# SPACE-TIME BALANCING DOMAIN DECOMPOSITION[*]

SANTIAGO BADIA[†] AND MARC OLM[†]

**Abstract.** In this work, we propose two-level space-time domain decomposition preconditioners for parabolic problems discretized using finite elements. They are motivated as an extension to space-time of balancing domain decomposition by constraints preconditioners. The key ingredients to be defined are the subassembled space and operator, the coarse degrees of freedom (DOFs) in which we want to enforce continuity among subdomains at the preconditioner level, and the transfer operator from the subassembled to the original finite element space. With regard to the subassembled operator, a perturbation of the time derivative is needed to end up with a well-posed preconditioner. The set of coarse DOFs includes the time average (at the space-time subdomain) of classical space constraints plus new constraints between consecutive subdomains in time. Numerical experiments show that the proposed schemes are weakly scalable in time, i.e., we can efficiently exploit increasing computational resources to solve more time steps in the same total elapsed time. Further, the scheme is also weakly space-time scalable, since it leads to asymptotically constant iterations when solving larger problems both in space and time. Excellent wall clock time weak scalability is achieved for space-time parallel solvers on some thousands of cores.

**Key words.** space-time parallelism, domain decomposition, BDDC, preconditioning, scalability

**AMS subject classifications.** 65N55, 65N22, 65F08

**DOI.** 10.1137/16M1074266

**1. Introduction.** At the beginning of the next decade, supercomputers are expected to reach a peak performance of one exaflop/s, which implies a 100 times improvement with respect to current supercomputers. This improvement will not be based on faster processors, but on a much larger number of processors (in a broad sense). This situation will certainly have an impact in large scale computational science and engineering. Parallel algorithms will be required to exhibit much higher levels of concurrency, keeping good scalability properties.

In mesh-based implicit simulations, e.g., finite element (FE), finite volume, or finite difference methods, one ends up with a linear system to be solved. The linear system solve is a bottleneck of the simulation pipeline and weakly scalable algorithms require complex mathematical approaches, like algebraic multigrid (AMG) or multilevel domain decomposition (DD) techniques. When dealing with transient problems, since information always moves forward in time, one can exploit sequentiality. At every time step one has to solve a spatial problem before proceeding to the next step and parallelism can be exploited at the linear system solve. Although parallel-in-time methods are becoming popular, the sequential-in-time approach is the standard procedure in scientific simulations. However, the tremendous amount of parallelism to be

---

[†]Centre Internacional de Mètodes Numèrics a l'Enginyeria (CIMNE), Parc Mediterrani de la Tecnologia, UPC, Esteve Terradas 5, 08860 Castelldefels, Spain and Universitat Politècnica de Catalunya, Jordi Girona 1-3, Edifici C1, 08034 Barcelona, Spain (sbadia@cimne.upc.edu, molm@cimne.upc.edu).

exploited in the near future certainly motivates changing this paradigm, since further concurrency opportunities must be sought.

In transient simulations, a natural way to go is to exploit concurrency not only in space but also in time. The idea is to develop space-time solvers that do not exploit sequentiality (at least at the global level) and thus provide the solution in space at all time values in one shot. Space-time parallelism is a topic that is receiving rapidly increasing attention. Different iterative methods have been considered so far. One approach is to use the Parareal method [16], which is a time-only parallel algorithm, combined with a parallel space preconditioner (see, e.g., [13]). Another space-time algorithm is PFASST [11, 19], which combines a spectral deferred correction time integration with a nonlinear multigrid spatial solver; the viability of the PFASST method has been proved in [20] at JUQUEEN. Weakly scalable space-time AMG methods can be found in [14, 23, 12].

The multilevel balancing DD by constraints (BDDC) preconditioner [18, 22] has recently been proved to be an excellent candidate for extreme scale simulations in [4], where a recursive implementation that permits overlapping among communication and computation at all levels has scaled up to almost half a million cores and two million subdomains (MPI tasks), for both structured and unstructured meshes with tens of billions of elements. The key ingredient of these methods relies on the definition of a FE space with relaxed interelement continuity. These spaces are defined by choosing the quantities to be continuous among processors, i.e., the coarse degrees of freedom (DOFs) [9]. As far as we know, scalable DD methods in space-time have not been considered so far.

In this work, we develop *weakly scalable space-time preconditioners based on BDDC methods.* In order to do that, we extend the key ingredients in the space-parallel BDDC framework, namely, the subassembled space and operator, coarse DOFs, and transfer operators, to space-time. We prove that the resulting method only involves a set of well-posed problems, and time causality can still be exploited at the local level. We have solved a set of linear and nonlinear problems that show the excellent weak scalability of the proposed preconditioners.

The outline of the article is as follows. In section 2 we set the problem and introduce notation. In section 3 we introduce the classical space-parallel BDDC preconditioners. In section 4 we develop space-time BDDC (STBDDC) preconditioners. In section 5 we present a detailed set of numerical experiments showing the scalability properties of the proposed methods. Finally, in section 6 we draw some conclusions.

**2. Problem setting.** In this section, we introduce the problem to be solved, the partition of the domain in space and time, and the space and time discretization. In what follows, calligraphic letters are used for operators. $\mathcal{M}$ denotes mass matrix operators related to the time derivative discretization, $\mathcal{K}$ is used for the rest of the terms in the PDE operator, and $\mathcal{A}$ is used for the sum of these two operators. Given an operator $\mathcal{A}$, we will use the notation $\mathcal{A}(u, v) \doteq \langle \mathcal{A}u, v \rangle$. Uppercase letters $(V, \ldots)$ are used for (FE-type) functional spaces whereas functions are represented by lowercase letters $(v, \ldots)$. We use classical functional analysis notation for Sobolev spaces.

**2.1. Domain partitions.** We consider a bounded space-time domain $\Omega \times (0, T]$, where $\Omega$ is an open polyhedral domain $\Omega \subset \mathbb{R}^d$, $d$ being the space dimension. Let us construct two partitions of $\Omega$, a *fine* partition into *elements* and a *coarse* partition into *subdomains*. The partition of $\Omega$ into elements is represented by $\theta$. In space, elements $e \in \theta$ are tetrahedra/hexahedra for $d = 3$ or triangles/quadrilaterals for $d = 2$. The *coarse* partition $\Theta$ of $\Omega$ into subdomains is obtained by *aggregation of elements* in $\theta$,

i.e., there is an element partition $\theta_\omega \doteq \{e \in \theta : e \subset \omega\} \subset \theta$ for any $\omega \in \Theta$. The interface of the subdomain partition is $\Gamma \doteq \cup_{\omega \in \Theta} \partial\omega \setminus \partial\Omega$.

For the time interval $(0, T]$, we define a time partition $\{0 = t^0, t^1, \ldots, t^K = T\}$ into $K$ time elements. We denote the $k$th element by $\delta_k \doteq (t_{k-1}, t_k]$ for $k = 1, \ldots, K$.

**2.2. Space-time discretization.** Let us consider as a model problem the following transient convection-diffusion–reaction (CDR) equation: find $u \in H^1(\Omega)$ such that

(1)     $\partial_t u - \nabla \cdot \nu\nabla u + \beta \cdot \nabla u + \sigma u = f$   on $\Omega$,   almost everywhere in $(0, T]$,

     $u = g$   in $\partial\Omega$,      $u(0, x) = u^0$,

where $\nu$ and $\sigma$ are positive constants, $\beta \in \mathbb{R}^d$, and $f \in H^{-1}(\Omega)$. We supplement this system with the initial condition $u(0) = u^0$. Homogeneous Dirichlet data are assumed for the sake of clarity in the exposition, but their extension to the general case is obvious. Besides, let us consider $\beta = 0$ and $\sigma = 0$ for simplicity in the exposition of the algorithm. In section 5 we will take a nonlinear viscosity $\nu(u) = \nu_0|\nabla u|^p$ (with $p \geq 0$ and $\nu_0 > 0$), i.e., the transient $p$-Laplacian problem.

