A parallel in time/spectral collocation combined with finite difference method for the time fractional differential equations

Xianjuan Li and Yanhui Su

Abstract
In this article, we consider the numerical solution for the time fractional differential equations (TFDEs). We propose a parallel in time method, combined with a spectral collocation scheme and the finite difference scheme for the TFDEs. The parallel in time method follows the same spirit as the domain decomposition that consists in breaking the domain of computation into subdomains and solving iteratively the sub-problems over each subdomain in a parallel way. Concretely, the iterative scheme falls in the category of the predictor-corrector scheme, where the predictor is solved by finite difference method in a sequential way, while the corrector is solved by computing the difference between spectral collocation and finite difference method in a parallel way. The solution of the iterative method converges to the solution of the spectral method with high accuracy. Some numerical tests are performed to confirm the efficiency of the method in three areas: (i) convergence behaviors with respect to the discretization parameters are tested; (ii) the overall CPU time in parallel machine is compared with that for solving the original problem by spectral method in a single processor; (iii) for the fixed precision, while the parallel elements grow larger, the iteration number of the parallel method always keep constant, which plays the key role in the efficiency of the time parallel method.

Keywords
Time parareal, spectral collocation method, finite difference method, time fractional differential equations

Introduction
We consider the time fractional differential equations as follows:

\[
\begin{aligned}
0_{D}^{a}u(x,t) - \Delta u(x,t) &= f(x,t) \quad \forall (x,t) \in \Lambda \times I, \\
u(x,t)_{|\partial \Omega} &= 0 \quad \forall t \in I, \\
u(x,0) &= \nu_0(x) \quad \forall x \in \Lambda
\end{aligned}
\]

where \( \Lambda := (a,b), I = (0,T], 0 < \alpha < 1, f \) is a given function. The left Caputo derivative of order \( \alpha \), \( 0_{D}^{a} \), are defined by

\[
0_{D}^{a}u(x,t) := g_{\alpha} \int_{0}^{t} \frac{\partial u(x,s)}{(t-s)^{\alpha}} ds
\]

where \( g_{\alpha} = \frac{1}{\Gamma(1-\alpha)} \), and \( \Gamma(\cdot) \) is the Gamma function.

The presence of the integral in the equation makes the problem “global time dependence”. This means that the solution at a time \( t_k \) depends on the solutions at all previous time \( t < t_k \). The fact that all previous solutions have to be used to compute the solution at the current time level would make the storage very expensive if a low-order method is employed. To overcome the difficulties caused by the low-order methods, some high accuracy methods have been proposed to solve such type of “global time dependence” problems.
such as spectral methods,\textsuperscript{2–7} spectral element methods (SEM),\textsuperscript{8} and references therein. However, the resulted linear system of the SM or SEM is dense due to the non-locality of the fractional operators and the condition number may grow fast. Furthermore, the required number of degrees of the freedom may be rather large due to the singularity of the solutions of FDEs, especially for a time-dependent problem.

To resolve this important issue, we adopt the idea of the time parallel method. The parallel in time algorithm for a model ODE was initially introduced by Lions, Maday, and Turinici\textsuperscript{9} as a numerical method to solve the evolution problems in parallel. It can be interpreted as a predictor-corrector scheme,\textsuperscript{10,11} which involves a prediction step based on a coarse approximation and a correction step computed in parallel based on a fine approximation. For the parallel in time method based on the finite difference scheme, the convergence analysis for a ODE problem was given in Gander and Vandewalle.\textsuperscript{12} For high accuracy parallel method, Li, Tang and Xu\textsuperscript{13} proposed a time parallel based on spectral method for Volterra integral equations and exponential convergence was proved. Xu et al.\textsuperscript{14} proposed a time parallel method based on spectral method for TFDEs.

