East Asian Math. J.
Vol. 40 (2024), No. 1, pp. 109–118
http://dx.doi.org/10.7858/eamj.2024.009

NON-ITERATIVE DOMAIN DECOMPOSITION METHOD FOR THE CONVECTION–DIFFUSION EQUATIONS WITH NEUMANN BOUNDARY CONDITIONS

Younbae Jun

Abstract. This paper proposes a numerical method based on domain decomposition to find approximate solutions for one-dimensional convection–diffusion equations with Neumann boundary conditions. First, the equations are transformed into convection–diffusion equations with Dirichlet conditions. Second, the author introduces the Prediction/Correction Domain Decomposition (PCDD) method and estimates errors for the interface prediction scheme, interior scheme, and correction scheme using known error estimations. Finally, the author compares the PCDD algorithm with the fully explicit scheme (FES) and the fully implicit scheme (FIS) using three examples. In comparison to FES and FIS, the proposed PCDD algorithm demonstrates good results.

1. Introduction

The convection–diffusion equation is a partial differential equation that describes the transport of a quantity such as heat or mass through a medium under the influence of both diffusion and advection. Some examples of initial–boundary value problems for the convection–diffusion equation with Neumann boundary conditions are heat transfer in a rod, groundwater flow, and diffusion in a slab.

We consider the one dimensional convection–diffusion equation of the form

$$u_t + \beta u_x = \gamma u_{xx}, \text{ for } (x,t) \in [0,1] \times [0,T], \tag{1}$$

with the initial condition

$$u(x,0) = f(x), \text{ for } x \in [0,1], \tag{2}$$

and the Neumann boundary conditions

$$u_x(0,t) = g_0(t) \text{ and } u_x(1,t) = g_1(t), \text{ for } t \in [0,T], \tag{3}$$

Received December 11, 2023; Revised January 15, 2024; Accepted January 23, 2024.
2010 *Mathematics Subject Classification.* Primary 65M06; Secondary 65M55.
Key words and phrases. Convection–diffusion equation, Domain decomposition, Finite difference method, Neumann condition, Efficiency.
This research was supported by Kumoh National Institute of Technology(2021).

(pISSN 1226-6973, eISSN 2287-2833)

where β and γ are given constants.

In recent years, significant attention has been devoted in the literature to developing numerical schemes for convection-diffusion equations. Baźan [2] introduced a highly accurate Chebyshev pseudospectral collocation method for 1D convection-diffusion equations. Eharhardt and Mickens [7] derived a new nonstandard finite difference scheme, using the subequation method, for a class of convection-diffusion equations with constant coefficients. Mohebbi and Dehghan [9] applied a fourth-order compact finite difference approximation to discretize spatial derivatives of one-dimensional heat and advection-diffusion equations, using the cubic spline collocation method. Salkuyeh [10] considered the system of ordinary differential equations arising from discretizing the convection-diffusion equation with respect to the space variable. Most of these works focused on Dirichlet boundary conditions. In [5], authors developed an accurate fourth-order compact finite difference scheme for solving the convection-diffusion equation with Neumann boundary conditions.

On the other hand, with recent advancements in powerful parallel computational capabilities, developing efficient parallel algorithms has become a significant challenge in computational mathematics. Domain decomposition (DD) [6, 8, 11] is one efficient parallel technique. Chen and Yang [6] proposed a non-overlapping domain decomposition algorithm for the Crank-Nicolson scheme with implicit Galerkin finite element methods and explicit flux approximation. Jiang and Xu [8] applied a two-level additive Schwarz preconditioner with overlapping domain decomposition. Yang [11] worked on non-iterative parallel Schwarz algorithms based on overlapping domain decomposition. However, few studies have explored Neumann boundary conditions of parabolic partial differential equations under domain decomposition architecture.

In this paper, we propose a numerical scheme based on domain decomposition to find approximate solutions for the one-dimensional convection-diffusion equation with Neumann boundary conditions, demonstrating the stability and efficiency of the method. In Section 2, we propose a non-iterative DD algorithm and analyze its stability. In Section 3, we provide numerical experiments and discuss the efficiency of the method. We then conclude in Section 4.

