Electronic Research Archive

http://www.aimspress.com/journal/era

ERA, 30(11): 4086–4107.
DOI: 10.3934/era.2022207
Received: 28 June 2022
Revised: 09 August 2022
Accepted: 11 August 2022
Published: 16 September 2022

Research article

A hybrid algorithm based on parareal and Schwarz waveform relaxation

Liping Yang[1] **and Hu Li**[2,*]

[1] Department of Mathematics, Zigong Vocational and Technical College, Zigong 643000, China
[2] School of Mathematics, Chengdu Normal University, Chengdu 611130, China

* **Correspondence:** Email: lihu_0826@163.com.

Abstract: In this paper, we present a hybrid algorithm based on parareal and Schwarz waveform relaxation (SWR) for solving time dependent partial differential equations. The parallelism can be simultaneously realized in the time direction by using a parareal and in the space direction via SWR. We give a convergence analysis for the hybrid algorithm for a 1D model problem, the reaction-diffusion equation. Weak scaling of the algorithm in terms of both the number of space subdomains and the number of paralleled time intervals were investigated via theoretical analysis and numerical experiments.

Keywords: Schwarz waveform relaxation (SWR); parareal algorithm; convergence analysis; weak scaling; domain decomposition

1. Introduction

For time dependent partial differential equations (PDEs), Schwarz waveform relaxation (SWR) is a powerful technique for realizing fast computation. It belongs to the widely used domain decomposition (DD) methods [1,2] but with a completely new implementation strategy: the classical strategy employs the alternating or parallel Schwarz method to the elliptic PDEs which result from semi-discretization of the original PDEs in time [3–5]; but, in the SWR framework one directly decomposes the space-time domain and solve each subproblem simultaneously or alternately. The main features (or say merits) of the SWR algorithms are able to treat different subproblems numerically differently with an adapted procedure for each subdomain. One can refer to [6–13] for more details about the SWR algorithm.

The aforementioned SWR algorithm realizes parallelism across space. Next, we introduce another efficient parallel algorithm, namely 'parareal', which realizes parallelism across time steps. This method was invented by Lions, Maday and Turinici in [14] as a numerical method to solve evolution problems in parallel. The name was chosen to indicate that the algorithm is well suited for parallel real time computations of evolution problems whose solution cannot be obtained in real time using one

processor only. The method approximates successfully the solution later in time before having fully accurate approximations from earlier times. Successful applications of this method include power system simulations [15, 16], differential algebraic equations [17], complex chemical reactions [18, 19], optimal control problems [20–23] and Volterra integro-differential equations [24–26]. Convergence analysis of the parareal algorithm can be found in [27–30].

It is natural to combine a parareal with other algorithms to formulate a hybrid algorithm with higher parallelism. For example, in [31–39] the classical waveform relaxation (WR) algorithm [40] is embedded into the parareal framework to reduce the computational cost. The authors proposed the "parareal-WR" algorithm and defined it by using the k_0-iteration WR algorithm as the fine propagator for the parareal algorithm. The algorithm is analyzed at a continuous level and superlinear and linear convergence were proved on short and long time intervals, respectively. In this paper we combine the SWR technique and the parareal algorithm to construct a hybrid algorithm, namely, Para-SWR, for time dependent PDEs, which can realize parallelism both in space and time directions. Combining a parareal with SWR has already been addressed in recent years; see, e.g., [41–43]. In those previous works, the SWR algorithm was used as the so-called fine propagator for the parareal algorithm and this leads to inner-outer iterations. This strategy is different from ours, since in our Para-SWR algorithm we use a single iteration of the parareal as a time integrator for the SWR algorithm. This actually leads to a special discrete SWR algorithm which can be used in parallel in terms of both space and time.

Here, we focus on analyzing the Para-SWR algorithm at the discrete level and deriving suitable conditions for convergence. We also investigate how the convergence rate of the Para-SWR algorithm behaves with respect to the number of subdomains and the number of parallelled time intervals. It is inspiring to report that the proposed Para-SWR algorithm posses the weak scaling property, which implies that increasing the number of subdomains and the number of parallelled time intervals has no (or slight) influence on the convergence rate. An algorithm scales weakly if it can solve a larger problem in reasonable time by increasing the number of processors.

The remainder of this paper is organized as follows. In Section 2, we describe the Para-SWR algorithm investigated in this paper. In Section 3, we analyze the Para-SWR algorithm and present suitable convergence conditions. Weak scaling of the algorithm is also investigated. Section 4 provides some numerical results to validate our theoretical predictions. We finish this paper in Section 5 with some conclusion remarks.

2. The Para-SWR hybrid algorithm

We consider the following 1-D linear reaction diffusion equation as the model problem

$$\begin{cases} \partial_t u - \nu^2 \partial_{xx} u + au = f(x,t), & (x,t) \in (0,L) \times \mathbb{R}^+, \\ u(0,t) = b_1(t), & t \geq 0, \\ u(L,t) = b_2(t), & t \geq 0, \\ u(x,0) = u_0(x), & x \in [0,L], \end{cases} \tag{2.1}$$

where a and $\nu(\neq 0)$ are constants. Without loss of generality, we assume $a > 0$ since otherwise we can make a change of variables $v = ue^{-\tau t}$ with $\tau + a > 0$ which leads to (2.1) with a positive reaction coefficient. We first introduce the SWR algorithm at a continuous level to (2.1). We let Δx be a suitable

small parameter and $L = [N \times (m-1)+2] \times \Delta x$ with some integers $N(\geq 2)$ and $m(\geq 2)$. We then decompose the space domain $[0, L]$ into N subdomains

$$\Omega_j := [L_j, R_j], L_j = (j-1)(m-1)\Delta x, R_j = (j(m-1)+2)\Delta x, j = 1, 2, \ldots, N. \tag{2.2}$$

In this situation, the overlap size between every two adjacent subdomains is $2\Delta x$. Then, the N-subdomain SWR algorithm for (2.1) can be written as

$$\begin{aligned}
x \in \Omega_1 : &\begin{cases} \partial_t u_1^k - \nu^2 \partial_{xx} u_1^k + a u_1^k = f(x,t), \\ u_1^k(0,t) = b_1(t), \\ (\partial_x + p) u_1^k(R_1,t) = (\partial_x + p) u_2^{k-1}(R_1,t), \\ u_1^k(x,0) = u_0(x), \end{cases} \\
x \in \Omega_j : &\begin{cases} \partial_t u_j^k - \nu^2 \partial_{xx} u_j^k + a u_j^k = f(x,t), \\ (\partial_x - p) u_j^k(L_j,t) = (\partial_x - p) u_{j-1}^{k-1}(L_j,t), \\ (\partial_x + p) u_j^k(R_j,t) = (\partial_x + p) u_{j+1}^{k-1}(R_j,t), \\ u_j^k(x,0) = u_0(x), \end{cases} \quad j = 2, 3, \ldots, N-1, \\
x \in \Omega_N : &\begin{cases} \partial_t u_N^k - \nu^2 \partial_{xx} u_N^k + a u_N^k = f(x,t), \\ (\partial_x - p) u_N^k(L_N,t) = (\partial_x - p) u_{N-1}^{k-1}(L_N,t), \\ u_N^k(0,t) = b_2(t), \\ u_N^k(x,0) = u_0(x), \end{cases}
\end{aligned} \tag{2.3}$$

where p is a free parameter. Note that, the transmission condition

$$(\partial_x - p) u_j^k(L_j,t) = (\partial_x - p) u_{j-1}^{k-1}(L_j,t),\ (\partial_x + p) u_j^k(R_j,t) = (\partial_x + p) u_{j+1}^{k-1}(R_j,t)$$

can be equivalently rewritten as

$$\left(\frac{\partial_x}{p} - 1\right) u_j^k(L_j,t) = \left(\frac{\partial_x}{p} - 1\right) u_{j-1}^{k-1}(L_j,t),\ \left(\frac{\partial_x}{p} + 1\right) u_j^k(R_j,t) = \left(\frac{\partial_x}{p} + 1\right) u_{j+1}^{k-1}(R_j,t).$$

Hence, by letting $p \to +\infty$ we get

$$u_j^k(L_j,t) = u_{j-1}^{k-1}(L_j,t),\ u_j^k(R_j,t) = u_{j+1}^{k-1}(R_j,t).$$

This is the so-called Dirichlet transmission condition and the corresponding algorithm is called classical SWR algorithm (see [9]). We next use the central finite difference method with mesh size Δx to discretize the SWR algorithm (2.3) and this leads to the following semi-discrete SWR algorithm:

$$\begin{cases} \partial_t v_j^k(i,t) - \nu^2 \frac{v_j^k(i-1,t) - 2v_j^k(i,t) + v_j^k(i+1,t)}{\Delta x^2} + a v_j^k(i,t) = f(x_i,t), \\ \frac{v_j^k((j-1)(m-1)+1,t) - v_j^k((j-1)(m-1),t)}{\Delta x} - p v_j^k((j-1)(m-1),t) = \\ \frac{v_{j-1}^{k-1}((j-1)(m-1)+1,t) - v_{j-1}^{k-1}((j-1)(m-1),t)}{\Delta x} - p v_{j-1}^{k-1}((j-1)(m-1),t), \\ \frac{v_j^k(j(m-1)+2,t) - v_j^k(j(m-1)+1,t)}{\Delta x} + p v_j^k(j(m-1)+2,t) = \\ \frac{v_{j+1}^{k-1}(j(m-1)+2,t) - v_{j+1}^{k-1}(j(m-1)+1,t)}{\Delta x} + p v_{j+1}^{k-1}(j(m-1)+2,t), \\ v_j^k(i,0) = u_0(i\Delta x), \end{cases} \tag{2.4a}$$

where i varies from $(j-1)(m-1)+1$ to $j(m-1)+1$ and $j = 2, 3, \ldots, N-1$. Similarly, for the left subdomain Ω_1 and the right subdomain Ω_N

$$
\begin{aligned}
&i \text{ from } 1 \text{ to } m:\\
&\begin{cases}
\partial_t v_1^k(i,t) - \nu^2 \frac{v_1^k(i-1,t)-2v_1^k(i,t)+v_1^k(i+1,t)}{\Delta x^2} + a v_1^k(i,t) = f(x_i,t),\\
v_1^k(0,t) = b_0(t),\\
\frac{v_1^k(m+1,t)-v_1^k(m,t)}{\Delta x} + p v_1^k(m+1,t) = \frac{v_2^{k-1}(m+1,t)-v_2^{k-1}(m,t)}{\Delta x} + p v_2^{k-1}(m+1,t),\\
v_1^k(i,0) = u_0(i\Delta x),
\end{cases}\\
&i \text{ from } (N-1)(m-1)+1 \text{ to } N(m-1)+1: \qquad (2.4b)\\
&\begin{cases}
\partial_t v_N^k(i,t) - \nu^2 \frac{v_N^k(i-1,t)-2v_N^k(i,t)+v_N^k(i+1,t)}{\Delta x^2} + a v_N^k(i,t) = f(x_i,t),\\
\frac{v_N^k((N-1)(m-1)+1,t)-v_N^k((N-1)(m-1),t)}{\Delta x} - p v_N^k((N-1)(m-1),t) =\\
\frac{v_{N-1}^{k-1}((N-1)(m-1)+1,t)-v_{N-1}^{k-1}((N-1)(m-1),t)}{\Delta x} - p v_{N-1}^{k-1}((N-1)(m-1),t),\\
v_N^k(N(m-1)+2,t) = b_1(t),\\
v_N^k(i,0) = u_0(i\Delta x).
\end{cases}
\end{aligned}
$$