For the space discretization, we use $H^1$-conforming FE spaces on conforming meshes with strong imposition of Dirichlet conditions. The discontinuous Galerkin (DG) case will not be considered in this work, but we refer to [10] for the definition of BDDC methods for DG discretizations. We define $\bar{V} \subset H_0^1(\Omega)$ as the global FE space related to the FE mesh $\theta$. Further, we define the FE-wise operators

$$\mathcal{M}_e(u, v) \doteq \int_e uv, \quad \mathcal{K}_e(u, v) \doteq \int_e \nu\nabla u \cdot \nabla v,$$
$$\mathcal{A}_e(u, v) \doteq \mathcal{M}_e(u, v) + |\delta_k|\mathcal{K}_e(u, v).$$

The global FE problem $\bar{\mathcal{A}} : \bar{V} \to \bar{V}$ can be written as the sum of element contributions, i.e.,

$$\bar{\mathcal{A}}(u, v) \doteq \sum_{e \in \theta} \mathcal{A}_e(u, v) \qquad \text{for } u, v \in \bar{V} \qquad \text{(analogously for } \bar{\mathcal{M}} \text{ and } \bar{\mathcal{K}}\text{)}.$$

In time, we make use of a collocation-type method of lines. For the sake of clarity, we will use the backward-Euler time integration scheme. In any case, the resulting preconditioner can readily be applied to any $\theta$-method or Runge–Kutta method. We are interested in solving the following fully discrete system: given $u(t^0) = 0$, find at every time step $k = 1, \ldots, K$ the solution $u(t^k) \in \bar{V}$ of

(2)              $\bar{A}u(t^k) = \bar{\mathcal{M}}u(t^k) + |\delta_k|\bar{\mathcal{K}}u(t^k) = \bar{g}(t^k)$        for any $v \in \bar{V}$,

with $\bar{g}(t^k) \doteq |\delta_k|\bar{f}(t^k) + \bar{\mathcal{M}}u(t^{k-1}) \in \bar{V}'$, where $\bar{V}'$ denotes the dual space of $\bar{V}$. Nonhomogeneous boundary conditions can be enforced by simply modifying the right-hand side at $t^1$, i.e., $\bar{g}(t^1) \doteq |\delta_k|\bar{f}(t^1) + \bar{\mathcal{M}}u^0 \in \bar{V}'$. Such imposition of boundary conditions, i.e., by enforcing homogeneous conditions in the FE space plus the modification of the right-hand side, is better suited for the space-time framework. (We note that this is the way strong Dirichlet boundary conditions are imposed in FE codes.)

**3. Space BDDC preconditioning.** In this section, we present first a parallel solver for the transient problem (2), which combines a sequential-in-time approach

with a space-parallel BDDC preconditioned Krylov solver at every time step [9]. It will serve to introduce space-parallel BDDC methods and related concepts that will be required in the space-time section. BDDC preconditioners involve the definition of three key ingredients: (1) a subassembled problem that involves independent subdomain corrections, (2) a set of coarse DOFs and the corresponding subspace of functions with continuous coarse DOFs among subdomains, and (3) the interior correction and transfer operators. Let us elaborate these ingredients.

**3.1. Subassembled problem.** Nonoverlapping DD preconditioners rely on the definition of a subassembled FE problem, in which contributions between subdomains have not been assembled. In order to do so, at every subdomain $\omega \in \Theta$, we consider the FE space $V_\omega$ associated with the element partition $\theta_\omega$ with homogeneous boundary conditions on $\partial \omega \cap \partial \Omega$. One can define the subdomain operator $\mathcal{A}_\omega(u, v) = \sum_{e \in \theta_\omega} \mathcal{A}_e(u, v)$, for $u, v \in V_\omega$ (analogously for $\mathcal{M}_\omega$ and $\mathcal{K}_\omega$).

Subdomain spaces lead to the subassembled space of functions $V \doteq \Pi_{\omega \in \Theta} V_\omega$. For any $u \in V$, we define its restriction to a subdomain $\omega \in \Theta$ as $u_\omega$. Any function $u \in V$ can then be represented by its unique decomposition into subdomain functions as $\{u_\omega \in V_\omega\}_{\omega \in \Theta}$. We also define the subassembled operator $\mathcal{A}(u, v) \doteq \Pi_{\omega \in \Theta} \mathcal{A}_\omega(u_\omega, v_\omega)$ (analogously for $\mathcal{M}$ and $\mathcal{K}$).

With these definitions, $\bar{V}$ can be understood as the subspace of functions in $V$ that are continuous on the interface $\Gamma$, and $\bar{\mathcal{A}}$ can be interpreted as the Galerkin projection of $\mathcal{A}$ onto $\bar{V}$. We note that $\theta$ and the FE type defines $\bar{V}$, whereas $\Theta$ is also required to define the local spaces $\{V_\omega\}_{\omega \in \Theta}$ and the subassembled space $V$, respectively.

At this point, we can state the following subassembled problem: given $g \in V'$, find $u \in V$ such that

$$(3) \qquad \mathcal{A}_\omega u_\omega = \mathcal{M}_\omega u_\omega + |\delta_k| \mathcal{K}_\omega u_\omega = g_\omega, \quad \omega \in \Theta.$$

With the previous notation, we can write the subassembled problem in a compact manner as $\mathcal{A}u = g$.

**3.2. Coarse DOFs.** A key ingredient in DD preconditioners is to classify the set of nodes of the FE space $\bar{V}$. The interface $\partial e$ of every FE in the mesh $\theta$ can be decomposed into vertices, edges, and faces. By a simple classification of these entities, based on the set of subdomains that contain them, one can also split the interface $\Gamma$ into faces, edges, and vertices (at the subdomain level), that will be called geometrical objects. We represent the set of geometrical objects by $\Lambda$. In all cases, edges and faces are open sets in their corresponding dimension. By construction, faces belong to two subdomains and edges belong to more than two subdomains in three-dimensional problems. This classification of $\Omega$ into objects automatically leads to a partition of interface DOFs into DOF objects, due to the fact that every DOF in an FE does belong to only one geometrical entity. These definitions are heavily used in DD preconditioners (see, e.g., [21, p. 88]).

Next, we associate with some (or all) of these geometrical objects a *coarse* DOF. In BDDC methods, we usually take as coarse DOFs, mean values on a subset of objects $\Lambda_O$. Typical choices of $\Lambda_O$ are $\Lambda_O \doteq \Lambda_C$, when only corners are considered, $\Lambda_O \doteq \Lambda_C \cup \Lambda_E$, when corners and edges are considered, or $\Lambda_O \doteq \Lambda$, when corners, edges, and faces are considered. These choices lead to three common variants of the BDDC method referred as BDDC(c), BDDC(ce), and BDDC(cef), respectively. This classification of DOFs into objects can be restricted to any subdomain $\omega \in \Theta$, leading to the set of subdomain objects $\Lambda_O(\omega)$.