2. Numerical scheme and Stability

In this section, we present our domain decomposition method for solving the one-dimensional convection-diffusion equation with Neumann boundary conditions (1)–(3) and analyze its stability. We employ a finite difference scheme to discretize both the partial differential equation and its domain.

The domain $[0,1] \times [0,T]$ is covered by a mesh of grid lines $x_i = i\Delta x$ and $t_n = n\Delta t$, where $\Delta x = \frac{1}{L} = h$ and $\Delta t = \frac{T}{N} = k$, $i = 0, \cdots, L$ and $n = 0, \cdots, N$ with the positive integers L and N. Let w_i^n be the approximation of $u(x_i, t_n)$. The finite difference operators for each derivative of the convection–diffusion

equation (1) at (x_i, t_n) are written as follows:

$$w_t^n = \frac{w_i^n - w_i^{n-1}}{\Delta t}, w_x^n = \frac{w_{i+1}^n - w_{i-1}^n}{2\Delta x}, \text{ and } w_{xx}^n = \frac{w_{i+1}^n - 2w_i^n + w_{i-1}^n}{(\Delta x)^2}.$$

The initial value problem with the Neumann boundary conditions (3) is denoted as the Neumann problem (1)–(3). In the Neumann problem, only the derivatives on the boundary are specified, and the values on the boundary are unknown. Therefore, the Neumann boundary condition is discretized in terms of fictitious points as follows:

$$\frac{w_1^n - w_{-1}^n}{2\Delta x} = u_x(x_0, t_n) = g_0(t_n) \text{ and } \frac{w_{L+1}^n - w_{L-1}^n}{2\Delta x} = u_x(x_L, t_n) = g_1(t_n),$$

and hence

$$w_{-1}^n = w_1^n - 2\Delta x \cdot g_0(t_n) \text{ and } w_{L+1}^n = w_{L-1}^n + 2\Delta x \cdot g_1(t_n). \tag{4}$$

Solving the convection-diffusion equation with Neumann boundary conditions can be accomplished easily using either the fully explicit scheme or the fully implicit scheme, without resorting to domain decomposition. However, these schemes come with certain limitations, either in terms of stability or efficiency. Let us take note of the existing finite difference representations and stability considerations [1, 3] for the one-dimensional convection-diffusion equation.

Remark 1. The fully explicit scheme (FES) is conditionally stable for $\lambda = \gamma\frac{\Delta t}{\Delta x^2} \leq \frac{1}{2}$:

$$\text{FES: } w_t^n + \beta w_x^{n-1} = \gamma w_{xx}^{n-1}. \tag{5}$$

Remark 2. The fully implicit scheme(FIS) is unconditionally stable:

$$\text{FIS: } w_t^n + \beta w_x^n = \gamma w_{xx}^n. \tag{6}$$

In this section and the following sections, we describe our domain decomposition method and analyze its stability and efficiency in comparison to those of FES and FIS. Initially, the spatial domain is decomposed into several non-overlapping sub-domains. Subsequently, the problem defined on each sub-domain is considered as an independent sub-problem that can be solved in parallel. The proposed method consists of three major steps: interface prediction, interior solver, and interface correction.

2.1. Interface prediction step: Dirichlet problem

To solve the decomposed sub-problems independently, it is necessary to estimate the values of u_x in advance at each interface point, and these values serve as the Neumann boundary conditions for each sub-problem.