Note that, $v_j^k(i,t) = u_j^k(i\Delta x,t) + O(\Delta x^2)$ if $i\Delta x \in (L_j, R_j)$ and $v_j^k(i,t) = u_j^k(i\Delta x,t) + O(\Delta x)$ if $i\Delta x = L_j$ or $i\Delta x = R_j$. However, upon convergence it holds that $v_j^\infty(i,t) = u_j^\infty(i\Delta x,t) + O(\Delta x^2) = u_j(i\Delta x,t) + O(\Delta x^2)$ for $i\Delta x \in [L_j, R_j]$, that is the discretization of the artificial boundary conditions does not effect the accuracy of the converged solutions [44]. Define

$$
f_+(t) = \begin{pmatrix} f(x_1,t) + \frac{\nu^2}{\Delta x^2} b_0(t) \\ f(x_2,t) \\ \vdots \\ f(x_m,t) \end{pmatrix}, f_j(t) = \begin{pmatrix} f(x_{(j-1)(m-1)+1},t) \\ f(x_{(j-1)(m-1)+2},t) \\ \vdots \\ f(x_{j(m-1)+1},t) \end{pmatrix}, f_-(t) = \begin{pmatrix} f(x_{(N-1)(m-1)+1},t) \\ \vdots \\ f(x_{N(m-1)},t) \\ f(x_{N(m-1)+1},t) + \frac{\nu^2}{\Delta x^2} b_1(t) \end{pmatrix},
$$

$$
q = \frac{1}{\Delta x p + 1}, \quad A = \frac{\nu^2}{\Delta x^2} \begin{pmatrix} -2+q & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2+q \end{pmatrix} - aI, V_j^k(t) = \begin{pmatrix} v_j^k((j-1)(m-1)+1,t) \\ v_j^k((j-1)(m-1)+2,t) \\ \vdots \\ v_j^k(j(m-1)+1,t) \end{pmatrix},
$$

$$
A_+ = \frac{\nu^2}{\Delta x^2} \begin{pmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2+q \end{pmatrix} - aI, \quad A_- = \frac{\nu^2}{\Delta x^2} \begin{pmatrix} -2+q & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{pmatrix} - aI,
$$

$$
B_+ = \frac{\nu^2}{\Delta x^2} \begin{pmatrix} 0 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 0 \\ -q & 1 & 0 & \cdots & 0 \end{pmatrix}, \quad B_- = \frac{\nu^2}{\Delta x^2} \begin{pmatrix} 0 & \cdots & 0 & 1 & -q \\ 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & 0 & 0 \\ 0 & \cdots & 0 & 0 & 0 \end{pmatrix}.
$$

Then the semi-discrete SWR algorithm (2.4) can be equivalently rewritten as

$$\begin{aligned}&\partial_t V_j^k(t) = AV_j^k(t) + B_- V_{j-1}^{k-1}(t) + B_+ V_{j+1}^{k-1}(t) + f_j(t),\ j = 2, 3, \ldots, N-1,\\ &\partial_t V_1^k(t) = A_+ V_1^k(t) + B_+ V_2^{k-1}(t) + f_+(t),\ \partial_t V_N^k(t) = A_- V_N^k(t) + B_- V_{N-1}^{k-1}(t) + f_-(t).\end{aligned} \tag{2.5}$$

Clearly, this can be written in a more concise form

$$\partial_t \mathcal{V}^k(t) = \mathcal{A}\mathcal{V}^k(t) + \mathcal{B}\mathcal{V}^{k-1}(t) + \tilde{f}(t), \tag{2.6}$$

where

$$\tilde{f}(t) = \begin{pmatrix} f_+(t) \\ f_2(t) \\ \vdots \\ f_{N-1}(t) \\ f_-(t) \end{pmatrix}, \mathcal{V}^k(t) = \begin{pmatrix} V_1^k(t) \\ V_2^k(t) \\ \vdots \\ V_{N-1}^k(t) \\ V_N^k(t) \end{pmatrix}, \mathcal{A} = \begin{pmatrix} A_+ & & & & \\ & A & & & \\ & & \ddots & & \\ & & & A & \\ & & & & A_- \end{pmatrix}, \mathcal{B} = \begin{pmatrix} O & B_+ & & & \\ B_- & O & B_+ & & \\ & \ddots & \ddots & \ddots & \\ & & B_- & O & B_+ \\ & & & B_- & O \end{pmatrix}. \tag{2.7}$$

We next introduce the parareal algorithm. For a general system of ordinary differential equations

$$\begin{cases} v'(t) = F(t, v(t)), & t \in [0, T], \\ v(0) = v_0, & t = 0, \end{cases} \tag{2.8}$$

the parareal algorithm can be described as follows. First, the whole time interval $[0, T]$ is divided into N_t time slices $[T_n, T_{n+1}]$, $n = 0, 1, \ldots, N_t - 1$. We suppose that all time slices are of uniform size, i.e., $T_{n+1} - T_n = \Delta T = \frac{T}{N_t}$. Second, each large time slice $[T_n, T_{n+1}]$ is divided into $J(\geq 2)$ small time slices. Then, two numerical propagators $\mathcal{G}$ and $\mathcal{F}$ are assigned to the coarse and fine time grids, respectively. We designate by the symbol $\ominus$ the time-sequential implementation, and by the symbol $\oplus$ the time parallel implementation. With the aforementioned introduction, the concrete parareal algorithm (for Problem (2.8)) proposed by Lions, Maday and Turinici [14] is given in Algorithm 2.1.

Algorithm 2.1 Parareal Algorithm.

$\ominus$ **Initialization**: Perform sequential computation $v_{n+1}^0 = \mathcal{G}(T_n, v_n^0, \Delta T)$ for $n = 0, 1, \ldots, N_t - 1$;
For $k = 0, 1, \ldots$

$\oplus$ **Step 1** In each subinterval $[T_n, T_{n+1}]$, compute $v_{n+\frac{j+1}{J}} = \mathcal{F}\left(T_{n+\frac{j}{J}}, v_{n+\frac{j}{J}}, \frac{\Delta T}{J}\right)$ with the initial value $v_n = v_n^k$, where $T_{n+\frac{j}{J}} = T_n + \frac{j\Delta T}{J}$ and $j = 0, 1, \ldots, J-1$.

$\ominus$ **Step 2** Perform sequential corrections

$$v_{n+1}^{k+1} = \mathcal{G}\left(T_n, v_n^{k+1}, \Delta T\right) + v_{n+1} - \mathcal{G}\left(T_n, v_n^k, \Delta T\right), v_0^{k+1} = v_0,\ n = 0, 1, \ldots, N_t - 1.$$

$\ominus$ **Step 3** If $\left\{v_n^{k+1}\right\}_{n=1}^{N_t}$ satisfy the stopping criteria, terminate the iteration; otherwise go to **Step 1**.

Clearly, the argument $\tilde{v}_{n+1}$ can be written as $\tilde{v}_{n+1} = \mathcal{F}^J\left(T_n, v_n^k, \Delta t\right)$ and therefore Algorithm 2.1 can be written compactly as

$$v_{n+1}^{k+1} = \mathcal{G}\left(T_n, v_n^{k+1}, \Delta T\right) + \mathcal{F}^J\left(T_n, v_n^k, \Delta t\right) - \mathcal{G}\left(T_n, v_n^k, \Delta T\right), \tag{2.9}$$

where $\Delta t = \frac{\Delta T}{J}$ and $\mathcal{F}^J\left(T_n, v_n^k, \Delta t\right)$ denotes a value obtained by running J steps of the fine propagator $\mathcal{F}$ with the initial value v_n^k and the fine step size Δt.

Our Para-SWR algorithm consists of applying the one-iteration parareal algorithm to System (2.6), which is derived from the semi-discrete SWR algorithm. We use the backward Euler method as both the coarse and the fine propagators. On each time interval $[T_n, T_{n+1}]$, we need to provide a seed value to start up the parallel computation and it is natural to use the $\mathcal{V}_n^{k-1}$ obtained in the previous iteration as the seed. Then, the parallel computation in each subinterval $[T_n, T_{n+1}]$ is

$$\begin{aligned}
&\mathcal{G}-\text{propagator}: \frac{\mathbb{V}_{n+1}^k - \mathcal{V}_n^{k-1}}{\Delta T} = \mathcal{A}\mathbb{V}_{n+1}^k + \mathcal{B}\mathcal{V}_{n+1}^{k-1} + \tilde{f}(t_{n+1}),\\
&\mathcal{F}-\text{propagator}: \frac{\mathbb{V}_{n,\frac{j}{J}}^k - \mathbb{V}_{n,\frac{j-1}{J}}^k}{\Delta t} = \mathcal{A}\mathbb{V}_{n,\frac{j}{J}}^k + \mathcal{B}\mathbb{V}_{n,\frac{j}{J}}^{k-1} + \tilde{f}(t_n + j\Delta t),\ j = 1, 2, \ldots, J,
\end{aligned} \tag{2.10}$$

where $\mathbb{V}_{n,0}^k = \mathcal{V}_n^{k-1}$ and $\Delta T = J\Delta t$. From (2.10) we get

$$\begin{aligned}
\mathbb{V}_{n+1}^k &= (I - \Delta T\mathcal{A})^{-1}\left(\mathcal{V}_n^{k-1} + \Delta T\mathcal{B}\mathcal{V}_{n+1}^{k-1} + \Delta T\tilde{f}(t_{n+1})\right),\\
\mathbb{V}_{n,\frac{j}{J}}^k &= (I - \Delta t\mathcal{A})^{-1}\left(\mathbb{V}_{n,\frac{j-1}{J}}^k + \Delta t\mathcal{B}\mathbb{V}_{n,\frac{j}{J}}^{k-1} + \Delta t\tilde{f}(t_n + j\Delta t)\right),\ j = 1, 2, \ldots, J.
\end{aligned} \tag{2.11}$$

With the values $\mathbb{V}_{n+1}^k$ and $\mathbb{V}_{n,1}^k$, the serial correction is

$$\mathcal{V}_{n+1}^k = \mathbb{V}_{n+1}^k - \mathcal{G}\left(T_n, \mathcal{V}_n^k, \Delta T\right) + \mathbb{V}_{n,1}^k, \tag{2.12}$$

where the argument $\mathcal{G}\left(T_n, \mathcal{V}_n^k, \Delta T\right)$ denotes the value generated by the backward Euler method with the starting value $\mathcal{V}_n^k$, that is

$$\mathcal{G}\left(T_n, \mathcal{V}_n^k, \Delta T\right) = (I - \Delta T\mathcal{A})^{-1}\left(\mathcal{V}_n^k + \Delta T\mathcal{B}\mathcal{V}_{n+1}^{k-1} + \Delta T\tilde{f}(t_{n+1})\right). \tag{2.13}$$

It then follows by substituting (2.11) and (2.13) into (2.12) that

$$\mathcal{V}_{n+1}^k = (I - \Delta T\mathcal{A})^{-1}\left(\mathcal{V}_n^{k-1} - \mathcal{V}_n^k\right) + \mathbb{V}_{n,1}^k. \tag{2.14}$$

Remark 1. *The uniform formula* (2.6) *of the semi-discrete SWR algorithm leads to* (2.14) *which will greatly facilitate the theoretical analysis done in the next section. In practical computation, it is convenient to directly apply the parareal algorithm on each subdomain* $\Omega_j \times [0, T]$, $j = 1, 2, \ldots, N$.