In this work, we propose a parallel in time method based on a spectral collocation scheme and a finite difference method for the TFDEs. The main goal of this method is to parallelize the time discretization to obtain an important speed up which makes it possible to consider the really long integration problems. The method consists in splitting the time domain into a set of subdomains, and breaking the original problem into a series of independent problems on the time subdomains. The key point of this method lies in the design of an efficient iterative process while each step containing a predictor and a corrector solvers. The predictor is solved by a coarse finite difference approximation. Corrector is solved by computing difference between the fine spectral collocation approximation and the finite difference approximation in parallel way. Here, the finite difference method we adopt in parallel method was proposed by Lin.\textsuperscript{15} The convergence of the finite difference method was proved to be $(\delta t)^{2-s}$ on the time direction. Some numerical tests implemented on the parallel machine are carried out to test the precision of the method and the efficiency of the proposed method.

This paper is organized as follows. In section 2, we construct the parallel in time method based on the spectral collocation scheme and the finite difference method for the underlying equation. Numerical experiments are carried out in section 3. The efficiency for the proposed method under some assumptions is given in section 4.

### The parallel in time algorithm

In order to describe the parallel method for problem (1), we separate the time interval $I = [0, T]$ into $L$ subintervals, determined by the grid points

$$0 = t_0 < t_1 < t_2 < \cdots < t_L = T,$$

with $t_i = i\Delta t$, $\Delta t = \frac{T}{L}$.

We denote this partition by $I = \bigcup_{l=1}^{L} I_l$, $I_l = [t_{l-1}, t_l]$.

Let $u = (u_1, u_2, \cdots, u_L)^T$, where $u_i(x, t)$ is the solution of (1) on the $i$-th element, i.e., $u_i(x, t) = u(x, t), \forall (x, t) \in \Omega_l := \Lambda \times [t_{l-1}, t_l], l = 1, \cdots, L$. It is readily seen that the following system of $L$ separate initial value problems holds:

$$
\begin{align*}
\frac{d}{dt}u_l(x, s) &= f(x, t), (x, t) \in \Omega_l, \\
\frac{d^2}{dt^2}u_l(x, s) &= \Delta u_l(x, t), (x, t) \in \Omega_l.
\end{align*}
$$

To simplify the notation, we introduce the operator $S_l$, such that $u_i(x, t) = S_l(x, t, u_{l-1}, \cdots, u_1)$.

Let $F_l$ be the fine approximation to $S_l$, which will be defined hereafter by a highly accurate spectral collocation method on element $\Omega_l$. To this end, we define $P_M(\Lambda)$ (resp. $P_N(l|l)$) as the polynomial spaces of degree less than or equal to $M$ (resp. $N$) with respect to $x$ (resp. $t$). Let $P_M(\Lambda) := P_M(\Lambda) \cap H_0^1(\Lambda)$. Then we define the spectral approximation space as:

$$SMN(\Omega_l) := P_M(\Lambda) \times P_N(l).$$

Let $\Lambda = [-1, 1]$. We denote by $J_{m}^{\alpha, \beta}(x), -1 \leq x \leq 1$, the Jacobi polynomial of degree $M$ associated with the weights $\omega_{\alpha, \beta}(x) = (1 - x)^{\alpha}(1 + x)^{\beta}$. Let $\chi_{m}^{\alpha, \beta}$ be the points of the Gauss-Jacobi (GJ) quadrature formula, arranged by the increasing order: $-1 < \chi_0^{\alpha, \beta} < \chi_1^{\alpha, \beta} < \cdots < \chi_M^{\alpha, \beta} < 1$. The associated weights of the GJ quadrature formula are denoted by $\omega_{m}^{\alpha, \beta}: 0 \leq m \leq M$. Then
it is well-known the following identity:

$$\int_\Lambda \varphi(x) dx = \sum_{m=0}^{M} \varphi(x_m^\beta) \omega_m^\beta, \quad \forall \varphi \in P_{2M+1}(\Lambda)$$