Throughout this paper, we assume that all the functions associated with the Neumann problem (1)–(3) are continuous, and all partial derivatives are also continuous. Let

$$q = u_x,$$

then the convection–diffusion equation (1) can be written as

$$q_t + \beta q_x = \gamma q_{xx}, \text{ for } (x,t) \in [0,1] \times [0,T], \tag{7}$$

with the initial condition

$$q(x,0) = f_x(x), \text{ for } x \in [0,1], \tag{8}$$

and the Neumann boundary conditions (3) become the Dirichlet boundary conditions

$$q(0,t) = g_0(t) \text{ and } q(1,t) = g_1(t), \text{ for } t \in [0,T]. \tag{9}$$

Remark 3. The Neumann problem (1)–(3) of $u(x,t)$ can be converted into the Dirichlet problem (7)–(9) of $q(x,t)$ provided that $q = u_x$.

Suppose the entire domain is now decomposed into P sub-domains, with $H = 1/P$. Let q_i^n represent the approximation of $q(x_i, t_n)$. Since the central difference schemes for $q_{xx}(x,t)$ and $q_x(x,t)$ can be formulated as follows:

$$q_x(x,t) = \frac{q(x+H,t) - q(x-H,t)}{2H} + O(H^2)$$

and

$$q_{xx}(x,t) = \frac{q(x+H,t) - 2q(x,t) + q(x-H,t)}{H^2} + O(H^2),$$

we define the central finite difference operators $\hat{q}_x^n$ and $\hat{q}_{xx}^n$ at the interface points (x_i, t_n) as

$$\hat{q}_x^n = \frac{q_{i+LH}^n - q_{i-LH}^n}{2H} \text{ and } \hat{q}_{xx}^n = \frac{q_{i+LH}^n - 2q_i^n + q_{i-LH}^n}{H^2},$$

where q_{i+LH}^n and q_{i-LH}^n are the unknown values at the adjacent interface points.

Remark 4. It is evident that $\hat{q}_x^n = q_x + O(H^2)$ and $\hat{q}_{xx}^n = q_{xx} + O(H^2)$.

Now we solve the the Dirichlet problem (7)–(9) for $q(x,t)$ using the implicit scheme defined by

$$q_t^n + \beta \hat{q}_x^n = \gamma \hat{q}_{xx}^n. \tag{10}$$

We observe that the interface prediction scheme (10) for $q(x_i, t_n)$ is the fully implicit scheme, which is unconditionally stable. Subsequently, the values of $q(x_i, t_n)$ at the interfaces serve as the Neumann boundary conditions for each sub-problem.

2.2. Interior solver step: Neumann problem

Suppose the entire domain is decomposed into P sub-domains. We can then define P sub-problems with Neumann boundary conditions obtained in the prediction step. Once the values of u_x at the interfaces are predicted, we solve the Neumann problem (1)–(3) for $u(x,t)$ in each sub-domain. The approximated solutions w_i^n for each of the P individual Neumann sub-problems can be obtained using the fully implicit scheme

$$w_t^n + \beta w_x^n = \gamma w_{xx}^n$$

with the fictitious points (4) and the values computed in the prediction step. Throughout the interior solver steps, linear systems are generated and solved using the non-iterative Crout factorization method [4].

2.3. Interface correction step

Finally, the approximated solutions w_i^n at the interfaces are corrected using the fully implicit scheme

$$w_t^n + \beta w_x^n = \gamma w_{xx}^n.$$

In this step, we do not solve any linear system; instead, we explicitly compute and update the value of w_i^n with the precomputed values of w_{i+1}^n and w_{i-1}^n.

Subsequently, we repeat the interface prediction, interior solver, and the interface correction steps until the last time level. This entire process is referred to as the Prediction/Correction Domain Decomposition method (PCDD). We summarize the algorithm as follows.