3. Analysis of the Para-SWR algorithm

In this section, we focus on investigating the theoretical property of the Para-SWR algorithm. The analysis is divided into two parts, i.e., the convergence of the algorithm and the weak scaling of the algorithm. In the sequel, we assume $\Delta xp \geq 1$ and this implies $q \geq 0$.

3.1. Convergence analysis

To analyze the convergence of the Para-SWR iteration (2.14), for simplicity and without loss of generality we set $\tilde{f}(t) = 0$. Let

$$\bar{\mathcal{A}}_c = (I - \Delta T\mathcal{A})^{-1}, \bar{\mathcal{A}}_f = (I - \Delta t\mathcal{A})^{-1}, \tag{3.1}$$

and by using the relation $\mathbb{V}^k_{n,0} = \mathcal{V}^{k-1}_n$ we know from the second equality of (2.11) that

$$\mathbb{V}^k_{n,\frac{j}{J}} = \bar{\mathcal{A}}^j_f \mathcal{V}^{k-1}_n + \sum_{i=1}^{j} \Delta t \bar{\mathcal{A}}^i_f \mathcal{B} \mathbb{V}^{k-1}_{n,\frac{j-i+1}{J}}, \quad j = 1, 2, \ldots, J-1. \tag{3.2}$$

It then follows by substituting (3.2) into (2.14) that

$$\mathcal{V}^k_{n+1} = \bar{\mathcal{A}}_c \left(\mathcal{V}^k_n - \mathcal{V}^{k-1}_n\right) + \underbrace{\bar{\mathcal{A}}^J_f \mathcal{V}^{k-1}_n + \Delta t \sum_{j=1}^{J} \bar{\mathcal{A}}^j_f \mathcal{B} \mathbb{V}^{k-1}_{n,\frac{J-j+1}{J}}}_{\mathbb{V}^k_{n,1}}. \tag{3.3}$$

Define

$$\begin{aligned}
&\lambda_c = \left\|(I - \Delta T\mathcal{A})^{-1}\right\|_2, \lambda_f = \left\|(I - \Delta t\mathcal{A})^{-1}\right\|_2, \mu = \Delta t\|\mathcal{B}\|_2, \gamma = \left\|\bar{\mathcal{A}}^J_f - \bar{\mathcal{A}}_c\right\|_2, \\
&\mathbb{D} = \begin{pmatrix} \lambda_f \\ \lambda_f^2 \\ \lambda_f^3 \\ \vdots \\ \lambda_f^J \end{pmatrix}, \mathbb{E} = \begin{pmatrix} \lambda_f & & & & \\ \lambda_f^2 & \lambda_f & & & \\ \lambda_f^3 & \lambda_f^2 & \lambda_f & & \\ \vdots & \ddots & \ddots & \ddots & \\ \lambda_f^J & \lambda_f^{J-1} & \lambda_f^{J-2} & \cdots & \lambda_f \end{pmatrix}, \epsilon^k_n = \begin{pmatrix} \left\|\mathbb{V}^k_{n,\frac{1}{J}}\right\|_2 \\ \left\|\mathbb{V}^k_{n,\frac{2}{J}}\right\|_2 \\ \vdots \\ \left\|\mathbb{V}^k_{n,\frac{J}{J}}\right\|_2 \end{pmatrix}.
\end{aligned} \tag{3.4}$$

Under the assumptions $q \geq 0$ and $a > 0$, it is easy to know that $\lambda_c < \lambda_f < 1$. From the second equality of (3.2) we get

$$\epsilon^k_n \leq \mathbb{D}\left\|\mathcal{V}^{k-1}_n\right\|_2 + \mu\mathbb{E}\epsilon^{k-1}_n. \tag{3.5}$$

From (3.3) we have

$$\left\|\mathcal{V}^k_{n+1}\right\|_2 \leq \lambda_c \left\|\mathcal{V}^k_n\right\|_2 + \gamma \left\|\mathcal{V}^{k-1}_n\right\|_2 + \mu\mathbb{E}_1\epsilon^{k-1}_n, \tag{3.6}$$

where $\mathbb{E}_1$ denotes the last row of the matrix $\mathbb{E}$. Define

$$\mathcal{V}^k = \max_{n\geq 1}\left\|\mathcal{V}^k_n\right\|_2, \epsilon^k_j = \max_{n\geq 0}\epsilon^k_{n,j}, \epsilon^k = \left(\epsilon^k_1, \epsilon^k_2, \ldots, \epsilon^k_J\right)^T. \tag{3.7}$$

Then from (3.5) and (3.6) it is easy to get

$$\begin{aligned}
&\mathcal{V}^k \leq \frac{1}{1-\lambda_c}\left(\gamma\mathcal{V}^{k-1} + \mu\mathbb{E}_1\epsilon^{k-1}\right), \\
&\epsilon^k \leq \mathbb{D}\mathcal{V}^{k-1} + \mu\mathbb{E}\epsilon^{k-1}.
\end{aligned} \tag{3.8}$$

Define

$$\beta = \frac{\gamma}{1-\lambda_c}, \eta = \frac{\mu}{1-\lambda_c}, \mathbb{M} = \begin{pmatrix} \beta & \eta\mathbb{E}_1 \\ \mathbb{D} & \mu\mathbb{E} \end{pmatrix} = \begin{pmatrix} \beta & \eta\lambda_f^J & \eta\lambda_f^{J-1} & \eta\lambda_f^{J-2} & \cdots & \eta\lambda_f \\ \lambda_f & \mu\lambda_f & & & & \\ \lambda_f^2 & \mu\lambda_f^2 & \mu\lambda_f & & & \\ \lambda_f^3 & \mu\lambda_f^3 & \mu\lambda_f^2 & \mu\lambda_f & & \\ \vdots & \vdots & \ddots & \ddots & \ddots & \\ \lambda_f^J & \mu\lambda_f^J & \mu\lambda_f^{J-1} & \mu\lambda_f^{J-2} & \cdots & \mu\lambda_f \end{pmatrix}. \tag{3.9}$$

Let the vectors $\{z^k\}$ be generated by the following recurrence relation

$$z^k = \mathbb{M}z^{k-1}, \tag{3.10}$$

with the initial vector $z^0 = \begin{pmatrix} \mathcal{V}^0 \\ \epsilon^0 \end{pmatrix}$. Since the matrix $\mathbb{M}$ is nonnegative, it is clear that $\begin{pmatrix} \mathcal{V}^k \\ \epsilon^k \end{pmatrix} \leq z^k$ in the component sense for all $k \geq 1$. Thus, it is sufficient to investigate the convergence of the iteration (3.10). It is well known that the iteration converges to zero if and only if the spectral radius of $\mathbb{M}$ is less than one, i.e., $\rho(\mathbb{M}) < 1$. To analyze the spectral radius of $\mathbb{M}$, we need the following definitions and lemmas concerning nonnegative matrices.

Definiton 1. [45, page 95] *For $n \times n$ real matrices P, P_1 and P_2, the relation $P = P_1 - P_2$ is called a regular splitting of P if P_1 is nonsingular with $P_1^{-1} \geq 0$ and $P_2 \geq 0$.*

Definiton 2. [45, pages 53–54] *Let P be a $n \times n$ real matrix and $G(P_B)$ be a directed graph associated with the matrix P, which has n vertices and an edge from the vertex i to vertex j precisely when $P_{ij} > 0$. Then the matrix P is irreducible if and only if its associated graph $G(P)$ is strongly connected, that is for any two vertices i and j of $G(P)$, $G(P)$ contains a path from i to j.*

Lemma 1. [45, page 90] *Let $P = (p_{ij})$ be a $n \times n$ real matrix with $p_{ij} \leq 0$ for all $i \neq j$; then, the following statements are equivalent:*

1) P is nonsingular and $P^{-1} \geq 0$;

2) the diagonal entries of P are positive real numbers and letting P_D be the diagonal matrix whose diagonal entries d_{ii} are defined as

$$d_{ii} := p_{ii}, \quad 1 \leq i \leq n,$$

then the matrix $P_B := I - P_D^{-1}P$ is nonnegative and $\rho(P_B) < 1$.

Lemma 2. [45, page 51] *Let $P \in \mathbb{R}^{n\times n}$ and $P = P_1 - P_2$ be a regular splitting of P. Then P is nonsingular with $P^{-1} \geq 0$ if and only if $\rho(P_1^{-1}P_2) < 1$.*

Lemma 3. [45, page 50] *Let $P \in \mathbb{R}^{n\times n}$ be an irreducible non-negative matrix with the spectral radius $\rho(P)$. Then, the number $\rho(P)$ is a positive real number and an eigenvalue of the matrix P, called the Perron-Frobenius eigenvalue.*

Theorem 1. *Let $q \geq 0$, $a > 0$, $\mu\lambda_f < 1$ and $\beta < 1$. Then the Para-SWR algorithm based on the backward Euler method is convergent and the convergence rate can be bounded by $\rho(\mathbb{M})$, provided the following condition is satisfied*

$$\frac{\eta\lambda_f^J\left(1-\left[1-\mu\lambda_f\right]^J\right)}{\mu(1-\beta)(1-\mu\lambda_f)^J} < 1, \tag{3.11}$$

where λ_f and μ are defined by (3.4) *and β and η are defined by* (3.9).

