{\xi}) d\Omega - \frac{1}{\alpha_0} \int_\Omega \alpha' u \vec{v}_0 \cdot \vec{\nabla} u^\star d\Omega. \tag{13}$$

At his point we choose the constant part of the coefficient to be $\alpha_0 = \alpha(\vec{\xi})$. Thus α' is equal to zero at the source point, and since the Kronecker delta is zero everywhere else, the first integral on the right hand side of Eq. (13) vanishes. Using this, Eq. (12) simplifies to

$$\int_\Omega u^\star \vec{\nabla} \cdot (\alpha' \vec{\nabla} u) d\Omega = \int_\Gamma u^\star \alpha' \vec{\nabla} u \cdot d\vec{\Gamma} - \int_\Gamma \alpha' u \vec{\nabla} u^\star \cdot d\vec{\Gamma} + \int_\Omega u \left(\vec{\nabla} \alpha' - \frac{\alpha'}{\alpha_0} \vec{v}_0 \right) \cdot \vec{\nabla} u^\star d\Omega. \tag{14}$$

Next, we turn our attention to the domain integral of Eq. (10), which is due to the variable velocity field. By using the definition of divergence and the fact, that the velocity field is solenoidal, we can write:

$$\begin{aligned} \int_\Omega u^\star \vec{v}' \cdot \vec{\nabla} u d\Omega &= \int_\Omega \vec{\nabla} \cdot (u^\star u \vec{v}') d\Omega - \int_\Omega u \vec{v}' \cdot \vec{\nabla} u^\star d\Omega \\ &= \int_\Gamma u^\star u \vec{v}' \cdot d\vec{\Gamma} - \int_\Omega u \vec{v}' \cdot \vec{\nabla} u^\star d\Omega, \end{aligned} \tag{15}$$

where the Gauss clause has been used to transform the domain integral into a boundary integral.

Noticing that $\vec{\nabla} \alpha = \vec{\nabla} \alpha'$, $\alpha'/\alpha_0 = \alpha/\alpha_0 - 1$, using (14) and (15) we rewrite Eq. (10) as

$$\begin{aligned} c(\vec{\xi})u(\vec{\xi}) + \int_\Gamma \alpha u \vec{\nabla} u^\star \cdot d\vec{\Gamma} = {} & \int_\Gamma u^\star \alpha \vec{\nabla} u \cdot d\vec{\Gamma} - \int_\Gamma u^\star u \vec{v} \cdot d\vec{\Gamma} + \int_\Omega u \left(\vec{\nabla} \alpha + \vec{v} - \frac{\alpha}{\alpha_0} \vec{v}_0 \right) \cdot \vec{\nabla} u^\star d\Omega \\ & + \int_\Omega \left(f - (\beta_1 u + \beta_2 u^c)\right) u^\star d\Omega. \end{aligned} \tag{16}$$

Here α is the coefficient and $\vec{v}$ is the fluid velocity, which both vary in space and time. Also, α_0 and $\vec{v}_0$ are the constant parts, which are used to calculate the fundamental solution and its gradient, where α_0 is the coefficient at the location of the source point. The constant part of the velocity can be chosen arbitrarily, for example, it can be the average of the velocity field in the domain or the velocity at the source point.

Eq. (16) includes boundary values of the function u as well as boundary values of the normal flux, $q = \vec{\nabla} u \cdot \vec{n}$. Time dependent boundary conditions in the form of either function or flux must be known in order to use (16) to solve for the other. With the boundary values of function and flux known, this formula may be used to find the time development of the unknown field u within the domain. Within each time step Eq. (16) is solved to obtain the new field based on the known velocity field, transport coefficient and sources. Since none of the fields depend on the function u, the problem is linear and within each time step only one solution of (16) is needed.

4. Discretization

The integral equation (16) features both boundary and domain integrals, thus a boundary-only discretization procedure is not possible. The variable velocity field and variable coefficient as well as presence of sources in the domain require the discretization of the whole domain.

A computational mesh, consisting of hexahedral elements, is set up in the domain, i.e. $\Omega = \sum_e \Omega_e$. The sides of the elements, which make up the outer boundary of the domain are deemed boundary elements, $\Gamma = \sum_b \Gamma_b$. With such a mesh, Eq. (16) may be rewritten as

$$c(\vec{\xi})u(\vec{\xi}) + \sum_b \int_{\Gamma_b} \alpha u \vec{\nabla} u^\star \cdot d\vec{\Gamma} = \sum_b \int_{\Gamma_b} u^\star \alpha q d\Gamma - \sum_b \int_{\Gamma_b} u^\star u \vec{v} \cdot d\vec{\Gamma} + \sum_e \int_{\Omega_e} u \left(\vec{\nabla}\alpha + \vec{v} - \frac{\alpha}{\alpha_0}\vec{v}_0 \right) \cdot \vec{\nabla} u^\star d\Omega$$
$$+ \sum_e \int_{\Omega_e} \left(f - (\beta_1 u + \beta_2 u^c) \right) u^\star d\Omega. \tag{17}$$

The hexahedral domain elements have 27 nodes, which are used for continuous quadratic interpolation of the function by employing Lagrangian interpolation functions Φ_i. Nine nodes on each boundary element allow for the continuous quadratic interpolation of a function using boundary interpolation functions φ_i. Finally, each boundary element includes four nodes for the discontinuous interpolation of the flux using linear interpolation functions ϕ_i. Thus, a function, u, is interpolated over a boundary element as $u = \sum \varphi_i u_i$, inside each domain element as $u = \sum \Phi_i u_i$, while the flux is interpolated over the boundary element as $q = \sum \phi_i q_i$. Inserting these interpolations into (17) we have

$$c(\vec{\xi})u(\vec{\xi}) + \sum_b \sum_i \alpha_{b,i} u_{b,i} \int_{\Gamma_b} \varphi_i \vec{\nabla} u^\star \cdot d\vec{\Gamma} = \sum_b \sum_i \alpha_{b,i} q_{b,i} \int_{\Gamma_b} \phi_i u^\star d\Gamma - \sum_b \sum_i u_{b,i} \vec{v}_{b,i} \cdot \int_{\Gamma_b} \varphi_i u^\star \vec{n} d\Gamma$$
$$+ \sum_e \sum_i u_{e,i} \left(\vec{\nabla}\alpha_{e,i} + \vec{v}_{e,i} - \frac{\alpha_{e,i}}{\alpha_0}\vec{v}_0 \right) \cdot \int_{\Omega_e} \Phi_i \vec{\nabla} u^\star d\Omega$$
$$+ \sum_e \sum_i \left(f_{e,i} - (\beta_1 u_{e,i} + \beta_2 u_{e,i}^c) \right) \int_{\Omega_e} \Phi_i u^\star d\Omega. \tag{18}$$