With the classification of the interface nodes and the choice of the objects in $\Lambda_O$, we can define the coarse DOFs and the corresponding BDDC space. Given an object $\lambda \in \Lambda_O(\omega)$, let us define its restriction operator $\tau_\lambda^\omega$ on a function $u \in V_\omega$ as follows: $\tau_\lambda^\omega(u)(\xi) = u(\xi)$ for a node $\xi$ that belongs to the geometrical object $\lambda$, and zero otherwise. We define the BDDC space $\widetilde{V} \subset V$ as the subspace of functions $v \in V$ such that the constraint

$$(4) \qquad \int_\lambda \tau_\lambda^\omega(v_\omega) \qquad \text{is identical for all } \omega \in \text{neigh}(\lambda),$$

where $\text{neigh}(\lambda)$ stands for the set of subdomains that contain the object $\lambda$. (The integral on $\lambda$ is just the value at the vertex, when $\lambda$ is a vertex.) Thus, every $\lambda \in \Lambda_O$ defines a coarse DOF value (4) that is continuous among subdomains. Further, we can define the BDDC operator $\widetilde{\mathcal{A}}$ as the Galerkin projection of $\mathcal{A}$ onto $\widetilde{V}$.

**3.3. Transfer operator.** The next step is to define a transfer operator from the subassembled space $V$ to the continuous space $\bar{V}$. The transfer operator is the composition of a weighting operator and a harmonic extension operator.

1. The weighting operator $\mathcal{W}$ takes a function $u \in V$ and computes mean values on interface nodes, i.e.,

$$(5) \qquad \mathcal{W}u(\xi) \doteq \frac{\sum_{\omega \in \text{neigh}(\xi)} u_\omega(\xi)}{|\text{neigh}(\xi)|},$$

at every node $\xi$ of the FE mesh $\theta$, where $\text{neigh}(\xi)$ stands for the set of subdomains that contain the node $\xi$. It leads to a continuous function $\mathcal{W}u \in \bar{V}$. It is clear that this operator only modifies the DOFs on the interface. Other choices can be defined for nonconstant physical coefficients.

2. Next, let us define the bubble space $V_0 \doteq \{v \in V : v = 0 \text{ on } \Gamma\}$ and the Galerkin projection $\mathcal{A}_0$ of $\mathcal{A}$ onto $V_0$. We also define the trivial injection $\mathcal{I}_0$ from $V_0$ to $\bar{V}$. The *harmonic* extension reads as $\mathcal{E}v \doteq (I - \mathcal{I}_0 \mathcal{A}_0^{-1} \mathcal{I}_0^T \bar{\mathcal{A}})v$. The computation of $\mathcal{A}_0^{-1}$ involves solving problem (3) with homogeneous boundary conditions on $\Gamma$. This operator corrects interior DOFs only.

The transfer operator $\mathcal{Q} : V \to \bar{V}$ is defined as $\mathcal{Q} \doteq \mathcal{E}\mathcal{W}$.

**3.4. Space-parallel preconditioner.** With all these ingredients, we are now in position to define the BDDC preconditioner. This preconditioner is an additive Schwarz preconditioner (see, e.g., [21, Chap. 2]), with corrections in $V_0$ and the BDDC correction in $\widetilde{V}$ with the transfer $\mathcal{Q}$. As a result, the BDDC preconditioner reads as

$$(6) \qquad \mathcal{B} \doteq \mathcal{I}_0 \mathcal{A}_0^{-1} \mathcal{I}_0^T + \mathcal{Q}\widetilde{\mathcal{A}}^{-1}\mathcal{Q}^T.$$

**4. Space-time BDDC preconditioning.** As commented on in section 1, the huge amounts of parallelism of future supercomputers will require scientists to seek additional concurrency. In the simulation of (2) using the space-parallel preconditioner (6), we are using a sequential-in-time approach by exploiting time causality. The objective of this section is to solve (2) at all time steps in one shot, by using a space-time–parallel preconditioner and a Krylov subspace method for nonsymmetric problems. In order to do so, we want to extend the BDDC framework to space-time.

In the following, we will use bold symbols, e.g., $\boldsymbol{u}$, $\boldsymbol{V}$, or $\boldsymbol{\mathcal{A}}$, for space-time functions, functional spaces, and operators, respectively. $\boldsymbol{I}$ is the identity matrix, which can have a different dimension in different appearances.

First, we must start with a space-time partition of $\Omega \times (0,T]$. We consider a time subdomain partition by aggregation of time elements, $\{0 = T_0, T_1, \ldots, T_N = T\}$ into $N$ time subdomains. We denote the $n$th subdomain as $\Delta_n \doteq (T_{n-1}, T_n]$ for $n = 1, \ldots, N$. By definition, $\Delta_n$ admits a partition into $K_n$ time elements $\{T_{n-1} = t_n^0, \ldots, t_n^{K_n} = T_n\}$. Next, we define the space-time subdomain partition as the Cartesian product of the space subdomain partition $\Theta$ and the time subdomain partition defined above; for every space subdomain $\omega$ and time subdomain $\Delta_n$, we have the space-time subdomain $\boldsymbol{\omega}_n \doteq \omega \times \Delta_n$.

The global space of continuous space-time functions in which we want to solve (2) is the FE space $\bar{V}$ of spatial functions times $K+1$ time steps, i.e., $\bar{\boldsymbol{V}} \doteq [\bar{V}]^{K+1}$, constrained to zero initial condition. Thus, by definition, $\boldsymbol{u} \in \bar{\boldsymbol{V}}$ can be expressed as $\boldsymbol{u} = (u(t^0) = 0, u(t^1), \ldots, u(t^K))$, and the original problem (2) (for all time step values) can be stated in a compact manner as

$$(7) \qquad \bar{\boldsymbol{A}}\boldsymbol{u} = \bar{\boldsymbol{f}} \qquad \text{in } \bar{\boldsymbol{V}}.$$

In order to define the STBDDC preconditioner for (7), we will use the same structure as above, extending the three ingredients in section 3 to the space-time case.

**4.1. Subassembled problem.** Using the space-time partition above, the trial (and test) space for the local space-time subdomain $\boldsymbol{\omega}_n$ is $\boldsymbol{V}_{\boldsymbol{\omega}_n} \doteq [V_\omega]^{K_n+1}$. Thus, by definition, $\boldsymbol{u}_{\boldsymbol{\omega}_n} \in \boldsymbol{V}_{\boldsymbol{\omega}_n}$ can be expressed as $\boldsymbol{u}_{\boldsymbol{\omega}_n} = (u_\omega(t_n^0), \ldots, u_\omega(t_n^{K_n}))$. Analogously, the subassembled space is $\boldsymbol{V} \doteq \Pi_{n=1}^N \Pi_{\omega \in \Theta} \boldsymbol{V}_{\boldsymbol{\omega}_n}$. Let us note that, using these definitions, functions in $\boldsymbol{V}$ have duplicated values at $T_n$ for $n = 1, \ldots, N-1$. The global space of continuous space-time functions $\bar{\boldsymbol{V}}$ can be understood as the subspace of functions in $\boldsymbol{V}$ that are continuous on the space-time interface $\Gamma \times T_n$ for $n = 1, \ldots, N-1$.