Proof. It is sufficient to prove that Condition (3.11) implies $\rho(\mathbb{M}) < 1$. In fact, this is an "if and only if" condition. Let

$$D = \begin{pmatrix} 0 & & & & \\ 1 & 0 & & & \\ & \ddots & \ddots & & \\ & & 1 & 0 & \\ & & & 1 & 0 \end{pmatrix}_{J\times J}, Q = \begin{pmatrix} 1 & & & & \\ \lambda_f & 1 & & & \\ \lambda_f^2 & \lambda_f & 1 & & \\ \vdots & \ddots & \ddots & \ddots & \\ \lambda_f^{J-1} & \cdots & \lambda_f^2 & \lambda_f & 1 \end{pmatrix}_{J\times J}. \tag{3.12}$$

It is easy to check that $Q = (I - \lambda_f D)^{-1}$. Let $X_1 = \eta\lambda_f \mathbb{E}_1^T$ and $X_2 = \lambda_f(I - \lambda_f D)\mathbb{D}$ and we have

$$\mathbb{M} = P_1^{-1}P_2, P_1 = \begin{pmatrix} 1 & \\ & I - \lambda_f D \end{pmatrix}, P_2 = \begin{pmatrix} \beta & X_1^T \\ X_2 & \mu\lambda_f I \end{pmatrix}. \tag{3.13}$$

Define

$$P := P_1 - P_2 = \begin{pmatrix} 1-\beta & -\eta\lambda_f^J & -\eta\lambda_f^{J-1} & \cdots & -\eta\lambda_f \\ -\lambda_f & 1-\mu\lambda_f & & & \\ 0 & -\lambda_f & 1-\mu\lambda_f & & \\ \vdots & & \ddots & \ddots & \\ 0 & & & -\lambda_f & 1-\mu\lambda_f \end{pmatrix}, P_D := \begin{pmatrix} 1-\beta & & & & \\ & 1-\mu\lambda_f & & & \\ & & & \ddots & \\ & & & & 1-\mu\lambda_f \end{pmatrix}.$$

Then we have

$$P_B := I - P_D^{-1}P = \begin{pmatrix} 0 & \frac{\eta\lambda_f^J}{1-\beta} & \frac{\eta\lambda_f^{J-1}}{1-\beta} & \cdots & \frac{\eta\lambda_f}{1-\beta} \\ \frac{\lambda_f}{1-\mu\lambda_f} & 0 & & & \\ & \frac{\lambda_f}{1-\mu\lambda_f} & 0 & & \\ & & \ddots & \ddots & \\ & & & \frac{\lambda_f}{1-\mu\lambda_f} & 0 \end{pmatrix}. \tag{3.14}$$

From Definition 1, we know that $P = P_1 - P_2$ is a regular splitting of the matrix P. Since $\mu\lambda_f < 1$ and $\beta < 1$, it is clear that the matrix P_B is nonnegative. Hence, we have

$$\rho(P_B) < 1 \underset{\text{Lem.1}}{\Longleftrightarrow} P^{-1} \geq 0 \underset{\text{Lem.2}}{\Longleftrightarrow} \rho(P_1^{-1}P_2) = \rho(\mathbb{M}) < 1. \tag{3.15}$$

Our ultimate goal is to prove that $\rho(\mathbb{M}) < 1$, and that due to (3.15) it is sufficient to prove that $\rho(P_B) < 1$. Our proof toward $\rho(P_B) < 1$ is divided into two parts.

Part 1: irreducibility of the matrix P_B. The directed graph $G(P_B)$ associated with the matrix P_B defined by (3.14) is shown in Figure 1. For the arbitrarily chosen vertices i and j, the path from i to j can be chosen as

- if $i > j$: the path is $(i \to i-1 \to i-2 \to \cdots \to j)$;
- if $i < j$: the path is $(i \to i-1 \to i-2 \to \cdots \to 2 \to 1 \to j)$.

Hence, $G(P_B)$ is strongly connected. From Definition 2, we know that the matrix P_B is irreducible.

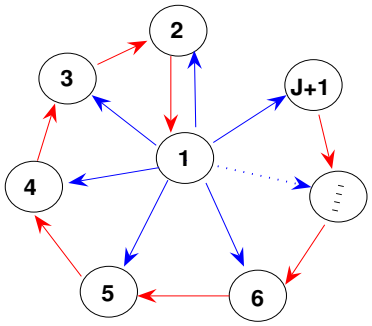

Figure 1. Directed graph $G(P_B)$ associated with the matrix P_B defined by (3.14).

Part 2: the characteristic polynomial of the matrix P_B. Let $R(\lambda) = \Delta(P_B - \lambda I)$ be the characteristic polynomial of the matrix P_B, where λ is a scale number and for any square matrix A, $\Delta(A)$ denotes the determinant of A. To derive an explicit expression of $R(\lambda)$, we recall a well known result which states that

$$\Delta(P_B - \lambda I) = -\lambda\Delta(R_{1,1}) - \frac{\eta\lambda_f^J}{1-\beta}\Delta(R_{1,2}) + \frac{\eta\lambda_f^{J-1}}{1-\beta}\Delta(R_{1,3}) + \cdots + (-1)^{1+J}\frac{\eta\lambda_f}{1-\beta}\Delta(R_{1,J+1}), \tag{3.16}$$

where $R_{1,j}$ is a $J \times J$ matrix obtained by removing the first row and the j-th column from the matrix $P_B - \lambda I$. Clearly, $\Delta(R_{1,1}) = (-\lambda)^J$ and $\Delta(R_{1,2}) = \frac{\lambda_f}{1-\mu\lambda_f}(-\lambda)^{J-1}$ since both $R_{1,1}$ and $R_{1,2}$ are lower triangular matrices. But the matrices $R_{1,j}$ with $3 \le j \le J+1$ are not lower triangular. For any $j \in \{3, 4, \ldots, J, J+1\}$, we have

$$R_{1,j} = \begin{pmatrix} \sigma & -\lambda & & & & & & \\ & \ddots & \ddots & & & & & \\ & & \sigma & -\lambda & & & & \\ & & & \sigma & 0 & & & \\ & & & 0 & -\lambda & & & \\ & & & & \sigma & -\lambda & & \\ & & & & & \ddots & \ddots & \\ & & & & & & \sigma & -\lambda \end{pmatrix}, \quad \sigma = \frac{\lambda_f}{1-\mu\lambda_f},$$

where the entries "$\sigma, 0$" corresponds to the $(j-1)$-th row of the matrix $R_{1,j}$. It follows by applying the expansion like (3.16) to $R_{1,j}$ that $\Delta(R_{1,j}) = \sigma^{j-1}(-\lambda)^{J-j+1}$. Hence, substituting these expressions into

(3.16) gives

$$\begin{aligned}\Delta(P_B-\lambda I) &= (-\lambda)^{J+1}+\sum_{j=2}^{J+1}(-1)^{j+1}\frac{\eta\lambda_f^{J+1}}{(1-\mu\lambda_f)^{j-1}(1-\beta)}(-\lambda)^{J-j+1}\\ &= \lambda^J(-1)^{J+1}\left(\lambda-\frac{\eta\lambda_f^{J+1}}{1-\beta}\sum_{j=2}^{J+1}\frac{1}{[(1-\mu\lambda_f)\lambda]^{j-1}}\right)\\ &= \lambda^J(-1)^{J+1}\left(\lambda-\frac{\eta\lambda_f^{J+1}}{1-\beta}\sum_{j=1}^{J}\frac{1}{[(1-\mu\lambda_f)\lambda]^{j}}\right).\end{aligned} \tag{3.17}$$

From (3.17), we know that the nonzero root of $\Delta(P_B-\lambda I)=0$ satisfies

$$\lambda=\frac{\eta\lambda_f^{J+1}}{1-\beta}\sum_{j=1}^{J}\frac{1}{[(1-\mu\lambda_f)\lambda]^{j}}. \tag{3.18}$$

Now, by combining Parts 1 and 2 and by using Lemma 3 we get

$$\rho(P_B)<1 \Leftrightarrow \frac{\eta\lambda_f^{J+1}}{1-\beta}\sum_{j=1}^{J}\frac{1}{(1-\mu\lambda_f)^{j}}<1\left(\Leftrightarrow \frac{\eta\lambda_f^{J}\left(1-\left[1-\mu\lambda_f\right]^{J}\right)}{\mu(1-\beta)(1-\mu\lambda_f)^{J}}<1\right), \tag{3.19}$$

since $\sum_{j=1}^{J}\frac{1}{[(1-\mu\lambda_f)\lambda]^{j}}$ is a decreasing function of λ for $\lambda>0$.

Remark 2 (speed-up analysis)**.** *On each time grid, let T_e and t_e be the CPU time spent for applying the backward Euler method directly to the original PDE and the PDE on a subdomain, respectively. Since the number of subdomains is N and the overlap size is small, it holds that $t_e\approx\frac{T_e}{N}$. Now, assume we have $N_t\times N$ processors at our disposal, where N_t processors are assigned to the parareal method and N processors are assigned to the SWR method. The overall cost of the Para-SWR algorithm for per iteration is given by the cost of applying the fine and the coarse propagators simultaneously on each processor—$(J+1)t_e$, the cost of applying the coarse propagator $\mathcal{G}$ sequentially—$N_t t_e$ and some necessary communication cost—T_c; in total*

$$T_{Para\text{-}SWR}:=Jt_e+N_t t_e+T_c. \tag{3.20}$$

At the current iteration, say the k-th iteration, the communication cost T_c spent on each coarse time grid T_n mainly comes from transmission of solutions on fine time grids between adjacent subdomains. For example, the processor $P_{n,i}$—assigned to the n-th coarse time grid T_n and the i-th subdomain, needs to get fine solutions $V_{T_n,1,2,\ldots,J}(i-1)$ and $V_{T_n,1,2,\ldots,J}(i+1)$ from its neighboring processor $P_{n,i-1}$ and $P_{n,i+1}$, respectively. Also, it needs to broadcast the solutions $V_{T_n,1,2,\ldots,J}(i)$ to its two neighboring processors. Clearly, this process can be done simultaneously in all parallelled time intervals. In the longitudinal direction, this can be done sequentially by two steps: first the N processors simultaneously transmit the fine solutions along the upward direction and then along the downward direction (see Figure 2 for illustration). Let δ be the unit cost spent to communicate a float number between two processors. Then, the total communication time T_c can be written as

$$T_c=2\times J\times m\times\delta, \tag{3.21}$$

where m is the number of space mesh points on a spatial subdomain. Substituting T_c into (3.20) *gives*

$$T_{Para\text{-}SWR} = (J + N_t)t_e + 2Jm\delta \approx (J + N_t)\frac{T_e}{N} + 2Jm\delta \tag{3.22}$$

The cost of applying the fine propagator sequentially and directly to the large circuit is

$$T_{Sequent} := N_t \times J \times T_e. \tag{3.23}$$

We therefore obtain a speed-up of using the Para-SWR algorithm as

$$\frac{T_{Sequent}}{T_{Para\text{-}SWR}} = \frac{N_t \times J \times T_e}{K_{\max}\left((J + N_t)\frac{T_e}{N} + 2Jm\delta\right)} \approx \frac{NN_tJ}{K_{\max}(J + N_t)} \ (if\ T_e \gg 2Jm\delta), \tag{3.24}$$

where $K_{\max}$ is the number of iterations performed to meet the stopping criterion. The condition $T_e \gg 2Jm\delta$ occurs naturally if the space domain of the original PDE is large and the discretization mesh Δx is small.