Please note that the integrals in (18) depend solely on the mesh shape and on the fundamental solution. Thus, after choosing the collocation point $\vec{\xi}$ and the constant part of the velocity field, they may be calculated beforehand and do not change through the time stepping simulation procedure. In order to calculate the integrals, a Gaussian quadrature algorithm is used. The integrals are calculated in a local coordinate system via weighted summation of 16 integration points per coordinate axis. In the case of high aspect ratios of hexahedral elements, the elements are divided into parts whose aspect ratio is approximately equal to one.

Calculation of the free coefficient $c(\vec{\xi})$ is preformed indirectly. We consider a known solution of the rigid body movement, $u = 1$, $q = 0$ and use it to calculate $c(\vec{\xi})$. If the source point is located on the surface, we know that $c = 1/2$, also if the source point is inside of the element then $c = 1$. These two relationships are used to check the accuracy of the calculated integrals, which depends on the number of integration points.

In order to set up a system of linear equations, one needs to place the collocation point $\vec{\xi}$ at different locations, each yielding a single equation. The integrals are calculated and grouped into matrices. We have

$$[H] = \int_\Gamma \varphi_i \vec{\nabla} u^\star \cdot \vec{n} d\Gamma, \qquad [G] = \int_\Gamma \phi_i u^\star d\Gamma, \tag{19}$$

$$[\vec{A}] = \int_\Gamma \varphi_i \vec{n} u^\star d\Gamma, \qquad [\vec{D}] = \int_\Omega \Phi_i \vec{\nabla} u^\star d\Omega, \qquad [B] = \int_\Omega \Phi_i u^\star d\Omega, \tag{20}$$

where the square brackets denote matrices of integrals. Let the curly brackets denote vectors of nodal values of functions. Then, the discrete version of Eq. (18) may be written as

$$[H]\{\alpha u\} = [G]\{\alpha q\} - [\vec{A}] \cdot \{\vec{v} u\} + [\vec{D}] \cdot \left\{ u \left(\vec{\nabla}\alpha + \vec{v} - \frac{\alpha}{\alpha_0}\vec{v}_0 \right) \right\} + [B] \left\{ f - (\beta_1 u + \beta_2 u^c) \right\}. \tag{21}$$

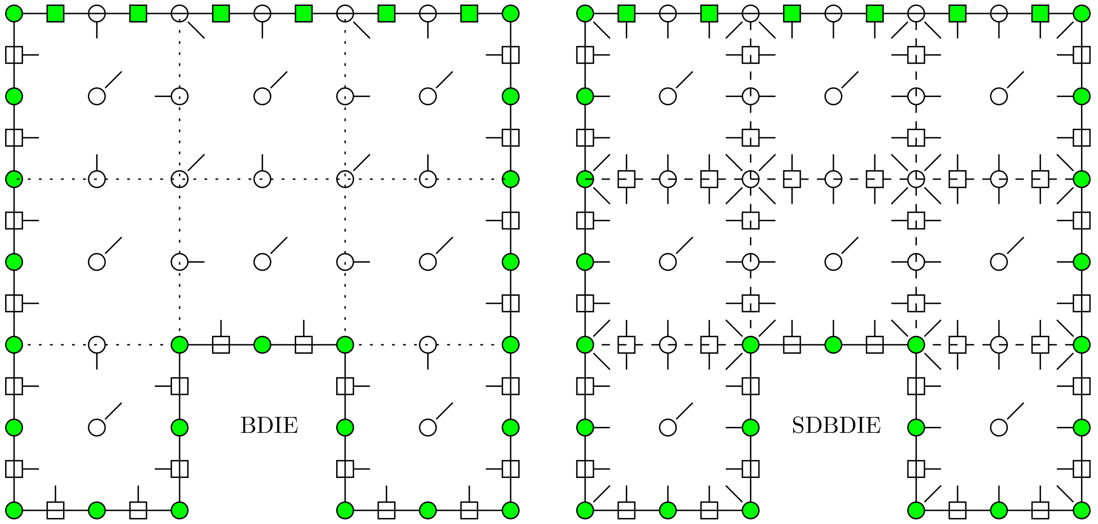


Fig. 1. Setup of system of linear equation for the BDIE (left) and the SDBDIE (right). A 2D domain is presented, where Neumann boundary conditions are known on the top wall and Dirichlet boundary conditions on all other walls. Empty circles denote nodes where function is unknown, full circles denote function nodes, where function is known via boundary conditions. Squares denote flux nodes. A short line coming out of a node denotes an equation with collocation point in that node. Solid line is the external boundary of the domain, dashed lines are subdomain boundaries and dotted lines are the mesh element boundaries.

In this work, we examine two collocation based discretization approaches. Both are based on Eq. (21) but differ in the placement of collocation points and in the setup of the linear system of equations. The first approach, abbreviated BDIE (boundary–domain integral equation), utilizes a single domain and produces a fully populated system of linear equations. The collocation point is placed at boundary nodes in accordance with boundary conditions, i.e. in boundary elements where Dirichlet boundary conditions are given, the collocation point is placed at flux nodes and where Neumann boundary conditions are known, the collocation point is placed at function nodes. Furthermore the collocation point is placed at all internal nodes as well, to facilitate simultaneous calculation of the solution in the entire domain. The resulting system matrix is full and a LU-decomposition based solver is used to solve the system.

The second approach, abbreviated SDBDIE (subdomain BDIE), employs a domain decomposition technique (Ramšak et al. [21]). It considers each mesh element to be a subdomain and sets up a separate system of equations for each individual subdomain. The six sides of each subdomain are treated as boundary elements and the collocation point is placed at boundary nodes according to the boundary conditions. The final solution is obtained, when compatibility boundary conditions are employed between subdomains. The conditions require the function and flux to be continuous across subdomains. Finally, a sparse over-determined system of linear equations is obtained. The system is over-determined due to the fact that nodes are shared between subdomains. A least-squares based solver with diagonal preconditioning (Paige and Saunders [22]) is used to obtain the solution of this system. The solver approximates a pseudo inverse of the over-determined system of equations, thus the diagonal preconditioner is calculated via a product of transpose of the system matrix multiplied by the system matrix.