Now, we propose the following subassembled problem in $\boldsymbol{V}$: find the solution $\boldsymbol{u} \in \boldsymbol{V}$ such that, at every $\boldsymbol{\omega}_n$ in the space-time partition, it satisfies the space-time problem

$$(8) \quad \sum_{k=1}^{K_n} \left\{ \mathcal{M}_\omega(u_\omega(t_n^k) - u_\omega(t_n^{k-1}), v_\omega(t_n^k)) + |\delta_n^k|\mathcal{K}_\omega(u_\omega(t_n^k), v_\omega(t_n^k)) \right\}$$
$$+ \frac{(1 - \text{Kr}_{1,n})}{2}\mathcal{M}_\omega(u_\omega(t_n^0), v_\omega(t_n^0)) - \frac{(1 - \text{Kr}_{N,n})}{2}\mathcal{M}_\omega(u_\omega(t_n^{K_n}), v_\omega(t_n^{K_n}))$$
$$= \sum_{k=1}^{K_n} |\delta_n^k| f_\omega(t_n^k)(v_\omega(t_n^{K_n}))$$

for any $\boldsymbol{v} \in \boldsymbol{V}$, where $\text{Kr}_{i,j}$ is the Kronecker delta. Note that the perturbation terms in the second line of (8) are introduced only on time interfaces, i.e., in the first and last time steps of the time subinterval, as long as the corresponding time step is not a time domain boundary. For subdomains with $n = 1$, and thus $t_n^0 = 0$, the first stabilization term vanishes. Analogously, the second stabilization term vanishes for $n = N$ and $t_n^{K_n} = T$. We can write (8) in a compact manner as

$$(9) \qquad \boldsymbol{A}\boldsymbol{u} = \boldsymbol{f} \qquad \text{in } \boldsymbol{V}.$$

The motivation of the perturbation terms is to have a positive semidefinite subassembled operator. In any case, the perturbation terms are such that, after intersubdomain assembly, we recover the original space, i.e., $\boldsymbol{A}$ is in fact a subassembled operator with respect to $\bar{\boldsymbol{A}}$, since interface perturbations between subdomains cancel out. We prove that these properties hold.

PROPOSITION 4.1. *The Galerkin projection of the subassembled space-time problem* (9) *onto* $\bar{V}$ *reduces to the original problem* (7). *Further, the subassembled operator* $\mathcal{A}$ *is positive definite.*

*Proof.* In order to prove the equivalence, we need to show that $\bar{\mathcal{A}}$ is the Galerkin projection of $\mathcal{A}$ onto $\bar{V}$, which amounts to saying that the perturbation terms vanish for $u \in \bar{V}$. First, we note that the following equality

$$\sum_{n=1}^{N} \left( \frac{(1 - \mathrm{Kr}_{1,n})}{2} u_\omega(t_n^0) - \frac{(1 - \mathrm{Kr}_{N,n})}{2} u_\omega(t_n^{K_n}) \right) = 0$$

holds for functions that are continuous in time, since $u_\omega(t_{n-1}^{K_{n-1}}) = u_\omega(t_n^0)$ for $n = 2, \ldots, N$. On the other hand, multiplying the right hand of (8) against $v = u$, using the fact that $(a - b)a = \frac{1}{2}(a^2 - b^2) + \frac{1}{2}(a - b)^2$, we get

$$(10) \quad \mathcal{A}(u, u) = \frac{1}{2}\|u(T)\|^2 + \sum_{n=1}^{N}\sum_{k=1}^{K} \left( \frac{1}{2}\|u(t_n^k) - u(t_n^{k-1})\|^2 + |\delta_n^k|\mathcal{K}(u(t_n^k), u(t_n^k)) \right).$$

Since $\mathcal{K}$ is positive semidefinite, $\mathcal{A}$ is positive semidefinite. On the other hand, $\mathcal{A}$ is a lower block triangular matrix. Restricted to one subdomain block, it has diagonal blocks $\frac{1}{2}\mathcal{M}_\omega$ at the first time step, $\mathcal{M}_\omega + |\delta_k|\mathcal{K}_\omega$ at intermediate time steps, and $\frac{1}{2}\mathcal{M}_\omega + |\delta_k|\mathcal{K}_\omega$ at the last time step. Since all these matrices are invertible, $\mathcal{A}$ is nonsingular. Further, $\mathcal{A}^T$ is an upper triangular nonsingular matrix. As a result, $\mathcal{A}$ is positive definite. □

**4.2. Coarse DOFs.** Let us define the continuity to be enforced among space-time subdomains. Let us consider a set of space objects $\Lambda_O$ (see section 3). We define $\widetilde{V} \subset V$ as the subspace of functions $v \in V$ such that the constraint

$$(11) \quad \sum_{k=1}^{K_n-1} |\delta_n^k| \int_\lambda \tau_\lambda^\omega(v_\omega(t_n^k)) \qquad \text{is identical for all } \omega \in \mathrm{neigh}(\lambda),$$

holds for every $\lambda \in \Lambda_O$, and

$$(12) \quad \int_\omega v_\omega(t_n^0) = \int_\omega v_\omega(t_{n-1}^{K_{n-1}}) \qquad \text{for all } \omega \in \Theta, \quad n = 2, \ldots, N.$$

The first set of constraints are the mean values of the space constraints in (4) over time subintervals $\Delta_n$. The second constraint enforces continuity between two consecutive-in-time subdomains $\omega_{n-1}$ and $\omega_n$ of the mean value of the function on their corresponding space subdomain $\omega$. The Galerkin projection of $\mathcal{A}$ onto $\widetilde{V}$ is denoted by $\widetilde{\mathcal{A}}$.

Additionally, motivated by a space-time definition of objects, i.e., applying the object generation above to space-time meshes, we could also enforce the continuity of the coarse DOFs

$$(13) \quad \int_\lambda \tau_\lambda^\omega(v_\omega(t_n^0)), \ n = 2, \ldots, N, \ \text{ and } \ \int_\lambda \tau_\lambda^\omega(v_\omega(t_n^{K_n})), \ n = 1, \ldots, N-1,$$

for every $\lambda \in \Lambda_O$. Thus, we are enforcing pointwise-in-time (in comparison to the mean values in (11)) space constraints on time interfaces. Figure 1 illustrates the resulting space-time set of objects where continuity is to be enforced in a subassembled space $V$.
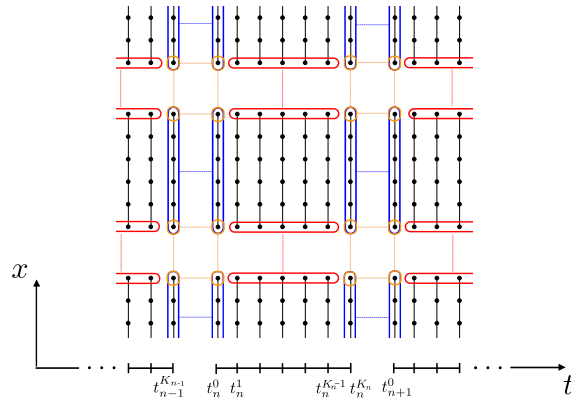
FIG. 1. *Continuity to be enforced among space-time subdomains, for the one-dimensional spatial domain case. The sets of nodes in red are related to the space constraints time averages over time subintervals in* (11)*, the ones in blue are the space mean value constraints on time interfaces in* (12)*, and the ones in orange are spatial constraints on time interfaces in* (13)*.*

**4.3. Transfer operator.** Next, we have to define a transfer operator from $\boldsymbol{V}$ to $\bar{\boldsymbol{V}}$, and the concept of harmonic extension in the space-time setting.

1. In order to define the space-time weighting operator, we make use of the spatial-only definition in (5). Let us define the subdomain restriction of the weighting operator as $\mathcal{W}_\omega u \doteq (\mathcal{W}u)_\omega$. We define the space-time weighting operator restricted to $\boldsymbol{\omega}_n$ as

$$(14) \qquad \boldsymbol{\mathcal{W}}_{\boldsymbol{\omega}_n}\boldsymbol{u} \doteq (\mathcal{W}_\omega u(t_{n-1}^{K_{n-1}}), \mathcal{W}_\omega u(t_n^1), \ldots, \mathcal{W}_\omega u(t_n^{K_n})),$$

and $\boldsymbol{\mathcal{W}} \doteq \Pi_{n=1}^N \Pi_{\omega \in \Theta} \boldsymbol{\mathcal{W}}_{\boldsymbol{\omega}_n} \boldsymbol{u}_{\boldsymbol{\omega}_n}$. We can observe that this weighting operator uses the space-only weighting operator in (5), in order to make the functions continuous in space. On the other hand, on the time interfaces $T_n$ between subdomains $\boldsymbol{\omega}_n$ and $\boldsymbol{\omega}_{n+1}$ (for $n = 1, \ldots, N-1$, where functions in $\boldsymbol{V}$ can be discontinuous in time) we take the value at the preceding subdomain, i.e., $\boldsymbol{\omega}_n$. This choice is motivated by the causality of the problem in time.