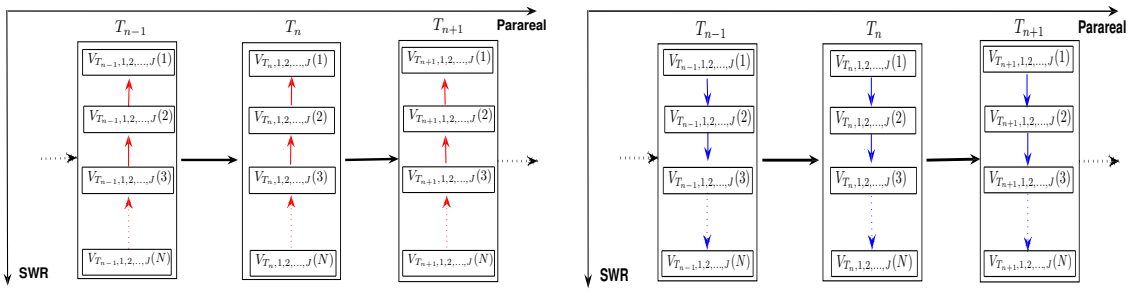

Figure 2. Upward and downward transmission of the solutions at fine time grids between subdomains.

3.2. Weak scaling

The analysis given above is performed on sufficiently long time domains and therefore $\rho(\mathbb{M})$ is independent of N_t, the number of paralleled time intervals. We now focus on proving that $\rho(\mathbb{M})$ is robust with respect to N, the number of subdomains. We assume $a > 0$ and consider the case $q = 0$. Let Δt, J, Δx and L (the length of the space domain) be fixed constants, and then we claim that the arguments λ_f and λ_c decrease as N increases. Indeed, since $q = 0$ we have $A_+ = A_- = A$ and the eigenvalues are given by

$$\lambda_i(A) = -\left(\frac{2\nu^2}{\Delta x^2} + a\right) + 2\frac{\nu^2}{\Delta x^2}\cos\left(\frac{i\pi}{m+1}\right) = -a - \frac{4\nu^2}{\Delta x^2}\sin^2\left(\frac{j\pi}{2(m+1)}\right), j = 1, 2, \ldots, m, \tag{3.25}$$

where $m = \frac{L/\Delta x - 2}{N} + 1$. From (3.25), we have

$$\lambda_f = \frac{1}{1 + \Delta t\left(a + \frac{4\nu^2}{\Delta x^2}\sin^2\left(\frac{\pi}{2(m+1)}\right)\right)}, \quad \lambda_c = \frac{1}{1 + \Delta T\left(a + \frac{4\nu^2}{\Delta x^2}\sin^2\left(\frac{\pi}{2(m+1)}\right)\right)}. \tag{3.26}$$

Clearly, both λ_f and λ_c increase as m increases, i.e., λ_f and λ_c decrease as N increases, since $L = [N \times (m-1)+2] \times \Delta x$ is constant. For the matrix $\mathcal{B}$ defined by (2.7) it is easy to know that the quantity $\|\mathcal{B}\|_2$ is independent of the number m and equal to $\frac{\nu^2}{\Delta x^2}$ since $q = 0$. Hence, we get $\mu = \Delta t \frac{\nu^2}{\Delta x^2}$ for all possible N.

Now, for the iteration matrix $\mathbb{M}$ defined by (3.9) the aforementioned analysis implies that, except for β, all of the other elements decrease as N increases. From (3.25), we also have

$$\begin{aligned} \left\|\bar{\mathcal{A}}_f^J - \bar{\mathcal{A}}_c\right\|_2 &= \max_{1\le i\le m} \left| \frac{1}{\left(1+\Delta t\left(a+\frac{4\nu^2}{\Delta x^2}\sin^2\left(\frac{i\pi}{2(m+1)}\right)\right)\right)^J} - \frac{1}{1+\Delta T\left(a+\frac{4\nu^2}{\Delta x^2}\sin^2\left(\frac{i\pi}{2(m+1)}\right)\right)} \right| \\ &\le \max_{z\ge 0}\Gamma(J), \end{aligned} \tag{3.27}$$

where $\Gamma(J) = \max\limits_{z\ge 0}\left|\frac{1}{1+Jz} - \frac{1}{(1+z)^J}\right|$. The function $\Gamma(J)$ is shown in Figure 3, and it holds that $\Gamma(J) \le 0.205$ for $J \in \{2, 3, \ldots, 500\}$. We therefore can bound β as $\beta \le \frac{\Gamma(J)}{1-\lambda_c}$ for all possible N.

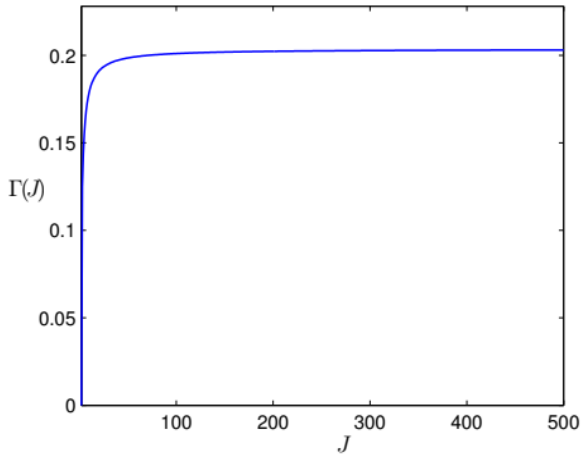

Figure 3. Quantity $\Gamma(J)$ as a function of J for $J \in \{2, 3, \ldots, 500\}$.

Corollary 1 (weak scaling)**.** *Let $q = 0$ and $a > 0$. Then, for a fixed Δt, J, Δx and L, the Para-SWR algorithm converges linearly on sufficiently long time intervals and the convergence rate can be bounded by $\rho(\mathbb{M}^*)$, which is independent of the number of subdomains, provided Condition* (3.11) *is satisfied with*

$$\begin{aligned} &\lambda_f^* = \frac{1}{1+\Delta t\left(a+\frac{4\nu^2}{\Delta x^2}\sin^2\left(\frac{\pi}{2(m_{\max}+1)}\right)\right)}, \lambda_c^* = \frac{1}{1+\Delta T\left(a+\frac{4\nu^2}{\Delta x^2}\sin^2\left(\frac{\pi}{2(m_{\max}+1)}\right)\right)}, \\ &\beta^* = \frac{\Gamma(J)}{1-\lambda_c^*}, \mu^* = \frac{\nu^2\Delta t}{\Delta x^2}, \eta^* = \frac{\mu^*}{1-\lambda_c^*}, \end{aligned} \tag{3.28}$$

where the matrix $\mathbb{M}^*$ is derived from $\mathbb{M}$ by letting $\lambda_f = \lambda_f^*$, $\lambda_c = \lambda_c^*$, $\eta = \eta^*$, $\beta = \beta^*$ and $\mu = \mu^*$, and $m_{\max} = 1 + \left\lfloor \frac{L}{2\Delta x} \right\rfloor$ and $\left\lfloor \frac{L}{2\Delta x} \right\rfloor$ denote the integer part of $\frac{L}{2\Delta x}$.

Proof. Since $N \ge 2$ and the length of the space domain $L = [N \times (m-1)+2]\Delta x$ is fixed, it is easy to know that the maximal integer m in (3.26) is $m_{\max} = 1 + \left\lfloor \frac{L}{2\Delta x} \right\rfloor$. Hence, for any possible integer N it

holds that $\mathbb{M} \leq \mathbb{M}^*$. Since both $\mathbb{M}$ and $\mathbb{M}^*$ are nonnegative matrices, we know from [45, Theorem 2.21, p. 51] that $\rho(\mathbb{M}) \leq \rho(\mathbb{M}^*)$.

Note that, the choice $q = 0$ corresponds to $p = \infty$ and in this case the transmission condition used in (2.3) reduces to the Dirichlet transmission condition. For the general Robin transmission condition, i.e., $q \in (0, +\infty)$, it is difficult to prove a similar conclusion like Corollary 1.

We next investigate how the convergence rate of the Para-SWR algorithm behaves with respect to the quantities $\frac{\nu^2}{\Delta x^2}$ and a. It is difficult to make a theoretical analysis, and we have to rely on numerical investigation. Since $\rho(\mathbb{M}^*)$ is a suitable upper bound of $\rho(\mathbb{M})$ and independent of the number of subdomains, we plotted it in Figure 4 on the top of the quantity $\rho(\mathbb{M}^*)$ as a function of $\frac{\nu^2}{\Delta x^2}$ and a, where $J = 50$ and $\Delta T = 0.1$ are considered. We see that $\rho(\mathbb{M}^*)$ decreases as a increases or $\frac{\nu^2}{\Delta x^2}$ decreases. This provides us with two possible acceleration strategies for the Para-SWR algorithm. The first one is the change of variables that we have described at the beginning of Section 2. For the model problem (2.1), by letting $u(x,t) = v(x,t)e^{\tau t}$ we get

$$\begin{cases} \partial_t v - \nu^2 \partial_{xx} v + (a+\tau)v = e^{-\tau t} f(x,t), & (x,t) \in (0,L) \times \mathbb{R}^+, \\ v(0,t) = e^{-\tau t} b_1(t), & t \geq 0, \\ v(L,t) = e^{-\tau t} b_2(t), & t \geq 0, \\ v(x,0) = u_0(x), & x \in [0,L], \end{cases} \tag{3.29}$$

and this is essentially the same PDE as (2.1), but with a new reaction parameter $a + \tau$. Due to the plot shown in Figure 4 on the top, it seems that the Para-SWR algorithm for (3.29) converges very fast if we choose a very large τ. However, this is a false prediction. Briefly speaking, for a large τ, we find numerically that even though $\max\limits_{j,\,n} \left\| V^k(x_j, T_n) - V(x_j, T_n) \right\|_2$ is already very small, $\max\limits_{j,\,n} \left\| U^k(x_j, T_n) - U(x_j, T_n) \right\|_2$ is still a huge number, where $U^k(x_j, T_n)$ is calculated by $U^k(x_j, T_n) = e^{\tau T_n} V^k(x_j, T_n)$; we denote V^k as the numerical solutions generated by the Para-SWR algorithm after k iterations for the transformed PDE (3.29). To illustrate this, by using the example given in (4.1) with $a = 1, \nu = 0.01, T = 5, \Delta T = 0.1, J = 50$ and $N = 8$, we plotted the diminution history of $\max\limits_{j,\,n} \left\| V^k(x_j, T_n) - V(x_j, T_n) \right\|_2$ and $\max\limits_{j,\,n} \left\| U^k(x_j, T_n) - U(x_j, T_n) \right\|_2$ in Figure 4 on the bottom for two choices of τ.