Fig. 1 shows a 2D cut through a sample domain to graphically illustrate the collocation node placement and the setup of the linear system of equations for BDIE and SDBDIE. On the boundary, both methods have the same number of unknowns (either function or flux nodes, based on boundary conditions). In the domain, only the function values at the nodes are unknown in the BDIE, while the function values and fluxes at nodes are unknown in the SDBDIE. This makes the system of linear equations in the case of SDBDIE larger.

4.1. Complexity estimates

If we consider a cubic domain meshed by n^3 elements (in case of the BDIE or n^3 subdomains in case of the SDBDIE), we notice that the number of unknowns in case of the BDIE is $(2n+1)^3+(6\cdot 4)n^2-b$, while for the SDBDIE it is $(6\cdot 4+3^3)\cdot n^3-b$. Here b is the number of values known through boundary conditions. For large problems, where n is large, we can estimate that the SDBDIE system of linear equations has approximately $51/8$ =6.375 times more equations than the BDIE system. However, when looking at the number of nonzero entries in the matrices, the situation is reversed. In case of the BDIE, the $[\vec{D}]$ and $[B]$ matrices are full. The number of rows in these matrices is equal to the number of collocation point, while the number of columns is equal to the number of function nodes. All together they have $\approx n^6$ nonzero entries. In the SDBDIE case, the matrices are sparse and the number of nonzero columns is determined by the number of subdomains surrounding the collocation point. In the worst case, when the collocation point belongs to eight subdomains, the number of nonzero entries is $5^3+108=233$ and the total number of nonzero entries is $\approx 233\cdot n^3$. Thus, the BDIE system of equations has $n^3/233$ times more nonzero matrix elements that the SBDIE. Since memory requirements for the SDBDIE scale as $\propto n^3$ and

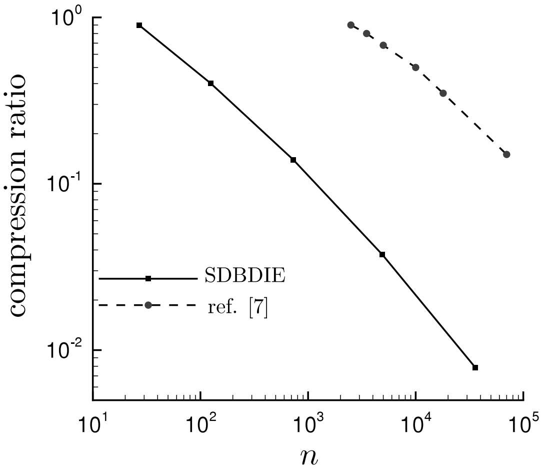


Fig. 2. Comparison of SDBDIE compression ratio with H-matrix/ACA approach of Grzhibovskis et al. [9].

for the BDIE they scale as $\propto n^6$, computer memory availability limits the BDIE mesh size, while with the SDBDIE much larger meshes can be considered.

The SDBDIE memory requirements can be compared to fast techniques of other authors such as wavelet compression, fast multipole method or adaptive cross approximation. For example, Grzhibovskis et al. [9] considered a linear second order elliptic PDE with variable coefficient and used the H-matrix/ACA compression for domain integrals. They report a ≈ 0.5 compression ratio for a $[D]$ integral matrix emerging from a 10^4 node mesh of a ball shaped geometry and a compression ratio of ≈ 0.15 for $7 \cdot 10^4$ nodes. Let us define the compression ratio of the SDBDIE as the ratio between the computer memory needed by the SDBDIE divided by the memory needed by the full matrix the BDIE approach. In Fig. 2 we compare the SDBDIE compression ratio with data reported by Grzhibovskis et al. [9]. A cubical domain was used to generate the SDBDIE compression ratios. Both approaches show a similar compression increase with increasing number of nodes. The SDBDIE yields compression for a low number of nodes, while the ACA does not. The reason for this is that the SDBDIE does not need any additional data structure, while for the ACA a hierarchical tree is needed which allows for the computation of matrix vector products. Looking at compression ratios we see that the SDBDIE compression is about an order of magnitude better than that of the ACA. This is due to the fact that through the use of domain decomposition the SDBDIE becomes a local method, while the ACA merely approximates a standard non-local approach. The main drawback of the local nature of the SDBDIE becomes evident in cases of simulation of high Peclet number flows, which feature sharp solution gradients, as it is shown in the test cases section.

5. Test cases

In order to verify the validity of the proposed integral equation (16) and to compare the BDIE and the SDBDIE discretization approaches, we considered several test cases and compared the approximate solutions to the analytical solutions. The test cases have been designed to test the validity of the proposed numerical algorithms. Oscillating solutions and different ratios between convection and diffusion processes were used to highlight the features of the methods. Furthermore, in order to examine the convergence properties of the algorithms, the test cases have been solved with several meshes and using different time steps.

As the algorithm was developed in 3D it requires a three-dimensional domain with Dirichlet and/or Neumann boundary conditions applied on the sides of the domain. In the case, where a 2D example was solved, it was solved in the $x - y$ plane with a Neumann zero flux boundary condition applied on the top and bottom z planes. For a 1D example, a Neumann zero flux boundary condition was used on both top and bottom z planes and front and back y planes and the solution was obtained in the x direction.

To estimate the approximation error, the root mean square norm was used. It is based on the difference between the simulation result u and the analytical solution u_a as

$$\|u - u_a\|_{RMS} = \left(\frac{\sum_{i=1}^{n} (u_i - u_{a,i})^2}{\sum_{i=1}^{n} u_{a,i}^2} \right)^{1/2}, \tag{22}$$

where i denotes a nodal value and n is the number of all nodes in the domain. In the case of 2D problems, only nodes in one plane are used, while for 1D problems only nodes on a line profile through the domain are considered.