2. Next, we define the space-time interior correction. In order to do so, we first define the space-time "bubble" space as $\boldsymbol{V}_0$, where its local component at $\boldsymbol{\omega}_n$ is

$$\boldsymbol{v}_{\boldsymbol{\omega}_n} = (0, v_\omega(t_n^1), \ldots, v_\omega(t_n^{K_n})), \qquad v(\cdot) \in V_0.$$

This definition of $\boldsymbol{V}_0$ naturally arises from the definition of the weighting operator (14). The nodes that are enforced to be zero in $\boldsymbol{V}_0$ are the ones that are modified by (14). $\boldsymbol{\mathcal{I}}_0$ is the trivial injection from $\boldsymbol{V}_0$ to $\bar{\boldsymbol{V}}$ and we denote the Galerkin projection of $\boldsymbol{\mathcal{A}}$ as $\boldsymbol{\mathcal{A}}_0$. Its inverse involves local subdomain problems like (8) with zero initial condition and homogeneous boundary conditions on $\Gamma$. Finally, we define the space-time "harmonic" extension operator as $\boldsymbol{\mathcal{E}}\boldsymbol{v} \doteq (\boldsymbol{I} - \boldsymbol{\mathcal{I}}_0\boldsymbol{\mathcal{A}}_0^{-1}\boldsymbol{\mathcal{I}}_0^T\bar{\boldsymbol{\mathcal{A}}})\boldsymbol{v}$. Functions $\boldsymbol{v} \in \bar{\boldsymbol{V}}$ such that $\boldsymbol{\mathcal{E}}\boldsymbol{v} = \boldsymbol{v}$ are denoted as harmonic functions.

We finally define the transfer operator $\boldsymbol{\mathcal{Q}} : \boldsymbol{V} \to \bar{\boldsymbol{V}}$ as $\boldsymbol{\mathcal{Q}} \doteq \boldsymbol{\mathcal{E}}\boldsymbol{\mathcal{W}}$.

**4.4. Space-time–parallel preconditioner.** After extending the previous ingredients to space-time, we can now define the STBDDC preconditioner as

$$(15) \qquad \boldsymbol{\mathcal{B}} \doteq \boldsymbol{\mathcal{I}}_0\boldsymbol{\mathcal{A}}_0^{-1}\boldsymbol{\mathcal{I}}_0^T + \boldsymbol{\mathcal{Q}}\widetilde{\boldsymbol{\mathcal{A}}}^{-1}\boldsymbol{\mathcal{Q}}^T.$$

In the following section, we will analyze the quality of $\boldsymbol{\mathcal{B}}$ as a preconditioner for $\bar{\boldsymbol{\mathcal{A}}}$. We are particularly interested in the weak scalability properties of the preconditioner. Again, this preconditioner can be cast in the additive Schwarz theory.

**4.5. Implementation aspects.** Let us make some comments about the efficient implementation of the STBDDC preconditioner. We want to solve system (2) (or, equivalently, (7)) for all time steps in one shot by using a Krylov iterative solver preconditioned with the STBDDC preconditioner (15). As usual in DD preconditioning, it is common to take as initial guess for the Krylov solver the interior correction $\boldsymbol{u}^0 = \boldsymbol{\mathcal{I}}_0 \boldsymbol{\mathcal{A}}_0^{-1} \boldsymbol{\mathcal{I}}_0^T \boldsymbol{f}$. In this case, it can be proved by induction that applying a Krylov method with $\boldsymbol{\mathcal{B}}$ as a preconditioner will give at each iterate $\boldsymbol{V}_0$-orthogonal residuals of the original problem (7) and harmonic directions (see, e.g., [17]). Thus, the application of the BDDC preconditioner applied to $\boldsymbol{r} \in \boldsymbol{V}'$ such that $\boldsymbol{r} \perp \boldsymbol{V}_0$ can be simplified as

$$\boldsymbol{\mathcal{B}} \boldsymbol{r} = \boldsymbol{\mathcal{E}} \boldsymbol{\mathcal{W}} \widetilde{\boldsymbol{\mathcal{A}}}^{-1} \boldsymbol{\mathcal{W}}^T \boldsymbol{r}.$$

It involves the following steps.

1. Compute $\boldsymbol{s} \doteq \boldsymbol{\mathcal{W}}^T \boldsymbol{r}$. By the definition in (14), the restriction of $\boldsymbol{s} = \boldsymbol{\mathcal{W}}^T \boldsymbol{r}$ to $\boldsymbol{\omega}_n$ is $\boldsymbol{s}_{\boldsymbol{\omega}_n} = (0, \mathcal{W}_\omega^T r(t_n^1), \ldots, \mathcal{W}_\omega^T r(t_n^{K_n}))$, where $\mathcal{W}_\omega = \text{diag}(1/|\text{neigh}(\xi)|)$. This operation implies nearest neighbor communications only.

2. Compute $\widetilde{\boldsymbol{\mathcal{A}}}^{-1} \boldsymbol{s}$. In order to compute this problem, we first use the following decomposition of $\widetilde{\boldsymbol{V}}$ into the subspaces $\widetilde{\boldsymbol{V}}_F$ and $\widetilde{\boldsymbol{V}}_C$. $\widetilde{\boldsymbol{V}}_F$ is the set of functions that vanish on the coarse DOFs (11)–(13). $\widetilde{\boldsymbol{V}}_C$ is the complement of $\widetilde{\boldsymbol{V}}_F$, which provides the values on the coarse DOFs. We define $\widetilde{\boldsymbol{V}}_C$ as the span of the columns of $\boldsymbol{\Phi}$, where $\boldsymbol{\Phi}$ is the solution of

$$(16) \qquad \begin{bmatrix} \boldsymbol{\mathcal{A}}_{\boldsymbol{\omega}_n} & \boldsymbol{\mathcal{C}}_{\boldsymbol{\omega}_n}^T \\ \boldsymbol{\mathcal{C}}_{\boldsymbol{\omega}_n} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Phi}_{\boldsymbol{\omega}_n} \\ \boldsymbol{l}_{\boldsymbol{\omega}_n} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{I} \end{bmatrix},$$

where we have introduced the notation $\boldsymbol{\mathcal{C}}_{\boldsymbol{\omega}_n}$ for the matrix associated with the coarse DOFs constraints. We can check that (see [5, p. 206] for the symmetric case) (1) $\widetilde{\boldsymbol{V}}_F \perp_{\boldsymbol{\mathcal{A}}} \widetilde{\boldsymbol{V}}_C$, (2) $\widetilde{\boldsymbol{V}} = \widetilde{\boldsymbol{V}}_F \oplus \widetilde{\boldsymbol{V}}_C$. The local problems in (16) are indefinite (and couple all time steps in one subdomain). In order to be able to use sequential-in-time local solvers and sparse direct methods for positive-definite matrices, we propose the following approach (see [9] for the space-parallel BDDC preconditioner). Using the fact that $\boldsymbol{\mathcal{A}}_{\boldsymbol{\omega}_n}$ is nonsingular (see Proposition 4.1), we can solve (16) using the Schur complement:

$$(17) \qquad -\boldsymbol{\mathcal{C}}_{\boldsymbol{\omega}_n} \boldsymbol{\mathcal{A}}_{\boldsymbol{\omega}_n}^{-1} \boldsymbol{\mathcal{C}}_{\boldsymbol{\omega}_n}^T \boldsymbol{l}_{\boldsymbol{\omega}_n} = \boldsymbol{I}, \qquad \boldsymbol{\Phi}_{\boldsymbol{\omega}_n} = -\boldsymbol{\mathcal{A}}_{\boldsymbol{\omega}_n}^{-1} \boldsymbol{\mathcal{C}}_{\boldsymbol{\omega}_n}^T \boldsymbol{l}_{\boldsymbol{\omega}_n}.$$