The second possibility is the change of space coordinates. Note that, if we let $x = \omega \tilde{x}$ the PDE problem (2.1) is transformed to

$$\partial_t u(\tilde{x}, t) - (\omega\nu)^2 \partial_{\tilde{x}}^2 u(\tilde{x}, t) + a u(\tilde{x}, t) = f(\tilde{x}, t), (\tilde{x}, t) \in (0, \tilde{L}) \times \mathbb{R}^+, \tag{3.30}$$

where $\tilde{L} = \frac{L}{\omega}$. The new diffusion coefficient $\omega\nu$ can be sufficiently small by choosing a small positive ω, which implies that the Para-SWR algorithm converges significantly fast. However, in the meantime the computational domain $\left(0, \frac{L}{\omega}\right)$ will become large and to maintain a required spatial accuracy, a uniform step size Δx should be used for all ω. Hence, for a fixed N—the number of subdomains, the computational cost at each iteration will also significantly increase. To reduce the computational cost at each iteration, we can increase the number of subdomains to $\tilde{N} = N\left\lceil \frac{1}{\omega} \right\rceil$. Since the convergence rate of the Para-SWR algorithm is robust with respect to the number of subdomains, we know that the total computational cost for the Para-SWR algorithm will be evidently reduced by using a small ω. In

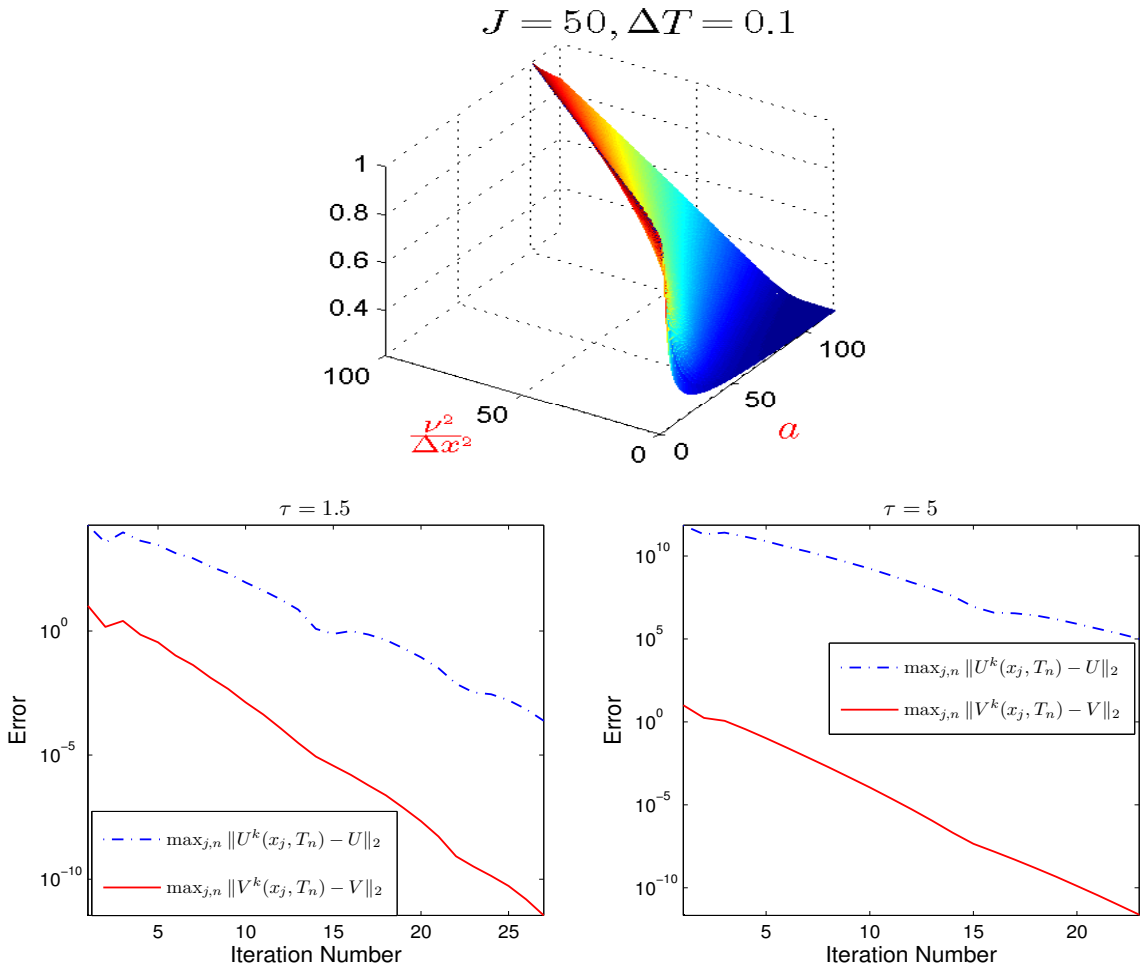

Figure 4. Top: the quantity $\rho(\mathbb{M}^*)$ as a function of $\frac{\nu^2}{\Delta x^2}$ and a, where $J = 50$ and $\Delta T = 0.1$. Bottom: diminution of $\max\limits_{j,\,n} \left\|V^k(x_j, T_n) - V(x_j, T_n)\right\|_2$ (solid line) and $\max\limits_{j,\,n} \left\|U^k(x_j, T_n) - U(x_j, T_n)\right\|_2$ (dash-dot line).

practical computation, we can choose $\omega = \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots$, since in this way it is convenient to increase the subdomains multiplicatively, as $\tilde{N} = 2N, 4N, 8N, \ldots$.

4. Numerical results

In this section, we show several numerical experiments to test the efficiency of the Para-SWR algorithm proposed in this paper. We focus on observing how the convergence rate of this algorithm behaves with respect to N—the number of subdomains, N_t—the number of paralleled time intervals and J—the multiplication ratio between ΔT and Δt. We consider the following PDE problem

$$\begin{cases} \partial_t u - \nu^2 \partial_{xx} u + au = 5\dfrac{\sin\left(3\left[\sqrt{(t-0.5T)^2+(10(x-0.5))^2}\right]\right)}{\sqrt{(t-0.5T)^2+(10(x-0.5))^2}}, & (x,t) \in (0,1)\times(0,T), \\ u(0,t) = u(1,t) = 0, & t \geq 0, \\ u(x,0) = 0, & x \in [0,1], \end{cases} \tag{4.1}$$

where $\nu = 0.01$ and $a = 6$. The space discretization parameter Δx is fixed as $\Delta x = \frac{1}{128}$ and the initial solution for the Para-SWR algorithm is chosen randomly in the interval $[0, T]$. For all experiments the iteration stops when the maximal error between the current iteration and the converged solution arrives at a prescribed tolerance.

Example 4.1 (Dirichlet transmission condition). We first consider the Dirichlet transmission condition, i.e., $q = 0$. Let $J = 50$ and $\Delta T = 0.1$, and then it is easy to get $\rho(\mathbb{M}^*) = 0.8465$. Hence, Corollary 1 implies that the convergence rate of the Para-SWR algorithm is robust with respect to N, the number of subdomains. The results shown in Figure 5 on the top indicate that this theoretical predication is correct. The other two panels of Figure 5 are concerned with the dependence of the convergence rate of the algorithm on the multiplication ratio J (bottom left) and the number of parallelled time intervals (bottom right). We see that the convergence rate is also robust with respect to J and N_t. The results shown in Figure 5 imply that the Para-SWR algorithm really scales weakly, and that this strong robustness indicates that the strategy of change of space coordinates mentioned in Subsection 3.2 is viable.

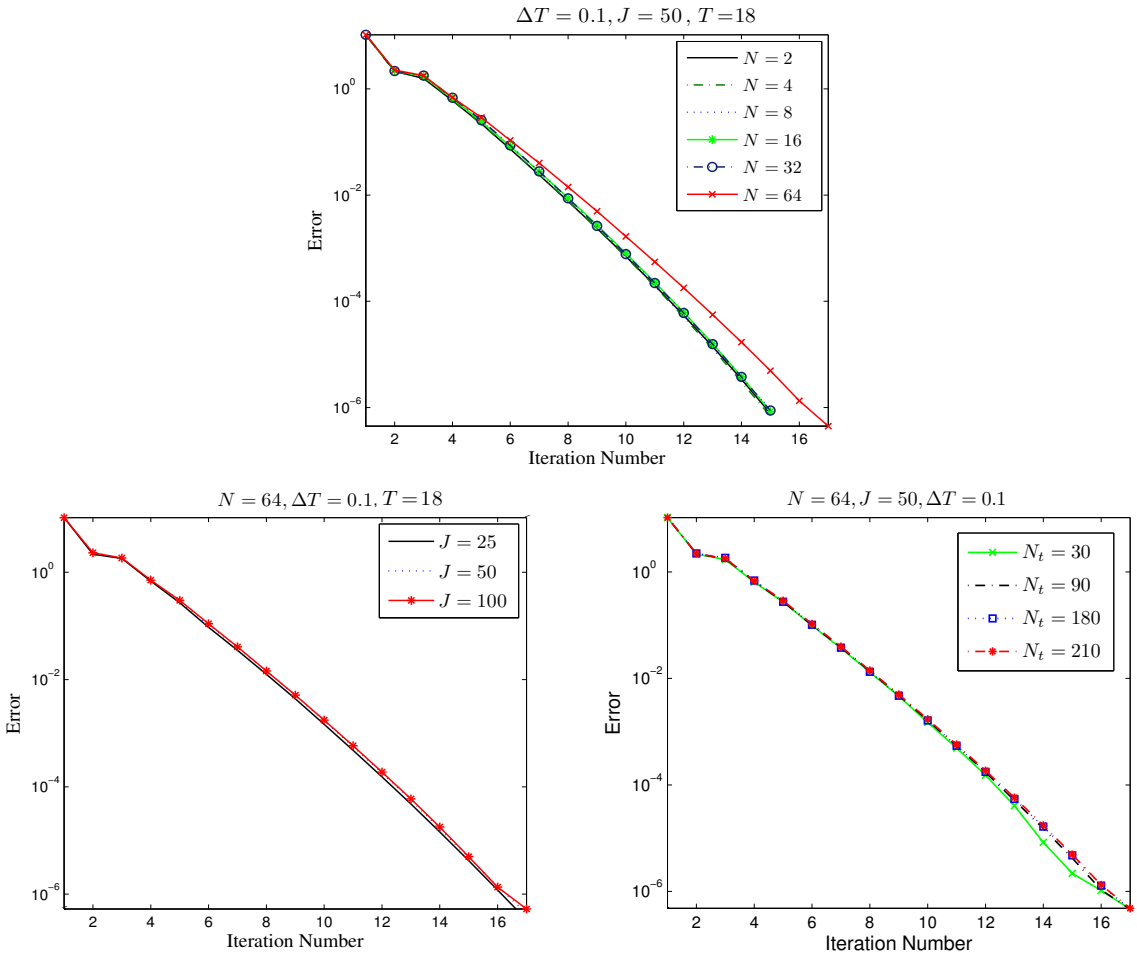

Figure 5. Illustration of the robustness of the Para-SWR algorithm with respect to N (top), J (bottom left) and N_t (bottom right).

Example 4.2 (Robin transmission condition). We next consider the Robin transmission condition.