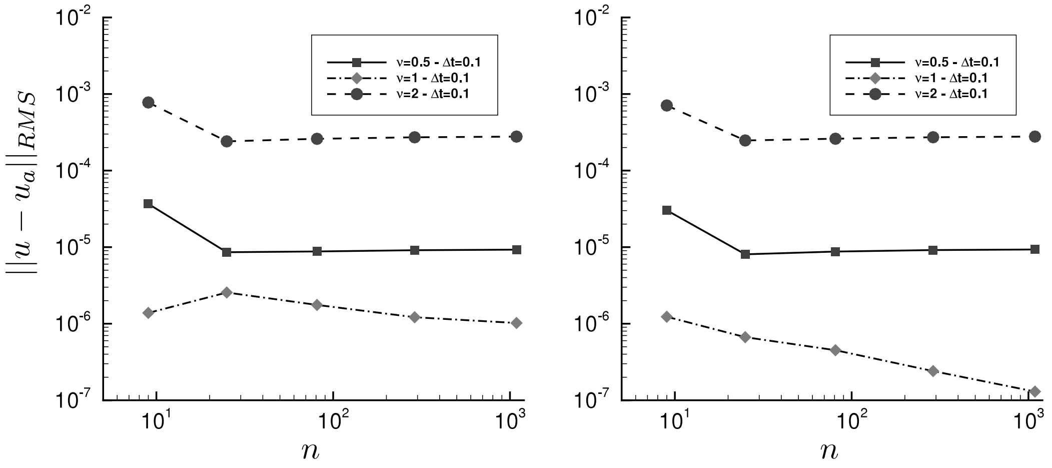


Fig. 3. Test case I — RMS versus n at time $t = 1$ calculated using BDIE (right) and SBDIE (left) for three values of parameter $\nu = 0.5, 1, 2$. Time step of $\Delta t = 0.1$ was used.

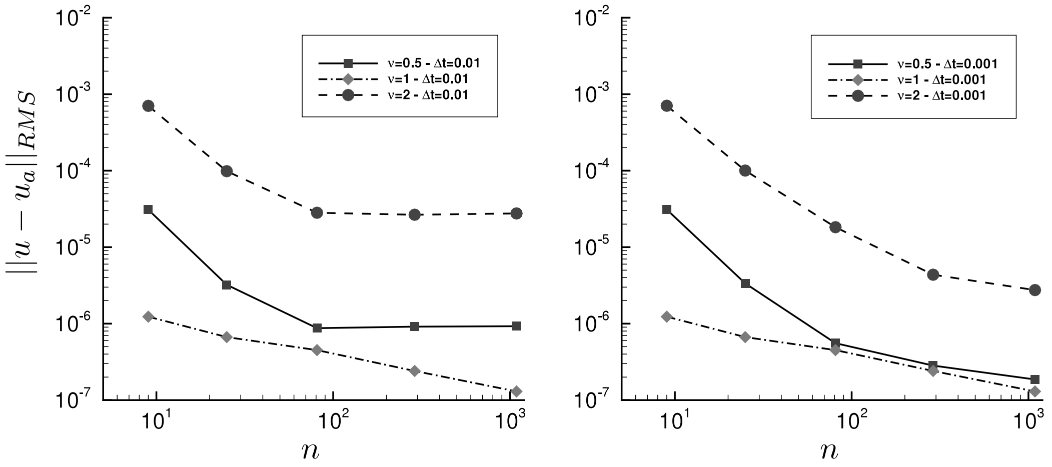


Fig. 4. Test case I — RMS versus n at time $t = 1$ calculated using single domain BEM (BDIE) for three values of parameter $\nu = 0.5, 1, 2$. Left: $\Delta t = 0.01$, right: $\Delta t = 0.001$.

5.1. Test case I

Let us consider a 2D case where the domain is $\Omega = [1, 2]^2$ and let ν be a parameter. The fluid velocity is constant; $\vec{v} = (1, 1)$. The transport coefficient depends on time and changes linearly within the domain; $\alpha = (t + x + y)/\nu$. The sources are defined by $f = \nu(t + x + y)^{\nu-1}$. Dirichlet boundary conditions are considered on the left and right walls, $u(1, y, t) = (t + 1 + y)^{\nu}$ and $u(2, y, t) = (t + 2 + y)^{\nu}$. Neumann boundary conditions are applied on the top and bottom walls, $q(x, 1, t) = -\nu(t + x + 1)^{\nu-1}$ and $q(x, 2, t) = \nu(t + x + 2)^{\nu-1}$. At time $t = 0$, the field is initialized to $u = (x + y)^{\nu}$. The analytical solution of this test case is $u_a = (t + x + y)^{\nu}$.

This case features a constant velocity field. The fluid is flowing across the domain at a 45° angle. The transport coefficient and sources are time and position dependent. We solved this test case for three values of the parameter $\nu = 0.5$, 1 and 2. Meshes had $1, 2^2, 4^2, 8^2$ and 16^2 equal sized elements in the solution plane. The total number of nodes used for discretization of u in these meshes was from 9 to 1089.

Some results are presented in Figs. 3–5 for the RMS norm versus the number of nodes in the computational mesh at time $t = 1$. At $\nu = 1$ this test case is particularly simple, as the solution is simply a linear function of position and time. This is reflected in the results, as the norms are $\approx 10^{-6}$ regardless of the mesh density and time step size. This was expected, as linear functions are fully captured by the discretization scheme.

However, in the case of $\nu = 0.5$ and $\nu = 2$ the solution is not a linear function of the location and time, thus, the accuracy of the solution depends on the mesh density and time step. For a large time step of $\Delta t = 0.1$ (Fig. 3) improvement of the solution accuracy is observed only up to a certain mesh density. After that point, increasing the mesh density does not improve the accuracy, as the time step is too coarse to allow improvement. When using a shorter time step, (Figs. 4 and 5) the accuracy improves with the mesh density. On the other hand, the solution accuracy is limited by $\approx 10^{-6}$. This is caused by the limited accuracy of the calculation of the integrals and the accuracy of the solution of systems of linear equations.