Further, for nonsymmetric problems (as the space-time problem considered herein), we also require defining $\widetilde{\boldsymbol{V}}_C^*$ as the span of the columns of $\boldsymbol{\Psi}$, where $\boldsymbol{\Psi}$ is the solution of

$$\begin{bmatrix} \boldsymbol{\mathcal{A}}_{\boldsymbol{\omega}_n}^T & \boldsymbol{\mathcal{C}}_{\boldsymbol{\omega}_n}^T \\ \boldsymbol{\mathcal{C}}_{\boldsymbol{\omega}_n} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Psi}_{\boldsymbol{\omega}_n} \\ \boldsymbol{l}_{\boldsymbol{\omega}_n} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{I} \end{bmatrix}.$$

This problem is similar to (16), but replacing $\boldsymbol{\mathcal{A}}_{\boldsymbol{\omega}_n}$ by $\boldsymbol{\mathcal{A}}_{\boldsymbol{\omega}_n}^T$ ($\boldsymbol{\mathcal{A}}_{\boldsymbol{\omega}_n}^T$ is an upper triangular nonsingular matrix from Proposition 4.1). Thus, we can use the Schur complement approach (as in (17)) to exploit sequentiality (backward in time) for the local problems.

With these spaces, the original problem to be solved, $\widetilde{\boldsymbol{\mathcal{A}}}\boldsymbol{u} = \boldsymbol{s}$, can be written as find $\boldsymbol{u} = \boldsymbol{u}_F + \boldsymbol{u}_C \in \widetilde{\boldsymbol{V}}$, where $\boldsymbol{u}_F \in \widetilde{\boldsymbol{V}}_F$ and $\boldsymbol{u}_C \in \widetilde{\boldsymbol{V}}_C$ satisfy

$$\boldsymbol{\mathcal{A}}(\boldsymbol{u}_F, \boldsymbol{v}_F) + \boldsymbol{\mathcal{A}}(\boldsymbol{u}_C, \boldsymbol{v}_C^*) = (\boldsymbol{s}, \boldsymbol{v}_F) + (\boldsymbol{s}, \boldsymbol{v}_C^*) \text{ for any } \boldsymbol{v}_F \in \widetilde{\boldsymbol{V}}_F,\ \boldsymbol{v}_C^* \in \widetilde{\boldsymbol{V}}_C^*,$$

where we have used the orthogonality property

$$\boldsymbol{\mathcal{A}}(\boldsymbol{u}_F + \boldsymbol{u}_C, \boldsymbol{v}_F + \boldsymbol{v}_C^*) = \boldsymbol{\mathcal{A}}(\boldsymbol{u}_F, \boldsymbol{v}_F) + \boldsymbol{\mathcal{A}}(\boldsymbol{u}_C, \boldsymbol{v}_C^*).$$

Thus, it involves a fine problem and a coarse problem that are independent. The computation of the fine problem has the same structure as (16) (with a different forcing term), and can be computed using the Schur complement approach in (17). The Petrov–Galerkin-type coarse problem couples all subdomains and is a basis for having a weakly scalable preconditioner. Its assembly, factorization, and solution is centralized in one processor or a subset of processors.

Summarizing, the STBDDC preconditioner can be implemented in such a way that standard sequential-in-time solvers can still be applied for the local problems. Due to the fact that coarse and fine problems are independent, we can exploit an overlapping implementation, in which computations at fine/coarse levels are performed in parallel. This implementation has been proved to be very effective at extreme scales for space-parallel BDDC solvers in [2, 4, 3]. The implementation of the STBDDC preconditioner used in section 5 also exploits this overlapping strategy.

**5. Numerical experiments.** In this section we evaluate the weak scalability for the CDR problem (1) of the proposed STBDDC preconditioner, when combined with the right-preconditioned version of the iterative Krylov-subspace method GMRES. The STBDDC-GMRES solver is tested with the two-dimensional (2D) CDR PDE on regular domains. Domains are discretized with structured Q1 FE meshes and backward-Euler time integration is performed with a constant step size $|\delta_k|$. As performance metrics, we focus on the number of STBDDC preconditioned GMRES iterations required for convergence, and the total computation time. This time will include both preconditioner setup and the preconditioned iterative solution of the linear system in all the experiments reported. The nonlinear case is linearized with a Picard algorithm and a relaxation factor of $\alpha = 0.75$. The stopping criterion for the iterative linear solver is the reduction of the initial residual algebraic $\ell_2$-norm by a factor $10^{-6}$. The nonlinear Picard algorithm stopping criterion is the reduction of the algebraic $\ell_2$-norm of the nonlinear residual below $10^{-3}$.

The problem to be solved is the CDR problem (1). We may consider the Poisson problem for $\beta = (0,0)$ and $\sigma = 0$. Further, we will also tackle the $p$-Laplacian problem, by taking the nonlinear viscosity $\nu(u) = \nu_0 |\nabla u|^p$ (with $p \geq 0$ and $\nu_0 > 0$), $\beta = (0,0)$, and $\sigma = 0$.

**5.1. Experimental framework.** The novel techniques proposed in this paper for the STBDDC-GMRES solver have been implemented in FEMPAR. FEMPAR, developed by the Large Scale Scientific Computing (LSSC) team at CIMNE-UPC, is a parallel hybrid OpenMP/MPI, object-oriented software package for the massively parallel FE simulation of multiphysics problems governed by PDEs. Among other features, it provides the basic tools for the efficient parallel distributed-memory implementation of substructuring DD solvers [1, 2, 4]. The parallel codes in FEMPAR heavily use standard computational kernels provided by (highly efficient vendor implementations of) BLAS and LAPACK. Besides, through proper interfaces to several

third party libraries, the local constrained Neumann problems and the global coarse-grid problem can be solved via sparse direct solvers. FEMPAR is released under the GNU GPL v3 license, and is more than 200K lines of Fortran95/2003/2008 code long. Here, we use the overlapped BDDC implementation proposed in [2], with excellent scalability properties. It is based on the overlapped computation of coarse and fine duties. As long as coarse duties can be fully overlapped with fine duties, perfect weak scalability can be attained. We refer to [2] and [4] for more details. Results reported in this section were obtained on two different distributed-memory platforms: the Gottfried complex of the HLRN-III Cray system, located in Hannover (Germany), and the MareNostrum III, in the Barcelona Supercomputing Centre. In all cases, we consider a one-to-one mapping among subdomains, cores, and MPI tasks, plus one additional core for the coarse problem (see [1, 2] for details).