For the SWR algorithm, it is well known that the Robin transmission condition is more powerful than the Dirichlet transmission condition; see, e.g., [8, 9]. For $T = 9$, $J = 50$ and $\Delta T = 0.1$, we show in Figure 6, on the top, the argument $\rho(\mathbb{M})$ as a function of the parameter q $\left(= \frac{1}{\Delta x p - 1}\right)$. We see that $\rho(\mathbb{M})$ attains its minimum at $q = 0$, i.e., $p = \infty$, which corresponds to the Dirichlet transmission condition. Does this predict the practical computation well? In Figure 6 on the bottom, we show the errors obtained by running the Para-SWR algorithms with Robin transmission conditions after k iterations and various choices of the free parameters q. We see that, indeed, $q = 0$ is the best choice; therefore, these numerical results confirm our theoretical predication shown on the top.

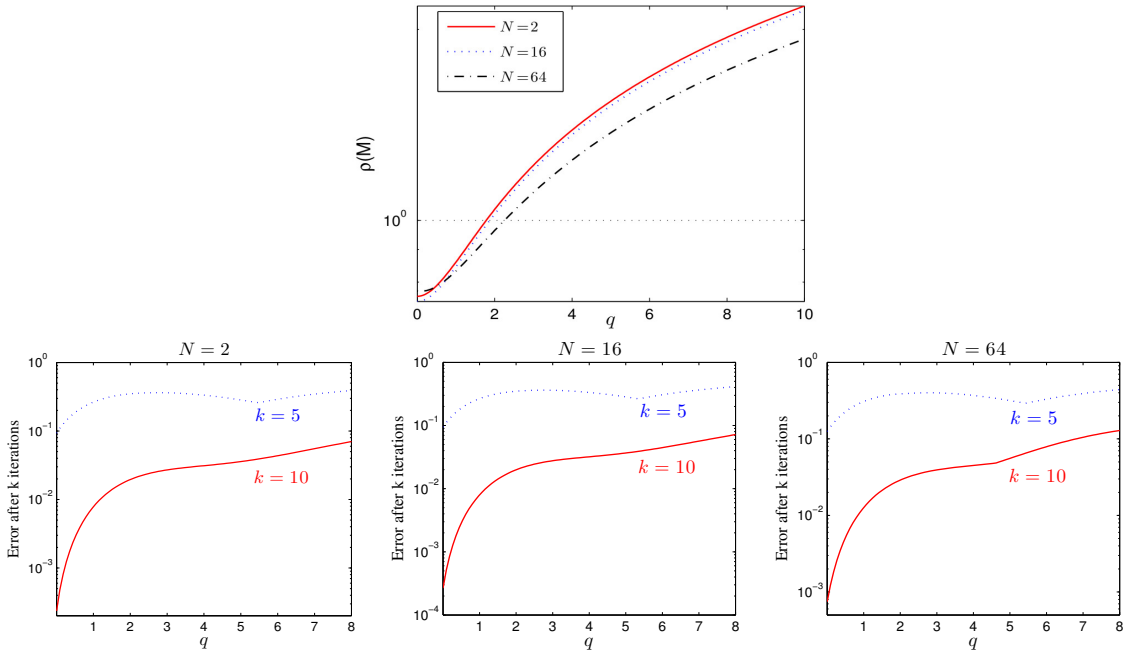

Figure 6. Errors obtained by running the Para-SWR algorithms with Robin transmission conditions after k iterations and various choices of the free parameters q. From left to right, $N = 2$, $N = 16$ and $N = 64$.

The plot shown in Figure 6 on the top also predicts that increasing q has a negative effect on the convergence rate of the Para-SWR algorithm, and that, for a very large q the algorithm will not converge. This is verified in Figures 7 and 8. In Figure 7, we show on the top row the third, the fourth and the fifth iterations of the 8-subdomain Para-SWR algorithm with $q = 0$, and below we show the same iterations for the algorithm with $q = 4$. This clearly shows that the choice $q = 0$ is more advisable than $q > 0$, which confirms the prediction shown in Figure 6 on the top.

In Figure 8, we show the measured convergence rate of the Para-SWR algorithm in the case of two subdomains and 32 subdomains. The results shown in this figure imply that a large q really slows down the convergence rate; note how the iteration stagnates for $q = 100$.

5. Conclusions

In this paper, we have presented a hybrid algorithm based on the well understood parareal algorithm and the SWR algorithm. The algorithm, namely, Para-SWR, has been defined by first applying the

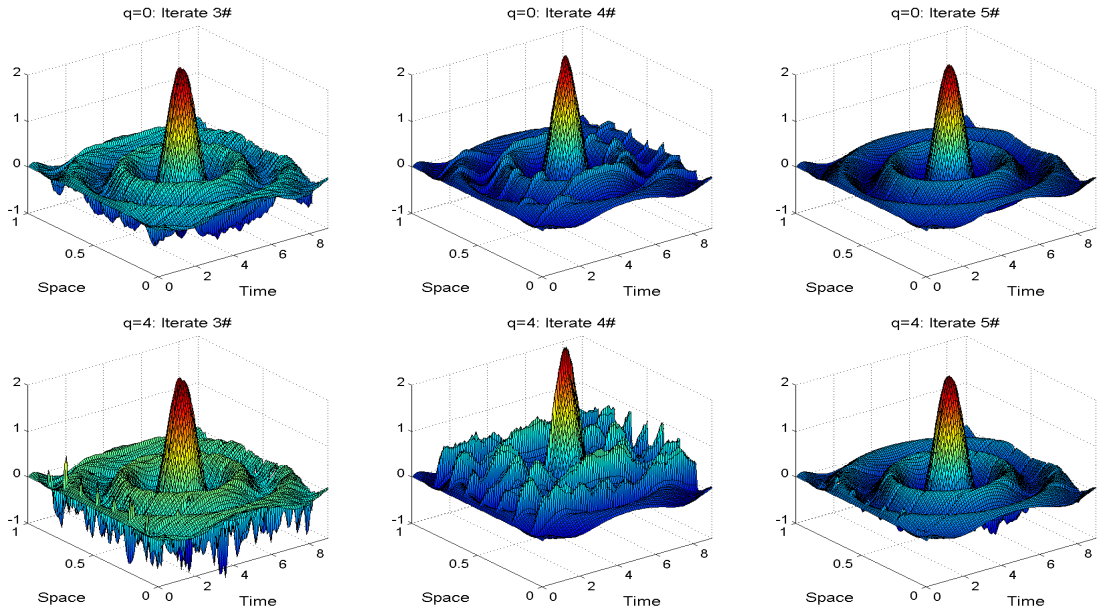

Figure 7. From left to right, the solution generated by the Para-SWR algorithm after 3, 4 or 5 iterations. Top row: $q = 0$ (the Dirichlet transmission condition). Bottom row: $q = 0$ (the Robin transmission condition with $p = \frac{5}{4\Delta xp}$).

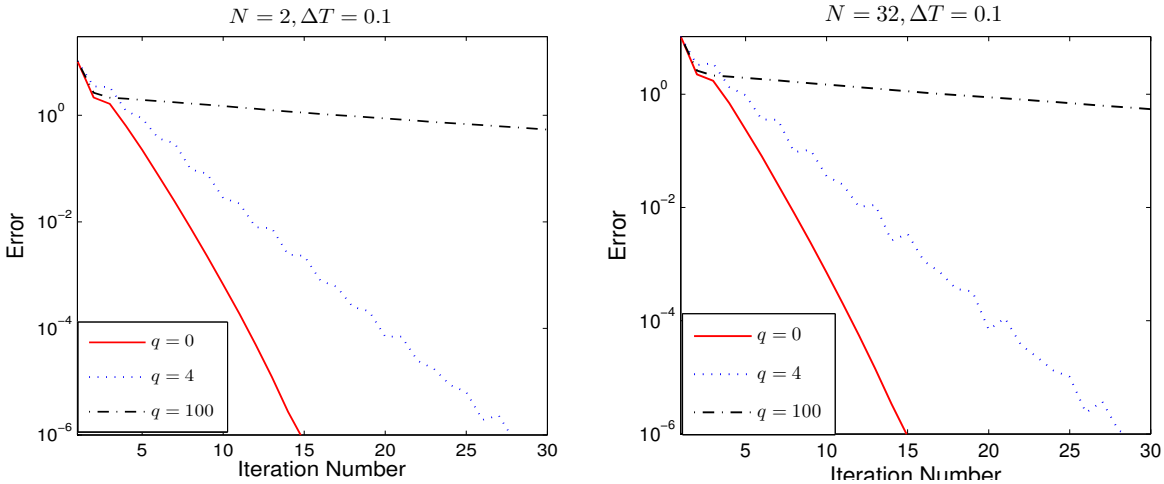

Figure 8. Convergence history of the Para-SWR algorithm under different choices of q. Left: two subdomain case. Right: 32 subdomain case.

SWR algorithm to time-dependent PDEs in the case of general N subdomains and then applying the one-iteration parareal algorithm to each subproblem. Convergence of the algorithm was analyzed, and it has been shown theoretically and numerically that the convergence rate is robust with respect to the number of subdomains and the number of parallelled time intervals. This implies that, compared to the parareal algorithm and the SWR algorithm, the proposed hybrid algorithm really brings more parallelism.

There are still some important problems that need to be answered in further work. First of all, the spectral radius $\rho(\mathbb{M})$ is not a good bound of the practical convergence rate. It is conservative and even

invalid in some cases. Briefly speaking, for the case $\frac{\nu^2}{\Delta x^2}/a \gg 1$ we found that $\rho(\mathbb{M}) > 1$, while the Para-SWR algorithm still converges. For example, by letting $a = 0$ in (4.1) (i.e., the heat equation), $\nu = 0.01$, $T = 10, \Delta x = \frac{1}{128}$, $\Delta T = 0.1$ and $J = 50$, we have $\rho(\mathbb{M}) > 1$ for all $q \geq 0$ and $\frac{\nu^2}{\Delta x^2} > 0$, but the numerical results indicate that the Para-SWR algorithm is actually convergent.

The other problem is how to generalize current work to other numerical methods, such as the Runge-Kutta (RK) methods. Such a generalization is not straightforward: if we choose an s-stage RK method with $\tilde{s}$ off-steps $\left(\text{the intermediary nodes located in the fine time slice}\left(T_{n+\frac{j\Delta t}{J}}, T_{n+\frac{(j+1)\Delta t}{J}}\right)\right)$, the iteration matrix $\mathbb{M}$ would be a $(1 + \tilde{s} + J) \times (1 + \tilde{s} + J)$ matrix and its structure would be completely different from the one given by (3.9). Therefore, the current approach for proving $\rho(\mathbb{M}) < 1$ is not applicable. However, one can check that the proof can be straightforwardly generalized to the linear–θ method, which can be regarded as a family of simple RK methods with $\tilde{s} = 0$.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (12101089) and the Natural Science Foundation of Sichuan Province (2022NSFSC1844).

Conflict of interest

The authors declare that there is no conflicts of interest.