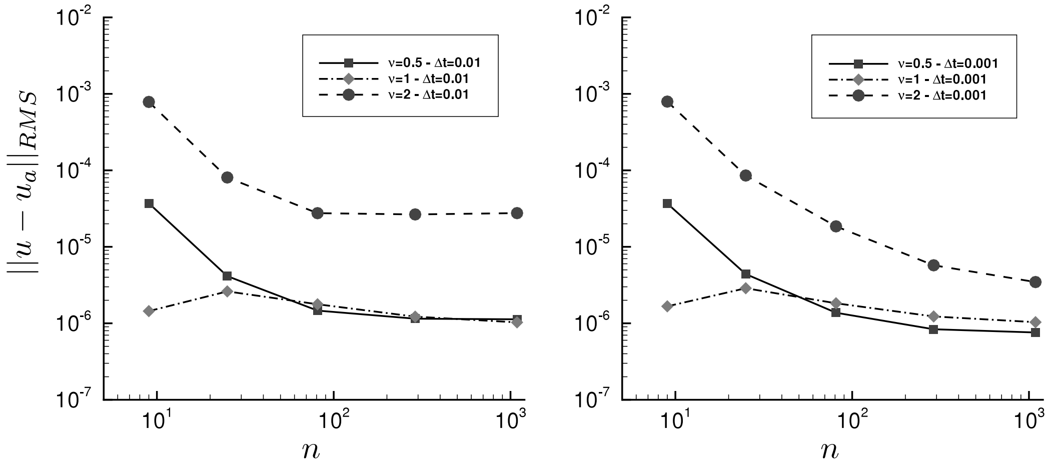


Fig. 5. Test case I — RMS versus n at time $t = 1$ calculated using subdomain domain BEM (SDBDIE) for three values of parameter $\nu = 0.5, 1, 2$. Left: $\Delta t = 0.01$, right: $\Delta t = 0.001$.

5.2. *Test case II*

Let us consider a 3D case where the domain is $\Omega = [1, 2]^3$ and ν a parameter. The coefficient varies with location and time in an oscillatory manner, where the parameter ν defines the frequency of oscillations; $\alpha = 2+\sin(t+\nu(x+y+z)\pi)$. The fluid velocity is constant $\vec{v} = (1, 1, 1)$. The sources are defined by $f = xyz+t(yz+xy+xz)(1-\pi\nu\cos(t+\pi(x+y+z)\nu))$. Dirichlet boundary conditions are considered on the left and right walls, $u(1, y, z, t) = tyz$ and $u(2, y, z, t) = 2tyz$. Neumann boundary conditions are applied on the top and bottom walls, $q(x, y, 1, t) = -txy$ and $q(x, y, 2, t) = txy$ as well as on the front and back walls $q(x, 1, z, t) = -txz$ and $q(x, 2, z, t) = txz$. At time zero, the field is initialized to $u = 0$. The analytical solution of this test case is $u_a = txyz$.

The meshes used consisted of 2^3, 4^3 and 8^3 elements with 125, 729 and 4913 nodes. The SDBDIE was tested on the 16^3 (35937) mesh as well, while due to computer memory constraints, the BDIE was not tested on this mesh. All mesh elements of a single mesh have the same size and shape.

Fig. 6 presents the norms for the BDIE and the SDBDIE calculated using $\Delta t = 0.1$. Both methods show good convergence properties. The accuracy of the approximation deteriorates with increasing frequency of coefficient oscillations and improves with mesh density. The size of the time step is not a limiting factor in this case, as using shorter time steps does not improve the results. This is due to the fact, that the solution is a linear function of time. The solution accuracy in this case is determined by the relationship between the degree to which the coefficient oscillates in the domain and the mesh density.

Since the size of the elements in our meshes decrease by a factor of 2, we were able to use the Richardson extrapolation to estimate the order of our methods. Comparing the simulation errors at mesh element sizes h and $h/2$, we define the order of the method $\mathcal{O}$ as

$$\|u - u_a\|_{RMS,\frac{h}{2}} = \left(\frac{1}{2}\right)^{\mathcal{O}} \|u - u_a\|_{RMS,h}. \tag{23}$$

Using the results of case II and averaging over different mesh sizes and ν, we obtain $\mathcal{O} = 2.51 \pm 0.4$ for the BDIE and $\mathcal{O} = 2.38 \pm 0.4$ for the SDBDIE, establishing more than second order accuracy for the proposed methods.

5.3. *Test case III*

In this test case we consider a generalization of a well known 1D entry flow problem. The entry flow problem balances diffusion and convection using the Peclet number ν, and features constant diffusivity and time independent boundary conditions. Here, we introduce a generalized entry flow problem, where diffusivity changes in the domain according to x^m, where m is a new parameter. The 1D transport equation for generalized entry flow is

$$\nu\frac{\partial u}{\partial x} = \frac{\partial}{\partial x}\left(x^m\frac{\partial u}{\partial x}\right), \tag{24}$$

which is a steady 1D version of Eq. (1) with $f = 0$, $\vec{v} = (\nu, 0, 0)$ and $\alpha = x^m$. Considering a domain $\Omega = [a, b]$, where $0 \notin \Omega$ and boundary conditions $u(a) = 0$, $u(b) = 1$ the following solution may be found:

$$u(x) = \begin{cases} \dfrac{x^\nu - a^\nu}{b^\nu - a^\nu} & m = 1 \\ \dfrac{\exp\left(-\frac{\nu}{m-1}\left(x^{1-m} - a^{1-m}\right)\right) - 1}{\exp\left(-\frac{\nu}{m-1}\left(b^{1-m} - a^{1-m}\right)\right) - 1} & m \neq 1 \;\&\; m \geq 0. \end{cases} \tag{25}$$

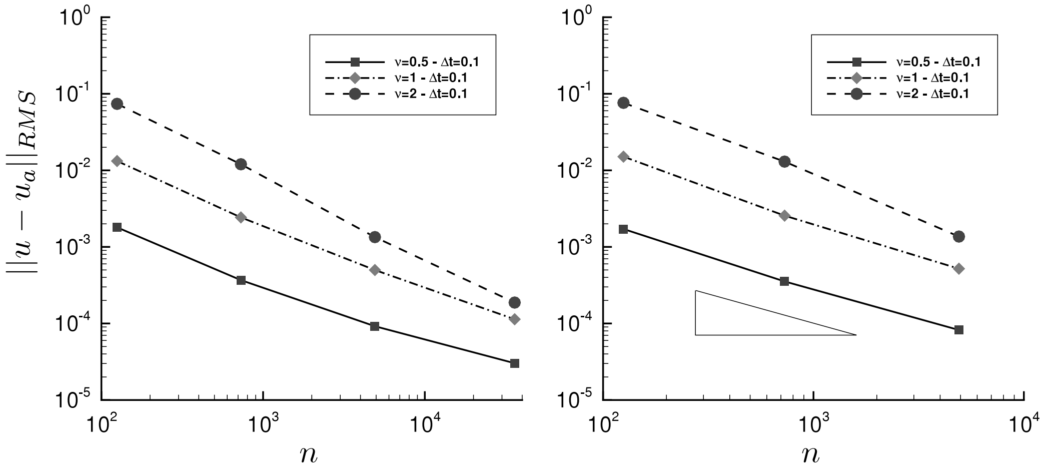


Fig. 6. Test case II — RMS versus n at time $t = 1$ calculated BDIE (right) and SDBDIE (left) for three values of parameter $\nu = 0.5, 1, 2$ using a time step of $\Delta t = 0.1$. Second order slope is also shown.