The discrete $L^2$ inner product associated to the GJ quadrature is denoted by:

$$\langle \varphi, \psi \rangle_{M, \omega^\beta} := \sum_{m=0}^{M} \varphi(x_m^\beta) \psi(x_m^\beta) \omega_m^\beta$$

(3)

Furthermore, we define the Legendre-Gauss (LG) points on the element $I_i = [t_{i-1}, t_i]$, i.e.,

$$\eta_n^i = \frac{t_{i-1} - t_i}{2} x_n^0 + \frac{1}{2} t_{i-1} + t_i, \quad 0 \leq n \leq N, \quad 1 \leq i \leq L$$

and the corresponding weights

$$\rho_n^i = \frac{\Delta t}{2} \omega_n^0, \quad 0 \leq n \leq N, \quad 1 \leq i \leq L$$

The LG points $\xi_m, 0 \leq m \leq M$, on the element $\Lambda$ are defined by a similar way.

Now the fine approximation $F_i$ are defined by the Legendre-collocation method for (2) as follows: find $U_i(x, t) \in S_{MN}(\Omega), i = 1, \cdots, L$, such that, for all $0 \leq m \leq M, 0 \leq n \leq N,$

$$\begin{cases}
\int_{t_{i-1}}^{t_i} \partial_t U_i(\xi_m, s) \frac{ds}{(\eta_n^i - s)^2} - \Delta U_i(x, t)|_{\eta_n^i} = f(\xi_m, \eta_n^i), \\
\int_{t_i}^{t_{i+1}} \partial_t U_i(\xi_m, s) \frac{ds}{(\eta_n^i - s)^2} - \Delta U_i(x, t)|_{\eta_n^i} = f(\xi_m, \eta_n^i) - \int_{t_i}^{t_{i+1}} \partial_t U_i(\xi_m, s) \frac{ds}{(\eta_n^i - s)^2}, \\
\vdots \\
\int_{t_{i-1}}^{t_{i+1}} \partial_t U_i(\xi_m, s) \frac{ds}{(\eta_n^i - s)^2} - \Delta U_i(x, t)|_{\eta_n^i} = f(\xi_m, \eta_n^i) - \sum_{k=1}^{L} \int_{t_{i-1}}^{t_i} \partial_t U_i(\xi_m, s) \frac{ds}{(\eta_n^i - s)^2}
\end{cases}$$

(4)

In the implementation, the integral terms on the left hand sides of (4) are evaluated by using the following Jacobi Gauss quadrature:

$$\int_{\eta_n^i}^{\eta_n^{i+1}} \partial_t U(\xi_m, s) \frac{ds}{(\eta_n^i - s)^2} = \left(\frac{\eta_n^{i+1} - \eta_n^i}{2}\right)^{-2} \int_1^{\frac{1}{2}} \partial_t U(\xi_m, s)(1 - y)^{-2} dy$$

$$\approx \left(\frac{\eta_n^{i+1} - \eta_n^i}{2}\right)^{-2} \left(\partial_t U(\xi_m, \eta_n^i), 1\right)_{N, \omega^\beta}$$

(5)

The right hand sides of (4) are computed in a similar way:

$$\int_{t_{i-1}}^{t_i} \partial_t U_i(\xi_m, s) \frac{ds}{(\eta_n^i - s)^2}$$

$$= \int_{t_{i-1}}^{t_i} \partial_t U_i(\xi_m, s)(\eta_n^i - s(y))^{-2} dy$$

$$\approx \left(\partial_t U(\xi_m, s), (\eta_n^i - s(y))^{-2}\right)_{N, \omega^\beta},$$

where

$$s(y) = \frac{t_i - t_{i-1}}{2} y + \frac{t_i + t_{i-1}}{2}, \quad -1 < y < 1.$$
Now we define the coarse solver $G_l$ by combining the above finite difference in time direction with the Legendre-collocation method in spacial direction for (2) as follows: find $V(x, i_l) \in P_M^0(\Lambda), i = 1, \ldots, qL$, such that, for all $0 \leq m \leq M$,
\[
\begin{align*}
\frac{g_x}{(1-x)\delta t^2} V(\zeta^m, i_l) - \Delta V(\zeta^m, i_l) = f(\zeta^m, i_l) + \sum_{j=1}^{N}(a_j - a_{j+1}) V(\zeta^m, i_l-j) + a_l V(\zeta^m, i_l), & \quad (7) \end{align*}
\]