**5.2. Weak scalability setting.** In computer science parlance, weak scalability is related to how the solution time varies with the number of processors for a fixed problem size per processor. When the time remains asymptotically constant, we say that we have a scalable algorithm. When we consider problems that are obtained after discretization of differential operators, the concept of weak scalability is suitable *as soon as* the relation between the different terms in the (discretization of the) PDE remains constant in the weak scalability analysis. This is the case in most scalability analyses of PDE solvers, which usually deal with steady Poisson or linear elasticity problems. However, the situation becomes more involved as one faces more complicated problems, that combine multiple differential terms of different natures. The simplest example is the CDR equation (1). One can consider a fixed domain $\Omega$ and fixed physical properties, and produce a weak scalability analysis by increasing the number of elements (i.e., reducing $h$) in FEs, and the number of subdomains (i.e., reducing $H$) in the same proportion. However, as we go to larger scales, the problem to be solved tends to a simple Poisson problem (convective terms are $\mathcal{O}(1/h)$ whereas diffusive terms are $\mathcal{O}(1/h^2)$). The same situation happens for space-only parallelization of transient problems because the CFL changes in the scalability analysis. This situation has already been identified in [12, 8], leading to what the authors call CFL-constant scalability. In these simulations, the CFL is constant, but still, spatial differential terms can change their relative weight in the scalability analysis, e.g., one keeps the convective CFL, i.e., $\mathrm{CFL}_\beta = |\beta|\frac{|\delta|}{h}$, but not the diffusive CFL, i.e., $\mathrm{CFL}_\nu = \nu\frac{|\delta|}{h^2}$ (see [8]).

Weak scalability analysis of PDE solvers should be such that the relative weight of all the discrete differential operators is kept. To do that, we keep fixed the physical problem to be solved (boundary conditions, physical properties, etc.), the FE mesh size $h$, and the subdomain size $H$, but increase (by scaling) the physical domain $\Omega \rightarrow \alpha\Omega$ and, subsequently, the number of subdomains and FEs. Let us consider that $\Omega = [0,1]^d$, an FE mesh of size $h = (1/n_h)^d$, and a subdomain size $H = (1/n_H)^d$. Now, we consider $\Omega' = \alpha\Omega = [0,\alpha]^d$, $\alpha \in \mathbb{N}^+$. The FE partition now must involve $\alpha n_h$ FE partitions per dimension ($\alpha^d n_h^d$ FEs) and $\alpha n_H$ subdomain partitions per dimension ($\alpha^d n_H^d$ subdomains). It is also possible to apply this approach to unstructured meshes and space-time domains. Weak scalability in the sense proposed herein is not only about the capability to solve larger problems but also more complicated ones; e.g., we keep fixed the local Reynolds or Péclet number or CFLs, but we increase the global Reynolds or Péclet number, facing not only a larger problem but also a harder one, in general. We have used this definition of weak scalability for PDE solvers in the numerical experiments below for time and space-time parallel solvers.

**5.3. Weak scalability in time.** In this case, the spatial domain is not partitioned and only the time integration is distributed through $P_t$ processors. This fact leads to enforced continuity of mean values of the function on the spatial domain $\Omega$ on time interfaces, i.e., constraint (12) with $\omega = \Omega$. In order to maintain a constant CFL number, the original time interval $(0, T]$ is scaled with the number of processors, i.e., $T' = P_t T$. As a result, using $P_t$ processors we solve a $P_t$ times larger time domain (and time steps). Note that with this approach neither $|\delta_k|$ nor $|\Delta_n|$ are modified through the analysis.

**5.3.1. Time-parallel Poisson solver.** Consider the transient Poisson equation ((1) with $\beta = (0, 0)$ and $\sigma = 0$) with $\nu = 1$ on the unit square spatial domain $\Omega = [0, 1]^2$ and $T = 1$. The source term $f$ is chosen such that $u(x, t) = \sin(\pi x) \sin(\pi y) \sin(\pi t)$ is the solution of the problem. Homogeneous Dirichlet boundary conditions and zero initial condition are imposed. We perform the weak scalability analysis of the TBDDC-GMRES solver with element size $h = \frac{1}{30}$ and several values of $K_n = \frac{|\Delta_n|}{|\delta_k|} = \{10, 15, 30, 60\}$.

Figure 2 reports the weak scalability analysis of the TBDDC-GMRES solver for this experiment. While $h$ is kept fixed, we evaluate different values of $K_n$ and $|\Delta_n|$, which lead to a wide range of diffusive CFLs, shown in 2(a), 2(b), and 2(c) .

Out of these plots, we can draw some conclusions. First, for a fixed local problem size and physical coefficients, reducing the diffusive CFL by reducing the time step size results in more iterations. Second, and most salient, the algorithm is in fact weakly scalable. In fact, for a fixed local problem size and diffusive CFL, as one increases the number of processors, i.e., computes more time steps, the number of iterations is asymptotically constant. In this range, the overlapped fine/coarse strategy leads to perfect weak scalability for time-parallel solvers too. As a result, this analysis shows the capability of the method to compute X times more time steps with X times more cores for the same total elapsed time, which is the main motivation of time parallelism.

**5.3.2. Time-parallel CDR solver.** Consider now the transient CDR equation (1) with $\nu = 10^{-3}$ and $\sigma = 10^{-4}$ on $\Omega = [0, 1]^2$ with homogeneous Dirichlet boundary conditions and null initial solution. We take $T = \frac{1}{10}$. In order to show results for different convective CFL ranges, two convection velocity fields are analyzed, namely, $\beta = (1, 0)$ and $\beta = (10, 0)$. The CFL values shown are those corresponding to the convective term since it is more restrictive than the diffusive CFL in all the cases being considered. The Streamline-Upwinding Petrov–Galerkin (SUPG) stabilization technique is employed (see [6]).

We perform the study with several values of $K_n = \frac{|\Delta_n|}{|\delta_k|} = \{10, 30, 60\}$, which lead to different convective CFLs. For the first case (Figure 3), the source term is chosen such that the function $u(x, t) = \sin(\pi x) \sin(\pi y) \sin(\pi t)$ is the solution of the problem. For the second test (Figure 4), we take $f = 1$, with a boundary layer formation.

Out of these plots we can extract the same conclusions from Figures 3 and 4 as for the Poisson problem. The method is weakly scalable in time for transient CDR problems.