Algorithm 2.1. (PCDD algorithm)

Step 1 (Interface Prediction): Predict $u_x(x_i, t_n)$ at the interfaces using Eq. (10)

$$q_t^n + \beta \hat{q}_x^n = \gamma \hat{q}_{xx}^n.$$

Step 2 (Interior): Solve interior linear systems to compute w_i^n values using Eq. (6)

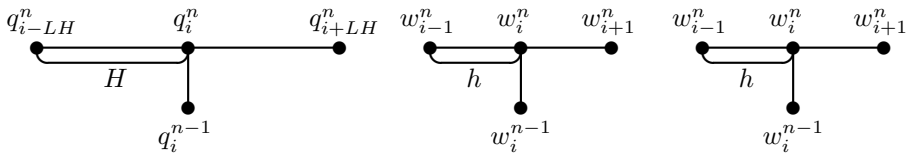
$$w_t^n + \beta w_x^n = \gamma w_{xx}^n.$$

Step 3 (Interface Correction): Correct w_i^n values at the interface using Eq. (6)

$$w_t^n + \beta w_x^n = \gamma w_{xx}^n.$$

Step 4: Repeat Step 1 through Step 3 until the last time level.

Note that w_{i+1}^n and w_{i-1}^n in Step 3 are computed values from Step 2. The backward stencils for the three steps of the Prediction/Correction Domain Decomposition algorithm are illustrated in Figure 1.



(a) Interface Prediction (b) Interior (c) Interface Correction

Figure 1. Stencils of the three steps of the PCDD method

We analyze the stability and truncation error of the new scheme. As evident from the algorithm, sequential fully implicit schemes are applied, resulting in the overall scheme being unconditionally stable.

Theorem 2.2. *The interface prediction scheme of the PCDD method is unconditionally stable, and the truncation error is* $O(H^2 + k)$.

Proof. It is well-known [1, 3] that the fully implicit scheme for solving the one-dimensional convection-diffusion equation with Neumann boundary conditions (1)–(3) is unconditionally stable. The scheme (10) of the PCDD method during the interface prediction of $q(x,t)$ involves central differences for the x-axis over the coarse mesh defined by the partition into sub-domains and backward differences for the t-axis. Thus, the interface prediction scheme $q_t^n + \beta \hat{q}x^n = \gamma \hat{q}_{xx}^n$ is a fully implicit scheme and is unconditionally stable. Since $q_t^n = q_t + O(k)$, $\hat{q}x^n = qx + O(H^2)$, and $\hat{q}_{xx}^n = qxx + O(H^2)$, we can clearly observe that the interface prediction scheme has errors $|q_i^n - q(x_i, t_n)| = O(H^2 + k)$. □

With the same argument as presented in the proof above, we can immediately derive the following theorem.

Theorem 2.3. *The interior scheme and the correction scheme of the PCDD method are unconditionally stable, and each truncation error is $O(h^2 + k)$.*

Proof. Both the interior scheme and the interface correction scheme of the PCDD method are $w_t^n + \beta w_x^n = \gamma w_{xx}^n$, which is an unconditionally stable fully implicit scheme. Thus, we immediately see that the truncation errors of those schemes are $|w_i^n - u(x_i, t_n)| = O(h^2 + k)$, since $w_t^n = w_t + O(k)$, $w_x^n = w_x + O(h^2)$, and $w_{xx}^n = w_{xx} + O(h^2)$. □

Note that each component of the PCDD scheme is unconditionally stable, and hence the stability of the overall PCDD method is unconditional.

3. Numerical experiments and Efficiency

In this section, we present numerical results for the new scheme regarding stability and efficiency. We compare the PCDD method with existing finite difference methods. It is worth noting that the PCDD method with $P = 1$ is equivalent to the fully implicit scheme (FIS), which is unconditionally stable and notably non-domain decomposition.

Problem 3.1. *Let us consider the following two model problems, MP1 and MP2:*

MP1: $u_t + u_x = u_{xx}$, *for* $(x,t) \in [0,1] \times [0,T]$
MP2: $u_t + 0.1u_x = 0.02u_{xx}$, *for* $(x,t) \in [0,1] \times [0,T]$

Note that the exact solutions are $u(x,t) = \exp(-t)\sin(x-t)$ for MP1 and $u(x,t) = \exp((5/2 - \sqrt{7}/2)x - 0.09t)$ for MP2, respectively. The initial conditions and the Neumann boundary conditions are derived from the exact solutions. The numerical experiments in this paper were carried out on a desktop computer with Intel(R) Core(TM) i7-8700 CPU with 8.0GB RAM.