Then the parallel in time algorithm proposes an approximation to each $U_l(x, t), l = 1, \ldots, L$, as the limit of the sequence $U_l^k(x, t)$ defined by
\[
\begin{align*}
U_l^k = G_l(t, U_{l-1}^k, U_{l-2}^k, \ldots, U_l^0) + F_l(t, U_{l-1}^{k-1}, U_{l-2}^{k-1}, \ldots, U_l^{k-1}) - G_l(t, U_{l-1}^{k-1}, U_{l-2}^{k-1}, \ldots, U_l^{k-1}), & \quad k \geq 1, \quad (8) \end{align*}
\]
with the initial value:
\[
\begin{align*}
U_l^0 = G_l(t, U_{l-1}^0, U_{l-2}^0, \ldots, U_l^0), \quad (9) \end{align*}
\]

where the subscript $l$ refers to the element number, the subscript $k$ refers to the iteration number, and $U_l^k(t)$ represents an approximation of the solution $U_l(t)$ at the time interval $[t_{l-1}, t_l]$.

It is an easy matter to realize first that the method is exact after enough iterations. Indeed, by induction we obtain that $U_l^k = U_l$ for any $l > 0$. However our numerical experiments in Figure 4 show that the convergence of $U_l^k$ to $U_l$ goes much faster as we shall see shortly.

**Numerical results**

In this section, we present some numerical results obtained by the proposed parallel in time scheme. First, we test the convergence behavior with respect to some parameters of the approximation. Second, we test the efficiency of the parallel iterative scheme.

We consider the following example:
\[
\begin{align*}
\frac{\partial^2 u(x, t)}{\partial x^2} - \Delta u(x, t) = f(x, t) \forall (x, t) \in [0, 1] \times [0, T], \quad (10) \end{align*}
\]

with the exact solution $u(x, t) = \sin(\pi x)\sin(\pi t)$ and $\pi = 0.5$. The first purpose is to investigate the convergence behavior of the numerical solutions with respect to the polynomial degrees $M, N$, the iteration number $k$ and the parameter $q$ for the coarse grids. In the top of Figure 1, we plot the $L^\infty$-errors in semi-log scale as a function of $N$ with $M = 15$, iteration number $k = 20$, time domain $T = 30$, $q = 1$. The domain is partitioned into 30 subintervals, $\Delta t = \delta t$. In order to separate different error sources, the solution is computed after large enough iteration steps such that the error produced by the coarse solver is negligible as compared with the error of the fine resolution. As expected, the errors show an exponential decay, since in this semi-log representation one observes that the error variations are linear versus the degrees of polynomial $N$. Similarly, from bottom of the Figure 1, one observes the errors show an exponential decay versus the degrees of polynomial $M$.

Next we investigate the convergence behavior with respect to the iteration number $k$. For a similar reason mentioned above, we now fix a large enough $M = 15$, $N = 15$, and let $k$ vary for different values of $q$. In Figure 2, we plot the error decay with increasing iteration number $k$ for several values of $q$. It is observed that the error curves are all straight lines in this semi-log representation, which means the convergence of order $k$ with respect to the error associated to the coarse resolution.

In Figure 3, we test the convergence behavior with respect to $q$ in log-log scale for three iteration number
\( k = 2, 3, 5 \). The errors show an algebraic decay, since in this log-log representation one observes that the error variations are linear versus \( q = \Delta t \).