References

1. A. Toselli, O. Widlund, *Domain Decomposition Methods: Algorithms and Theory*, Springer, Berlin, 2005.
2. T. P. A. Mathew, *Domain Decomposition Methods for the Numerical Solution of Partial Differential Equations*, Springer-Verlag, Berlin, 2008.
3. X. C. Cai, Additive Schwarz algorithms for parabolic convection-diffusion equations, *Numer. Math.*, **60** (1991), 41–61. https://doi.org/10.1007/BF01385713
4. X. C. Cai, Multiplicative Schwarz methods for parabolic problems, *SIAM J. Sci. Comput.*, **15** (1994), 587–603. https://doi.org/10.1137/0915039
5. L. Qin, X. J. Xu, Optimized Schwarz methods with Robin transmission conditions for parabolic problems, *SIAM J. Sci. Comput.*, **31** (2008), 608–623. https://doi.org/10.1137/070682149
6. M. Al-Khaleel, S. L. Wu, Quasi-overlapping semi-discrete Schwarz waveform relaxation algorithms: The hyperbolic problem, *Comput. Methods Appl. Math.*, **20** (2020), 397–417. https://doi.org/10.1515/cmam-2018-0188
7. D. Bennequin, M. J. Gander, L. Gouarin, L. Halpern, Optimized Schwarz waveform relaxation for advection reaction diffusion equations in two dimensions, *Numer. Math.*, **134** (2016), 513–567. https://doi.org/10.1007/s00211-015-0784-8
8. D. Bennequin, M. J. Gander, L. Halpern, A homographic best approximation problem with application to optimized Schwarz waveform relaxation, *Math. Comput.*, **78** (2009), 185–223. https://doi.org/10.1090/S0025-5718-08-02145-5

9. M. J. Gander, L. Halpern, Optimized Schwarz waveform relaxation for advection reaction diffusion problems, *SIAM J. Numer. Anal.*, **45** (2007), 666–697. https://doi.org/10.1007/s00211-015-0784-8
10. M. J. Gander, A. M. Stuart, Space-time continuous analysis of waveform relaxation for the heat equation, *SIAM J. Sci. Comput.*, **19** (1998), 2014–2031. https://doi.org/10.1137/S1064827596305337
11. E. Giladi, H. B. Keller, Space-time domain decomposition for parabolic problems, *Numer. Math.*, **93** (2002), 279–313. https://doi.org/10.1007/s002110100345
12. S. L. Wu, M. D. Al-Khaleel, Semi-discrete Schwarz waveform relaxation algorithms for reaction diffusion equations, *BIT Numer. Math.*, **54** (2014), 831–866. https://doi.org/10.1007/s10543-014-0475-3
13. S. L. Wu, M. D. Al-Khaleel, Convergence analysis of the Neumann-Neumann waveform relaxation method for time-fractional RC circuits, *Simul. Model. Pract. Theory*, **64** (2016), 43–56. https://doi.org/10.1016/j.simpat.2016.01.002
14. J. L. Lions, Y. Maday, G. Turinici, Résolution d'EDP par un schéma en temps pararéel, *C. R. Acad. Sci. Paris Sér. I Math.*, **332** (2001), 661–668. https://doi.org/10.1016/S0764-4442(00)01793-6
15. M. Cai, J. Mahseredjian, I. Kocar, X. Fu, A. Haddadi, A parallelization-in-time approach for accelerating EMT simulations, *Electr. Power Syst. Res.*, **197** (2021), 107346. https://doi.org/10.1016/j.epsr.2021.107346
16. T. Cheng, N. Lin, V. Dinavahi, Hybrid parallel-in-time-and-space transient stability simulation of large-ccale AC/DC grids, *IEEE Trans. Power Syst.*, in press, 2022. https://doi.org/10.1109/TPWRS.2022.3153450
17. I. C. Garcia, I. Kulchytska-Ruchka, S. Schops, Parareal for index two differential algebraic equations, *Numer. Alg.*, **91** (2022), 389–412. https://doi.org/10.1007/s11075-022-01267-1
18. E. Celledoni, T. Kvamsdal, Parallelization in time for thermo-viscoplastic problems in extrusion of aluminium, *Int. J. Numer. Methods Eng.*, **79** (2009), 576–598. https://doi.org/10.1002/nme.2585
19. D. Max, M. Marc, D. Stéphane, Parareal operator splitting techniques for multi-scale reaction waves: numerical analysis and strategies, *ESAIM Math. Model. Numer. Anal.*, **45** (2011), 825–852. https://doi.org/10.1051/m2an/2010104
20. L. Fang, S. Vandewalle, J. Meyers, A parallel-in-time multiple shooting algorithm for large-scale PDE-constrained optimal control problems, *J. Comput. Phys.*, **452** (2021), 110926. https://doi.org/10.1016/j.jcp.2021.110926
21. M. J. Gander, F. Kwok, J. Salomon, PARAOPT: a parareal algorithm for optimality systems, *SIAM J. Sci. Comput.*, **42** (2020), A2773–A2802. https://doi.org/10.1137/19M1292291
22. Y. Maday, M. K. Riahi, J. Salomon, Parareal in time intermediate targets methods for optimal control problems, in *Control and Optimization with PDE Constraints*, Birkhäuser, Basel, **164** (2013), 79–92. https://doi.org/10.1007/978-3-0348-0631-2_5
23. M. K. Riahi, J. Salomon, S. J. Glaser, D. Sugny, Fully efficient time-parallelized quantum optimal control algorithm, *Phys. Rev. A*, **93** (2016), 043410. https://doi.org/10.1103/PhysRevA.93.043410

24. X. M. Gu, S. L. Wu, A parallel-in-time iterative algorithm for Volterra partial integro-differential problems with weakly singular kernel, *J. Comput. Phys.*, **417** (2020), 109576. https://doi.org/10.1016/j.jcp.2020.109576

25. X. M. Gu, Y. L. Zhao, X. L. Zhao, B. Carpentieri, Y. Y. Huang, A note on parallel preconditioning for the all-at-once solution of Riesz fractional diffusion equations, *Numer. Math. Theor. Meth. Appl.*, **14** (2021), 893–919. https://doi.org/10.4208/nmtma.OA-2020-0020

26. Y. L. Zhao, P. Y. Zhu, X. M. Gu, X. L. Zhao, H. Y. Jian, A preconditioning technique for all-at-once system from the nonlinear tempered fractional diffusion equation, *J. Sci. Comput.*, **83** (2020), 10. https://doi.org/10.1007/s10915-020-01193-1

27. M. J. Gander, S. Vandewalle, Analysis of the parareal time-parallel time-integration method, *SIAM J. Sci. Comput.*, **29** (2007), 556–578. https://doi.org/10.1137/05064607X

28. M. J. Gander, S. L. Wu, A diagonalization-based parareal algorithm for dissipative and wave propagation problems, *SIAM J. Numer. Anal.*, **58** (2020), 2981–3009. https://doi.org/10.1137/19M1271683

29. S. L. Wu, Toward parallel coarse grid correction for the parareal algorithm, *SIAM J. Sci. Comput.*, **40** (2018), A1446–A1472. https://doi.org/10.1137/17M1141102

30. S. L. Wu, T. Zhou, Convergence analysis for three parareal solvers, *SIAM J. Sci. Comput.*, **37** (2015), A970–A992. https://doi.org/10.1137/140970756

31. T. Cadeau, F. Mafoules, Coupling the parareal algorithm with the waveform relaxation method for the solution of differential algebraic equations, in *10th International Symposium on Distributed Computing and Applications to Business, Engineering and Science*, (2011), 15–19. https://doi.org/10.1109/DCABES.2011.34

32. Y. L. Jiang, B. Song, Coupling parareal and Dirichlet-Neumann/Neumann-Neumann waveform relaxation methods for the heat equation, in *International Conference on Domain Decomposition Methods*, Springer, Cham, **125** (2018), 405–413. https://doi.org/10.1007/978-3-319-93873-8_38

33. T. Cadeau, F. Magoulès, Coupling parareal and waveform relaxation methods for power systems, in *IEEE International Conference on Electrical and Control Engineering*, (2011), 2947–2950. https://doi.org/10.1109/ICECENG.2011.6057305

34. J. Li, Y. L. Jiang, Z. Miao, A parareal approach of semi-linear parabolic equations based on general waveform relaxation, *Numer. Methods Partial Differ. Equations*, **35** (2019), 2017–2043. https://doi.org/10.1002/num.22390

35. J. Liu, Y. L. Jiang, A parareal algorithm based on waveform relaxation, *Math. Comput. Simul.* **82** (2012), 2167–2181. https://doi.org/10.1016/j.matcom.2012.05.017

36. J. Liu, Y. L. Jiang, A parareal waveform relaxation algorithm for semi-linear parabolic partial differential equations, *J. Comput. Appl. Math.*, **236** (2012), 4245–4263. https://doi.org/10.1016/j.cam.2012.05.014

37. B. Song, Y. L. Jiang, Analysis of a new parareal algorithm based on waveform relaxation method for time-periodic problems, *Numer. Algorithms*, **67** (2014), 599–622. https://doi.org/10.1007/s11075-013-9810-z

38. B. Song, Y. L. Jiang, A new parareal waveform relaxation algorithm for time-periodic problems, *Int. J. Comput. Math.*, **92** (2015), 377–393. https://doi.org/10.1080/00207160.2014.891734

39. B. Song, Y. L. Jiang, X. Wang, Analysis of two new parareal algorithms based on the Dirichlet-Neumann/Neumann-Neumann waveform relaxation method for the heat equation, *Numer. Algorithms*, **6** (2021), 1685–1703. https://doi.org/10.1007/s11075-020-00949-y

40. E. Lelarasmee, A. E. Ruehli, A. L. Sangiovanni-Vincentelli, The waveform relaxation methods for time-domain analysis of large scale integrated circuits, *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, **1** (1982), 131–145. https://doi.org/10.1109/TCAD.1982.1270004

41. D. Q. Bui, C. Japhet, Y. Maday, P. Omnes, Coupling parareal with optimized Schwarz waveform relaxation for parabolic problems, *SIAM J. Numer. Anal.*, **60** (2022), 913–9399. https://doi.org/10.1137/21M1419428

42. M. J. Gander, Y. Jiang, R. Li, Parareal Schwarz waveform relaxation methods, in *Domain Decomposition Methods in Science and Engineering XX*, Springer, Berlin, Heidelberg, **91** (2012), 451–458. https://doi.org/10.1007/978-3-642-35275-1_53

43. M. Gander, Y. Jiang, B. Song, A superlinear convergence estimate for the parareal Schwarz waveform relaxation algorithm, *SIAM J. Sci. Comput.*, **41** (2019), A1148–A1169. https://doi.org/10.1137/18M1177226

44. M. J. Gander, M. Al-Khaleel, A. Ruehli, Optimized waveform relaxation methods for the longitudinal partitioning of transmission lines, *IEEE Trans. Circuits Syst. I Regul. Pap.* , **56** (2009), 1732–1743. https://doi.org/10.1109/TCSI.2008.2008286

45. R. S. Varga, *Matrix Iterative Analysis*, 2^{nd} edition, Spring-Verlag, Berlin Heidelberg, 2000.

AIMS Press