The derivative of the solution du/dx at $x = a$ is ≈ 0, while at $x = b$ the derivative limits for high Peclet number values to $\lim_{\nu\to\infty} du/dx|_{x=b} = \nu/b^m$.

The case without varying coefficient ($m = 0$) has been solved by Qiu et al. [23] for high Peclet number values. They used the standard BEM in two dimensions and a special treatment of the singular integrals. Furthermore, they used transformations to avoid directly evaluating Bessel functions for Cauchy principal value and hypersingular integrals, which yielded virtually analytical results for Peclet numbers up to $\nu = 10^7$. Other methods, such as the finite volume method [21] are unable to reach such high values of the Peclet number, as they cannot capture the sharp solution profile.

We solved this test case for several values of ν and m examining the properties of our proposed solution algorithms. We choose $a = 1$ and $b = 2$ in a block shaped domain $(a, 0, 0) \times (b, 0.01, 0.01)$. Dirichlet boundary conditions were applied at $x = a$ and $x = b$ while no-flux Neumann boundary conditions were used on four other walls.

As the solution features a very sharp profile at $x = b$, the meshes used in our calculation were concentrated towards $x = b$. Mesh concentration was performed using a geometrical series using a mesh concentration ratio R. The mesh concentration ratio R is defined as the ratio between the longest and the shortest element of the mesh.

The most challenging part of this test case is the correct estimation of the flux at $x = b$. As this value is proportional to the Peclet number, it becomes very large in cases of high Peclet numbers and the corresponding function resembles a step function. In Fig. 7 the RMS error of the flux at $x = b$ is shown for $\nu = 10^3$ and $m = 0 \ldots 4$. Two different mesh concentration ratios have been used along with several meshes. We observe good convergence properties. The results for $R = 100$ are more accurate, since a short element in the area where a high gradient is expected, yields a better estimation of the solution.

In Table 1 we compare the performance of the BDIE and the SDBDIE. All simulations are done on a 17 node mesh with $R = 100$. We observe that the BDIE yields reasonably accurate results for a Peclet number $\leq 10^5$. At $\nu = 10^6$ the results for $m = 0$ and $m = 1$ are poor, while for larger m they are still accurate. In the case of the SDBDIE results only up to $\nu = 10^3$ are accurate, while for higher values of the Peclet number, the method fails. The reason for this is the local nature of the SDBDIM, which is a consequence of the usage of domain decomposition. Since the integral equation is solved for each individual subdomain, the approximation is limited to the subdomain and the fundamental solution is not able to correctly approximate the solution. On the other hand, in the case of the BDIE, which is non-local, the fundamental solution is able to capture the flow physics through the whole domain. Thus the results show that the SDBDIE behaves like a finite-element or finite-volume approach, which are also local and governed by sparse system matrices.

Table 2 shows the influence of mesh compression on the accuracy of the results. We can observe that for low Peclet number values an uncompressed ($R = 1$) mesh yields best results, while the accuracy deteriorates for high compression ratios. This behaviour is expected, as for a low Peclet number the solution profile resembles a linear function, thus equidistant mesh elements are the most appropriate choice. However, for very large Peclet number values, the solution has a very steep gradient at $x = b$, thus mesh compression towards the area of high gradients is necessary to achieve accurate results and the uncompressed mesh yields inaccurate results. This is clearly observed when looking at $\nu \geq 10^3$ results in Table 2.

5.4. Test case IV

Let us consider a 3D case where the domain is $\Omega = [1, 2]^3$ and ν a parameter. The coefficient varies linearly with location and time as $\alpha = t + x + y + z$. The fluid velocity varies with location and increases with increasing parameter; $\vec{v} = \nu(yz, xz, xy)$. The sources are described by $f = \nu(t + x + y + z)^{\nu-1}(1 + (-3 + yz + x(y + z))\nu)$. Dirichlet boundary conditions are considered on the left and right walls, $u(1, y, z, t) = (1 + t + y + z)^\nu$ and $u(2, y, z, t) = (2 + t + y + z)^\nu$. Neumann boundary conditions are applied on the front and back walls, $q(x, 1, z, t) = -\nu(1 + t + x + z)^{\nu-1}$ and $q(x, 2, z, t) = \nu(2 + t + x + z)^{\nu-1}$ as well as on the top and bottom walls $q(x, y, 1, t) = -\nu(1 + t + x + y)^{\nu-1}$ and

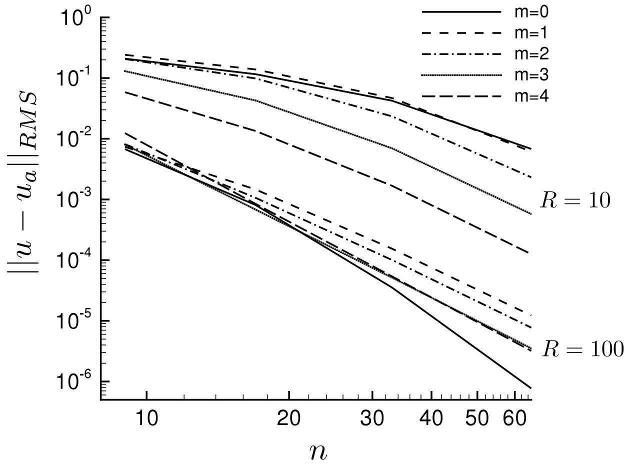


Fig. 7. Test case III — RMS versus n calculated using BDIE for $\nu = 1000$ using two different mesh concentration ratios, $R = 10$ and $R = 100$.

Table 1
Comparison of end node flux value $du/du|_{x=b}$. A 17 node mesh was used with $R = 100$. BDIE and SDBDIE solutions are compared with analytical values.