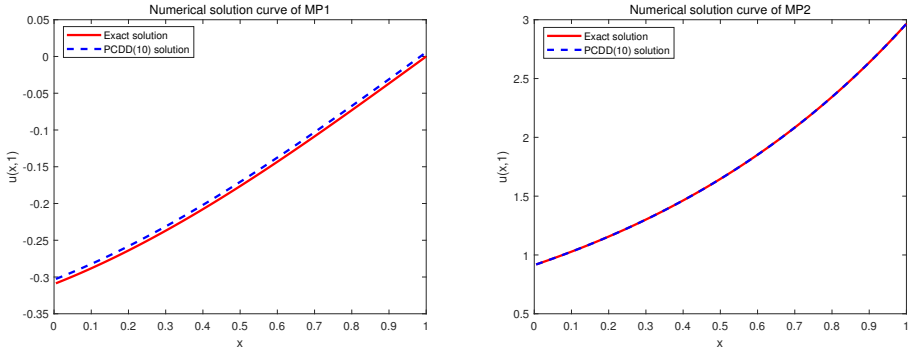
Firstly, we test the stability of the PCDD method using the two model problems. Table 1 shows the maximum error $||u^N - w^N||_\infty$ between the exact solutions and the numerical solutions for MP1 and MP2 at $t = 1$ with various λ defined by $\lambda = \gamma \frac{\Delta t}{(\Delta x)^2}$ ranging from 80 to 40000 for the three methods: FES,

FIS, and PCDD with 10 sub-domains. As observed in Table 1, FES is not convergent for those λ, whereas FIS and PCDD(10) are both unconditionally stable. Furthermore, the accuracy of PCDD(10) is highly comparable to FIS, which serves as a benchmark for this research.

Table 1. Maximum errors for various λ values in the model problems

	Δx	Δt	λ	FES	FIS	PCDD(10)
	1/2000	1/100	40000	∞	**0.590e-2**	**0.596e-2**
MP1	1/2000	1/400	10000	∞	0.147e-2	0.162e-2
	1/2000	1/1000	4000	∞	0.590e-3	0.738e-3
	1/2000	1/100	800	∞	0.107e-3	0.181e-3
MP2	1/2000	1/400	200	∞	0.269e-4	0.896e-4
	1/2000	1/1000	80	∞	**0.107e-4**	**0.570e-4**

Figure 2 depicts the exact solution curves and the numerical solution curves of PCDD(10) at $t = 1$ with $\Delta x = 1/2000$ and $\Delta t = 1/100$ for MP1, and $\Delta x = 1/2000$ and $\Delta t = 1/1000$ for MP2, respectively. It is evident from Figure 2 that PCDD(10) is a highly accurate method when Δt is small.



(a) Solution curves for MP1 (b) Solution curves for MP2

Figure 2. Numerical solution curves for MP1 and MP2

Secondly, we investigate the efficiency of the PCDD method for the Neumann problem. A common measurement of the efficiency of a parallel algorithm is the parallel execution time (PET). Since the algorithm is simulated with one processor in this numerical experiments, the true PET using P processors is roughly equivalent to the total CPU time (TCPU) divided by P.