The second purpose is to test the efficiency of the scheme in a parallel machine. In the top of Figure 4, for fixed error less than \( 10^{-12} \), we compare the CPU time of iterative scheme in parallel machine with the CPU time of the fine solver in single processor. The blue line shows the time cost with respect to the number of parallel elements \( L \) for solving the problem by scheme (8) while the red line shows CPU time cost for solving the problem by spectral collocation method in sequential way. Meanwhile, in the bottom of Figure 4, we plot the iteration number corresponding to blue line in the left of Figure 4 for fixed error less than \( 10^{-12} \). One observes that the iteration number keeps constant while the number of elements grow larger.

**Parallelism efficiency**

Firstly, the classical sequential scheme based on the fine mesh consists of solving the problems \( F_l(t, U_{l-1}, U_{l-2}, \ldots, U_1) \) consecutively for \( l = 1, 2, \ldots, L \). The computational complexity is equal to the sum of all the elementary operations in \( I_l; l = 1, 2, \ldots, L \). Let’s denote the total computational cost by \( C_F \) and the cost for \( l \)-th subproblem by \( C_{Fl} \), then \( C_F = \sum_{l=1}^{L} C_{Fl} \). For the \( l \)-th subproblem, spectral discretization \( F_l \) consists of evaluating the integral terms on the right hand sides and solving the linear system. The cost of evaluating the integral terms on the right hand sides approximately equal to \( O(N^2M) \), while the cost for solving the linear system approximately equal to \( O((M^2 + N^2)MN) \), where \( O(M^2 + N^2) \) is the number of operations needed for the matrix vector multiplication and \( O(MN) \) is the estimated iteration number required to achieve the convergence of the iterative method. Then \( C_{Fl} = O((M^2 + N^2)MN) + O(N^2M) \). As a result, the total computational complexity of the sequential fine solutions is

\[
C_F = O((M^2 + N^2)MNL) + O(L^2 N^2 M) \tag{11}
\]
If we implement the scheme (8) in a parallel architecture with enough processors, then for each iteration, the time cost of the master process consists of the computational time for solving a sequential set of \(L\) coarse subproblems and the waiting time for solving a fine subproblem on a non-master process. For the coarse subproblem, we apply the finite difference scheme (6) for the time discretization and the spectral method for space discretization. The cost of solving the sequential set of \(L\) coarse subproblems is estimated to be 
\[
\sum_{l=1}^{L} C^l, \text{ where } C^l = O(M^q) \text{ is the cost to solve the } l\text{-th coarse subproblem.}
\]
The cost of solving a single fine subproblem is approximately \(O((M^2 + N^2)MN) + O(N^2M)\). Note also, in the implementation of the parallel in time scheme there is a need to interpolate the solution between the fine mesh and coarse mesh, the cost of which is \(O(LMNq)\). Therefore if \(k\) is the number of iterations required to achieve the desired convergence of the parallel in time algorithm, then the total computational complexity is
\[
k[O(M^qL) + O((M^2 + N^2)MN) + O(LN^2M) + O(LMNq)]
\]
Comparing (11) with (12), we obtain a speed up
\[
k[O(M^qL) + O((M^2 + N^2)MN) + O(LN^2M) + O(LMNq)]
\]
\[
O((M^2 + N^2)MNL) + O(L^2N^2M)
\]

If the number of processors is large enough, i.e., \(L >> M, L >> N\), and the number of degrees of freedom for the coarse solver is far less than that of the fine solver which means \(q\) is small, then the speed up is close to \(O(k)\). From the right of Figure 4, we observe that the iteration number \(k\) keeps constant while the number of elements \(L\) grow larger.