ν	$m = 0$	$m = 1$	$m = 2$	$m = 3$	$m = 4$
Analytical solution of $du/du|_{x=b}$					
10^0	1.582	1	0.635	0.3997	0.247
10^1	10.0005	5.0049	2.517	1.28	0.661
10^2	100	50	25	12.5	6.25
10^3	1000	500	250	125	62.5
10^4	10000	5000	2500	1250	625
10^5	100000	50000	25000	12500	6250
10^6	1000000	500000	250000	125000	62500
BDIE, 17 nodes mesh with $R = 100$					
10^0	1.578	1.0000001	0.634	0.393	0.233
10^1	9.97	4.96	2.45	1.22	0.612
10^2	99.94	49.95	24.96	12.47	6.24
10^3	999.19	499.27	249.73	124.92	62.45
10^4	10000.003	5000.3	2501.2	1249.5	624.1
10^5	100007.4	50000	25000	12500	6250.1
10^6	*2068237*	*623279*	257233	125065	62500.012
SDBDIE, 17 nodes mesh with $R = 100$					
10^0	1.575	0.999999	0.633	0.390	0.228
10^1	9.94	4.92	2.41	1.18	0.58
10^2	100.0	49.94	24.92	12.42	6.18
10^3	*943.7*	497.8	250.1	124.9	62.33

Table 2
Comparison of end node flux value $du/du|_{x=b}$ in the case of $m = 4$ simulated with BDIE using a 9 node mesh and different compression ratios R.

ν	Analytical	$R = 1$	$R = 10$	$R = 100$
10^0	0.247	0.245	0.226	0.193
10^1	0.66	0.65	0.59	0.52
10^2	6.25	5.98	6.23	5.4
10^3	62.5	37	59	61.7
10^4	625	336	475	620
10^5	6250	5168	6183	6251
10^6	62500	62738	62501	62500

$q(x, y, 2, t) = \nu(2 + t + x + y)^{\nu-1}$. At time $t = 0$, the field is initialized to $u = (x + y + z)^\nu$. The analytical solution of this test case is $u_a = (t + x + y + z)^\nu$.

To solve this test case we used a mesh with only one element (and 27 nodes) and meshes with 2^3, 4^3 and 8^3 elements with 125, 729 and 4913 nodes. All mesh elements of each mesh have the same size and the same cubical shape. Two time step sizes were considered; $\Delta t = 0.1$ and $\Delta t = 0.01$.

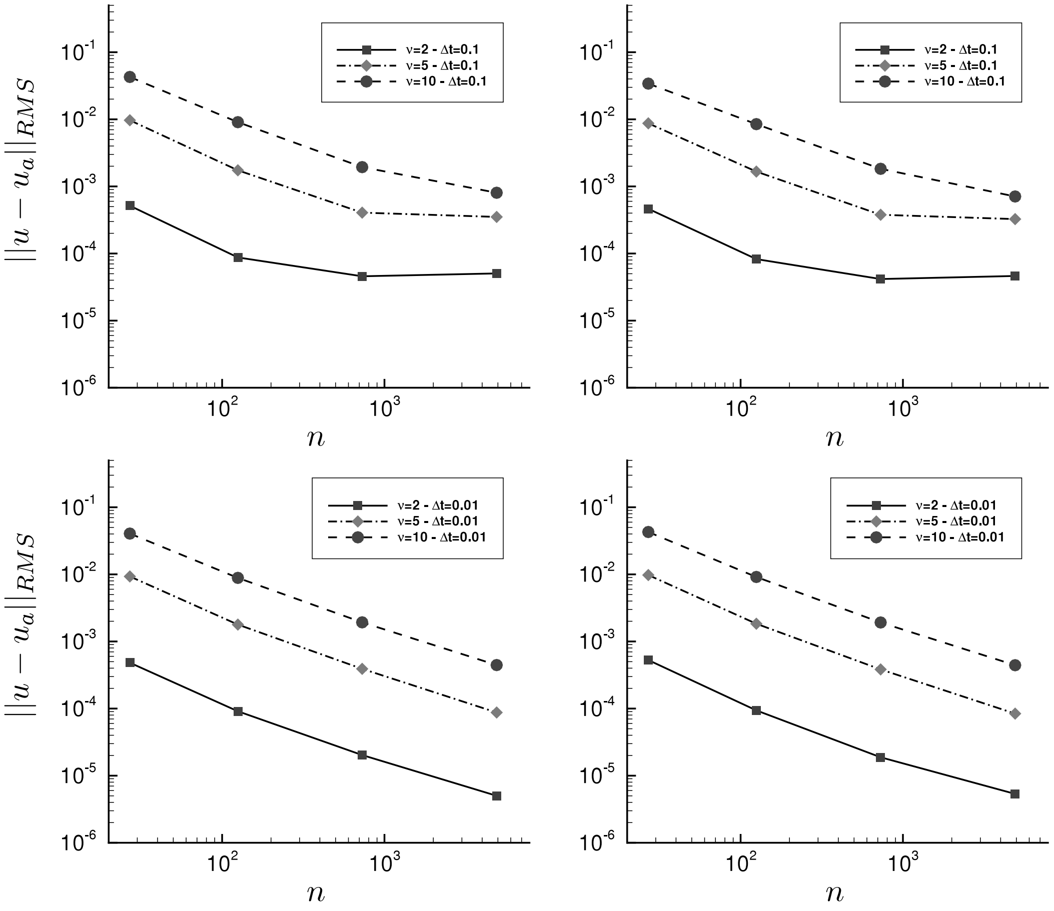


Fig. 8. Test case IV — RMS versus *n* calculated using BDIE (left) and SBDIE (right) using time step of $\Delta t = 0.1$ (top) and $\Delta t = 0.01$ (bottom).

The test case is designed in such a way, that the velocity of the flow is proportional to the value of the parameter ν. Also, the solution field has an exponential dependence on the parameter ν. Thus, for large values of the parameter, the flow is fast and the solution features exponential growth based on a large exponent. Fig. 8 shows RMS errors for the solutions of the test case with $\nu = 2$, $\nu = 5$ and $\nu = 10$.