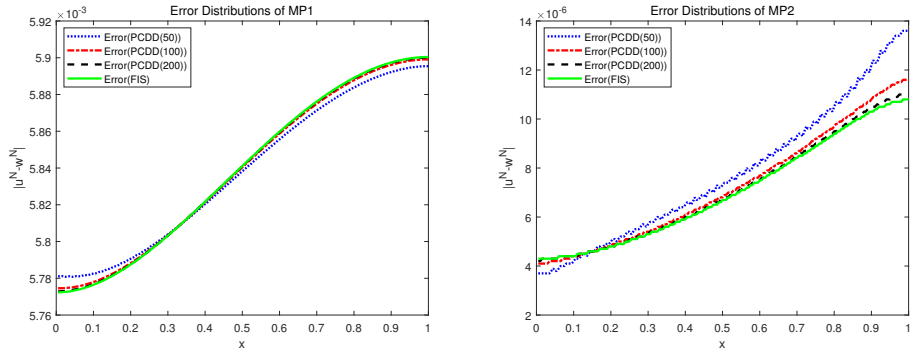
Table 2 presents the maximum errors at $t = 1$, total CPU time (TCPU), and parallel execution time (PET) of the PCDD method for various numbers of sub-domains P for MP1 and MP2, respectively. For MP1, we provide the numerical results for the discretization $\Delta x = 1/2000$, $\Delta t = 1/100$. And for MP2, $\Delta x = 1/2000$, $\Delta t = 1/1000$ were used. As observed in Table 2, the

various PCDD(P) methods are as accurate as FIS, and their parallel execution time is much less than the CPU time of FIS.

Table 2. Maximum errors and Speedups for various P of the PCDD(P) (TCPU=Total CPU time, PET=Parallel Execution Time)

	P	1(=FIS)	10	20	50	100	200
MP1	Error	**5.90e-3**	**5.96e-3**	5.86e-3	5.89e-3	5.89e-3	5.90e-3
	TCPU	0.0156	0.0313	0.0469	0.0938	0.2031	0.3750
	PET	0.0156	0.0031	0.0023	0.0019	0.0020	0.0019
MP2	Error	**1.07e-5**	**5.70e-5**	2.49e-5	1.36e-5	1.15e-5	1.10e-5
	TCPU	0.1094	0.2813	0.4688	1.0313	1.9375	3.7813
	PET	0.1094	0.0281	0.0234	0.0206	0.0194	0.0189

Figure 3 illustrates the absolute error distributions of the various PCDD(P) methods compared with the errors of the FIS method in the experiments shown in Table 2. It can be seen in Figure 3 that the error distribution of PCDD(P) approaches the distribution of FIS when P is large.



(a) Error distributions for MP1 (b) Error distributions for MP2

Figure 3. Error distributions of PCDD(P) with various P

Problem 3.2. *Let us consider the following model problem, MP3, which has a very small diffusion coefficient* $\gamma = 10^{-6}$*:*

MP3: $u_t + u_x = 10^{-6}u_{xx}$, *for* $(x,t) \in [0,1] \times [0,T]$

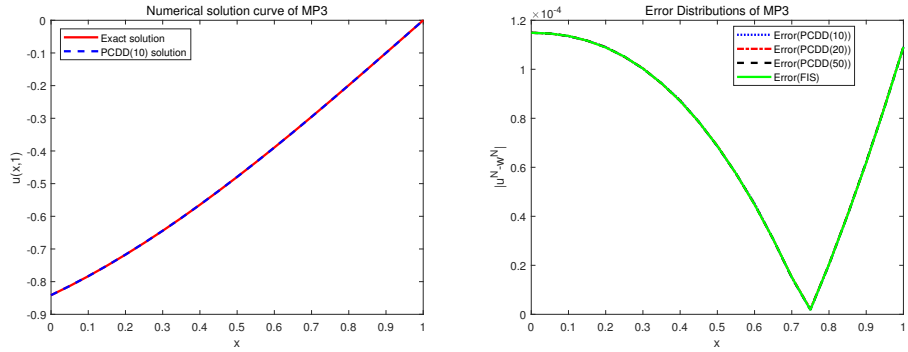
Note that the exact solution is $u(x,t) = \exp(-10^{-6}t)\sin(x-t)$. The initial condition and the Neumann boundary conditions are derived from the exact solutions.

Table 3 presents the maximum error $||u^N - w^N||_\infty$ between the exact solutions and the numerical solutions for MP3 at $t = 1$ with various λ values and various methods: FES, FIS, PCDD(10), PCDD(20, and PCDD(50). It is evident from Table 3 that the accuracy of PCDD(P) aligns with the observatoins in Table 1 and Table 2.