Conclusions
We propose a parallel in time method, combined with the spectral collocation scheme and the fine difference scheme for the TFDEs. As the domain decomposition method, we break the domain of computation into \(L\) subdomains and thus the problem is divided into \(L\) sub-problems consequently. Then we solve these sub-problems by a predictor-corrector scheme in a parallel way. The predictor is solved by a cheap coarse approximation based on the finite difference method in sequential way, while the corrector is solved by the expensive fine approximation based on the spectral collocation method in parallel way. Numerical results are carried out to confirm the efficiency of the parallel method. The solution of the iterative scheme converges to the solution of the spectral method. The errors decay spectrally with respect to the polynomial degrees \(N\) and \(M\), while the errors decay algebraically with respect to the parameter \(q\) of coarse approximation. The convergence of order \(k\) (iteration number) with respect to the error associated to the solution solved by the finite difference method is given in Figure 2. In Figure 4, the CPU time cost is compared between the parallel scheme and the sequential scheme based on the spectral collocation method. The efficiency analysis of the proposed method is given. If the number of processors is large enough, and the number of degrees of freedom for the coarse solver is far less than that of the fine solver which means \(q\) is small, then the speed up is close to \(O(k)\). In the future work, we will try to analyze the convergence of the parallel method for the fractional differential equations theoretically.

Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This article is supported by NSFC, grant no. 11771083 and NSF of Fuzhou University through grant GXRC-18035.

ORCID iD
Xianjuan Li https://orcid.org/0000-0001-5456-4395

References
1. Podlubny I. Fractional differential equations: an introduction to fractional derivatives, fractional differential equations, to methods of their solution and some of their applications. Amsterdam: Elsevier, 1998.
2. Li X and Xu C. A space-time spectral method for the time fractional diffusion equation. *SIAM J Numer Anal* 2009; 47: 2108–2131.
3. Li X and Xu C. Existence and uniqueness of the weak solution of the space-time fractional diffusion equation and a spectral method approximation. *CiCP* 2010; 8: 1016–1051.
4. Mao Z, Chen S and Shen J. Efficient and accurate spectral method using generalized Jacobi functions for solving Riesz fractional differential equations. *Appl Numer Math* 2016; 106: 165–181.
5. Mao Z and Karniadakis GE. A spectral method (of exponential convergence) for singular solutions of the diffusion equation with general two-sided fractional derivative. *SIAM J Numer Anal* 2018; 56: 24-49.
6. Mao Z and Shen J. Efficient spectral–Galerkin methods for fractional partial differential equations with variable coefficients. *J Comput Phys* 2016; 307: 243–261.
7. Zayernouri M and Karniadakis GE. Fractional Sturm–Liouville Eigen-problems: theory and numerical approximation. *J Comput Phys* 2013; 252: 495–517.

8. Zayernouri M and Karniadakis GE. Discontinuous spectral element methods for time-and space-fractional advection equations. *SIAM J Sci Comput* 2014; 36: B684–B707.

9. Lions J, Maday Y and Turinici G. Résolution d’EDP par un schéma en temps parallèle: A. Comptes Rendus de L’Académie Des Sciences-Series I-Mathematics 2001; 332: 661–668.

10. Bal G. Parallelization in time of (stochastic) ordinary differential equations. *Math Meth Anal Num* 2003. https://www.researchgate.net/publication/237456541_PARALLELIZATION_IN_TIME_OF_STOCHASTIC_ORDINARY_DIFFERENTIAL_EQUATIONS

11. Bal G and Maday Y. A parareal time discretization for non-linear PDEs with application to the pricing of an American put. In: *Proceedings of the workshop on domain decomposition, LNCSE Series* 2001; 23:189–201.

12. Gander M and Vandewalle S. Analysis of the parareal time-parallel time-integration method. *SIAM J Sci Comput* 2007; 29: 556–578.

13. Li X, Tang T and Xu C. Parallel in time algorithm with spectral-subdomain enhancement for volterra integral equations. *SIAM J Numer Anal* 2013; 51: 1735–1756.

14. Xu Q, Hesthaven JS and Chen F. A parareal method for time-fractional differential equations. *J Comput Phys* 2015; 293: 173–183.

15. Lin Y, Li X and Xu C. Finite difference/spectral approximations for the fractional cable equation. *Math Comp* 2010; 80: 1369–1396.