We observe a deterioration of the solution accuracy with increasing values of the parameter ν. This is expected, as it is increasingly more difficult to capture the sharp solution profiles as the parameter increases. On the other hand, in the case of the short time step ($\Delta t = 0.01$) we observe that the methods keep good mesh convergence properties regardless of the value of the parameter ν. Thus, the order of the methods does not depend on ν. When using a longer time step ($\Delta t = 0.1$), the convergence properties are lost, as the increase of mesh density does not contribute to an increase of accuracy, since the long time step limits the solution accuracy.

6. Conclusions

We derived an integral formulation of an unsteady convection–diffusion problem with sources and variable velocity field and variable diffusion coefficient. The main advantage of this formulation over previously proposed solutions is the fact that it does not include the gradient of the unknown function. Without the gradient, the problem is linear and can be, after discretization, solved by a single solution of a linear system of equations.

We proposed two three-dimensional discretization approaches for the proposed formulation. Both are based on the use of the boundary element method combined with domain integration. The first employs the boundary–domain integral equation method, which requires the calculation of domain integrals in full and yields a fully populated system of equations. The second approach is based on domain decomposition, treating each mesh element as a subdomain and employing the boundary–domain integral equation method for each individual subdomain. After the application of the compatibility conditions between subdomains a sparse over-determined system of equations is obtained.

The test cases show that in most cases both discretization approaches yield results of comparable accuracy. The mesh convergence of both methods is of second order. An analysis of the time step showed that in order to have good convergence a sufficiently short time step must be used. Only in extreme cases of very sharp solution profiles, the non-local nature of the boundary–domain integral equation method outperforms the local domain decomposition approach.

The proposed integral formulation will in the future be used in a turbulent flow solver for the solution of momentum transport equation, which features a spatially and time varying velocity field. Additionally, when turbulence is modelled with the eddy viscosity approach, it requires the use of turbulent viscosity, which varies in space and in time, making the proposed formulation an ideal choice.

References

[1] S. Renganathan, R.E. White, Semianalytical method of solution for solid phase diffusion in lithium ion battery electrodes: variable diffusion coefficient, J. Power Sources 196 (1) (2011) 442–448.

[2] M. Dehghan, Numerical solution of the three-dimensional advection-diffusion equation, Appl. Math. Comput. 150 (1) (2004) 5–19.

[3] J.A. Pudykiewicz, Numerical solution of the reaction advection diffusion equation on the sphere, J. Comput. Phys. 213 (1) (2006) 358–390.

[4] K. Sakai, I. Kimura, A numerical scheme based on a solution of nonlinear advection diffusion equations, J. Comput. Appl. Math. 173 (1) (2005) 39–55.

[5] M. Remešikova, Numerical solution of two-dimensional convection–diffusion-adsorption problems using an operator splitting scheme, Appl. Math. Comput. 184 (1) (2007) 116–130. International Conference on Computational Methods in Sciences and Engineering 2004 (ICCMSE-2004 (Special issue)).

[6] A. Kumar, D. Kumar Jaiswal, N. Kumar, Analytical solutions of one-dimensional advection–diffusion equation with variable coefficients in a finite domain, J. Earth Syst. Sci. 118 (5) (2009) 539–549.

[7] M.D. Mikhailov, On the solution of the heat equation with time dependent coefficient, Int. J. Heat Mass Transfer 18 (2) (1975) 344–345.

[8] W.T. Ang, J. Kusuma, D.L. Clements, A boundary element method for a second order elliptic partial differential equation with variable coefficients, Eng. Anal. Bound. Elem. 18 (4) (1996) 311–316.

[9] R. Grzhibovskis, S. Mikhailov, S. Rjasanow, Numerics of boundary-domain integral and integro-differential equations for BVP with variable coefficient in 3D, Comput. Mech. 51 (2013) 495–503.

[10] S.E. Mikhailov, Localized boundary-domain integral formulations for problems with variable coefficients, Eng. Anal. Bound. Elem. 26 (2002) 681–690.

[11] S.E. Mikhailov, I.S. Nakhova, Mesh-based numerical implementation of the localized boundary domain integral equation method to a variable-coefficient Neumann problem, J. Eng. Math. 51 (2005) 251–259.

[12] S.E. Mikhailov, Analysis of united boundary-domain integro-differential and integral equations for a mixed BVP with variable coefficient, Math. Methods Appl. Sci. 29 (2006) 715–739.

[13] M.A. AL-Jawary, L.C. Wrobel, Radial integration boundary integral and integro-differential equation methods for two-dimensional heat conduction problems with variable coefficients, Eng. Anal. Bound. Elem. 36 (2012) 685–695.

[14] M.A. AL-Jawary, L.C. Wrobel, Numerical solution of two-dimensional mixed problems with variable coefficients by the boundary-domain integral and integro-differential equation methods, Eng. Anal. Bound. Elem. 35 (2011) 1279–1287.

[15] M.A. AL-Jawary, J. Ravnik, L.C. Wrobel, L. Škerget, Boundary element formulations for numerical solution of two-dimensional diffusion problems with variable coefficients, Comput. Math. Appl. 64 (2012) 2695–2711.

[16] K. Yang, X.-W. Gao, Radial integration BEM for transient heat conduction problems, Eng. Anal. Bound. Elem. 34 (2010) 557–563.

[17] J. Ravnik, L. Škerget, A gradient free integral equation for diffusion–convection equation with variable coefficient and velocity, Eng. Anal. Bound. Elem. 37 (2013) 683–690.

[18] L. Škerget, M. Hriberšek, G. Kuhn, Computational fluid dynamics by boundary domain integral method, Int. J. Numer. Methods Eng. 46 (1999) 1291–1311.

[19] B.J. Driessen, J.L. Dohner, A finite element boundary element method for advection diffiusion problems with variable advective fields and infinite domains, Int. J. Heat Mass Transfer 44 (2001) 2183–2191.

[20] L.C. Wrobel, The Boundary Element Method, John Willey & Sons, LTD, 2002.

[21] M. Ramšak, L. Škerget, M. Hriberšek, Z. Žunič, A multidomain boundary element method for unsteady laminar flow using stream function vorticity equations, Eng. Anal. Bound. Elem. 29 (2005) 1–14.

[22] C.C. Paige, M.A. Saunders, LSQR: an algorithm for sparse linear equations and sparse least squares, ACM Trans. Math. Software 8 (1982) 43–71.

[23] Z.H. Qiu, L. Wrobel, H. Power, Numerical solution of convection–diffusion problems at high Peclet number using boundary elements, Int. J. Numer. Methods Eng. 41 (1998) 899–914.