Table 3. Maximum errors for various λ values and various P of MP3

Δx	Δt	λ	FES	FIS	PCDD(10)	PCDD(20)	PCDD(50)
1/200	1/2000	2e-5	1.166e-4	1.174e-4	1.173e-4	1.174e-4	1.174e-4
1/200	1/4000	1e-5	5.748e-5	6.266e-5	6.265e-5	6.266e-5	6.267e-5
1/200	1/8000	5e-6	2.871e-5	3.531e-5	3.530e-5	3.530e-5	3.531e-5

Figure 4 illustrates the exact solution curves and the numerical solution curves of PCDD(10) at $t = 1$ with $\Delta x = 1/200$ and $\Delta t = 1/2000$ for MP3, along with the absolute error distributions of the various PCDD(P) methods compared with the errors of the FIS method in the experiments shown in Table 3. It can be seen in Figure 4 that the error distribution of PCDD(P) approaches the distribution of FIS when Δt is small.



(a) Solution curves for MP3 (b) Error distributions for MP3

Figure 4. Numerical solution curve and Error distributions for MP3

4. Conclusion

In this paper, we introduced a numerical method based on domain decomposition to find the numerical solutions of one-dimensional convection-diffusion equations with Neumann boundary conditions. The prediction scheme of the new method is of order $O(H^2+k)$. The interior scheme and the interface correction scheme are of order $O(h^2 + k)$. The PCDD(P) scheme is unconditionally stable and as accurate as the fully implicit scheme. Furthermore, the parallel execution time of the new scheme is much less than the CPU time of the fully implicit scheme. It is hoped that the PCDD method for one-dimensional convection-diffusion equations with Neumann boundary conditions can be easily applied to higher-dimensional Neumann convection-diffusion problems. Finally, we plan to study more complex problems in the future, such as nonlinear convection-diffusion equations.

Acknowledgement

The author expresses gratitude to the referees for their valuable comments.

References

[1] W.F. Ames, *Numerical methods for partial differential equations*, Academic Press, 1992.
[2] F.S.V. Baźan, *Chebyshev pseudospectral method for computing numerical solution of convection-diffusion equation*, Appl. Math. Comput. **200** (2008), 537–546.
[3] S. Biringen, *A note on the numerical stability of the convection-diffusion equation*, J. Comput. Appl. Math. **7** (1981), 17–20.
[4] R.L. Burden, J.D. Faires, A.M. Burden *Numerical Analysis*, Cengage Learning, 2014.
[5] H.H. Cao, L.B. Liu, Y. Zhang, S.M. Fu, *A fourth-order method of the convection-diffusion equations with Neumann boundary conditions*, Appl. Math. Comput. **217** (2011), 9133–9141.
[6] J. Chen, D. Yang, *Explicit/implicit and Crank-Nicolson domain decomposition methods for parabolic partial differential equations*, Comput. Math. Appl. **77** (2019), 1841–1863.
[7] M. Ehrhardt, R.E. Mickens, *A nonstandard finite difference scheme for convection-diffusion equations having constant coefficients*, Appl. Math. Comput. **219** (2013), 6591–6604.
[8] Y. Jiang, X. Xu, *Domain decomposition methods for space fractional partial differential equations*, J. Comput. Phys. **350** (2017), 573–589.
[9] A. Mohebbi, M. Dehghan, *High-order compact solution of the one-dimensional heat and advection-diffusion equations*, Appl. Math. Model. **34** (2010), 3071–3084.
[10] D.K. Salkuyeh, *On the finite difference approximation to the convection-diffusion equation*, Appl. Math. Comput. **179** (2006), 79–86.
[11] D. Yang, *Non-iterative parallel Schwarz algorithms based on overlapping domain decomposition for parabolic partial differential equations*, Math. Comp. **86** (2017), 2687–2718.

Younbae Jun
Department of Mathematics and Big Data Science, Kumoh National Institute of Technology, Gyeongbuk 39177, Republic of Korea
Email address: yjun@kumoh.ac.kr