**5.4. Weak scalability in space-time.** The STBDDC preconditioner is considered here, for the set of constraints (11)–(13). In this case, the spatial domain $\Omega$ is scaled by $P_x$ and $P_y$ in the corresponding directions, where $P_x = P_y$ in all cases. On the other hand, the time interval $(0, T]$ is scaled by $P_t$, leading to a $P = P_x \times P_y \times P_t$ partition of the scaled space-time domain $P_x \Omega \times P_t(0, T]$. Therefore, the relative weight of the operators is kept constant through a weak scalability analysis. Local problem loads will be given by $\frac{H}{h}$ in space and $K_n = \frac{|\Delta_n|}{|\delta_k|}$ in time, i.e., $(\frac{H}{h})^d \times \frac{|\Delta_n|}{|\delta_k|}$.
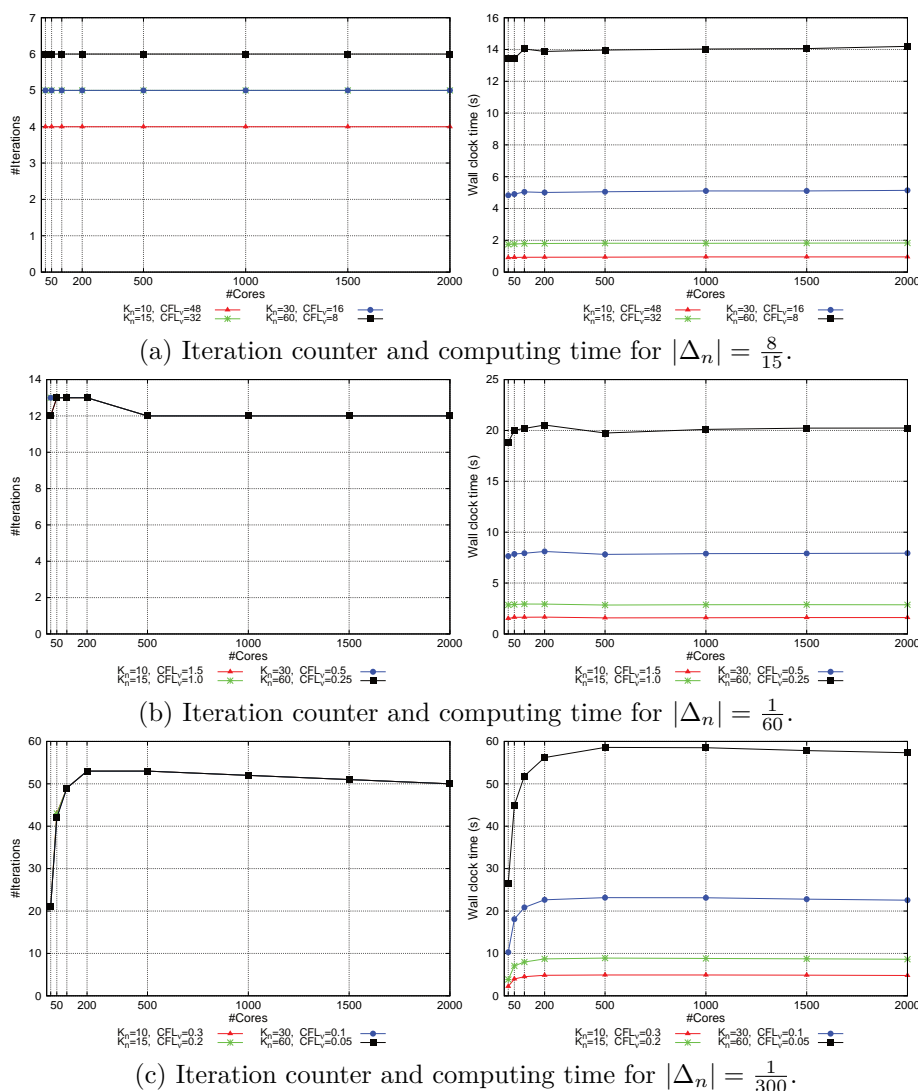
(a) Iteration counter and computing time for $|\Delta_n| = \frac{8}{15}$.



(b) Iteration counter and computing time for $|\Delta_n| = \frac{1}{60}$.



(c) Iteration counter and computing time for $|\Delta_n| = \frac{1}{300}$.

FIG. 2. *Weak scalability for the iterations count (left) and total elapsed time (right) of the TBDDC-GMRES solver in the solution of the unsteady* 2D *Poisson problem on HLRN. Fixed element size $h = 1/30$ with time partition on $P_t$ subdomains.*

**5.4.1. Space-time Poisson solver.** Consider the Poisson problem ((1) with $\beta = (0,0)$ and $\sigma = 0$) with $\nu = 1$, $f = 1$, and homogeneous Dirichlet boundary conditions and zero initial condition. The original spatial domain is $\Omega = [0,1]^2$ while the original time domain is $(0, 0.1]$.

Figure 5 shows weak scalability results for the STBDDC-GMRES solver. Number of iterations (left) and total elapsed times (right) have been reported for three different ratios between spatial and time partitions $\frac{P_x}{P_t}$ in 5(a), 5(b), and 5(c). Also, at every figure, three different local problem sizes, and thus diffusive CFLs, have been considered.

The most salient information from these figures is the fact that the scheme is also scalable in space-time simulations. Here, one is not only solving a larger problem
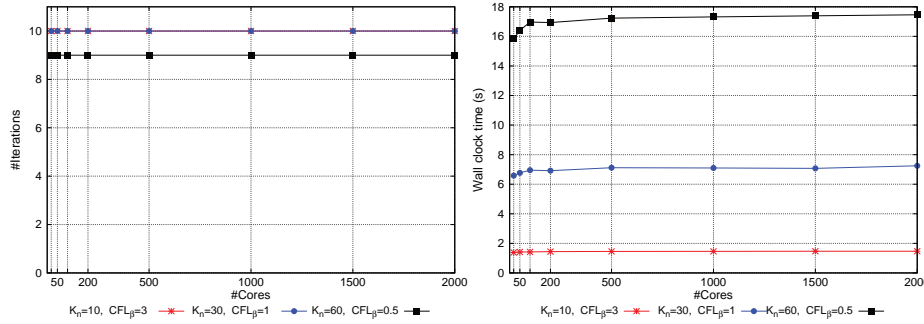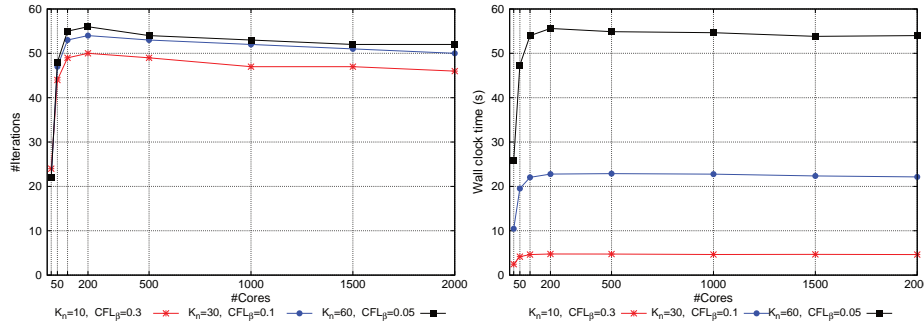
(a) Iteration counter and computing time with $\beta = (10, 0)$.



(b) Iteration counter and computing time with $\beta = (1, 0)$.

Fig. 3. *Weak scalability for the total elapsed time (right) and number of GMRES iterations (left) of the TBDDC solver in the solution of the 2D CDR equation on HLRN (sinusoidal solution). Fixed element size to $h = 1/30$.*
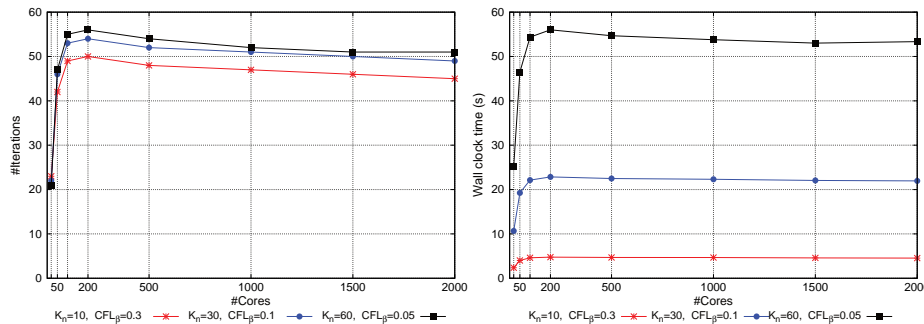


Fig. 4. *Weak scalability for the total elapsed time (right) and number of GMRES iterations (left) of the TBDDC solver in the solution of the 2D CDR equation (boundary layer) on HLRN. Fixed element size to $h = 1/30$.*

in time (as above) but also in space. This result is not surprising since the spatial BDDC preconditioner is known to be weakly scalable and the time-parallel version has also been proved to be weakly scalable in section 5.3. The influence of $\frac{P_x}{P_t}$ on the number of iterations is very mild; also, the effect of the diffusive CFL is mild in this case. The overlapping strategy is fully effective in the range under consideration, because perfect weak scalability can be observed. The effect of the diffusive CFL for a fixed local problem size, obtained by multiplying the viscosity by 10, is reported
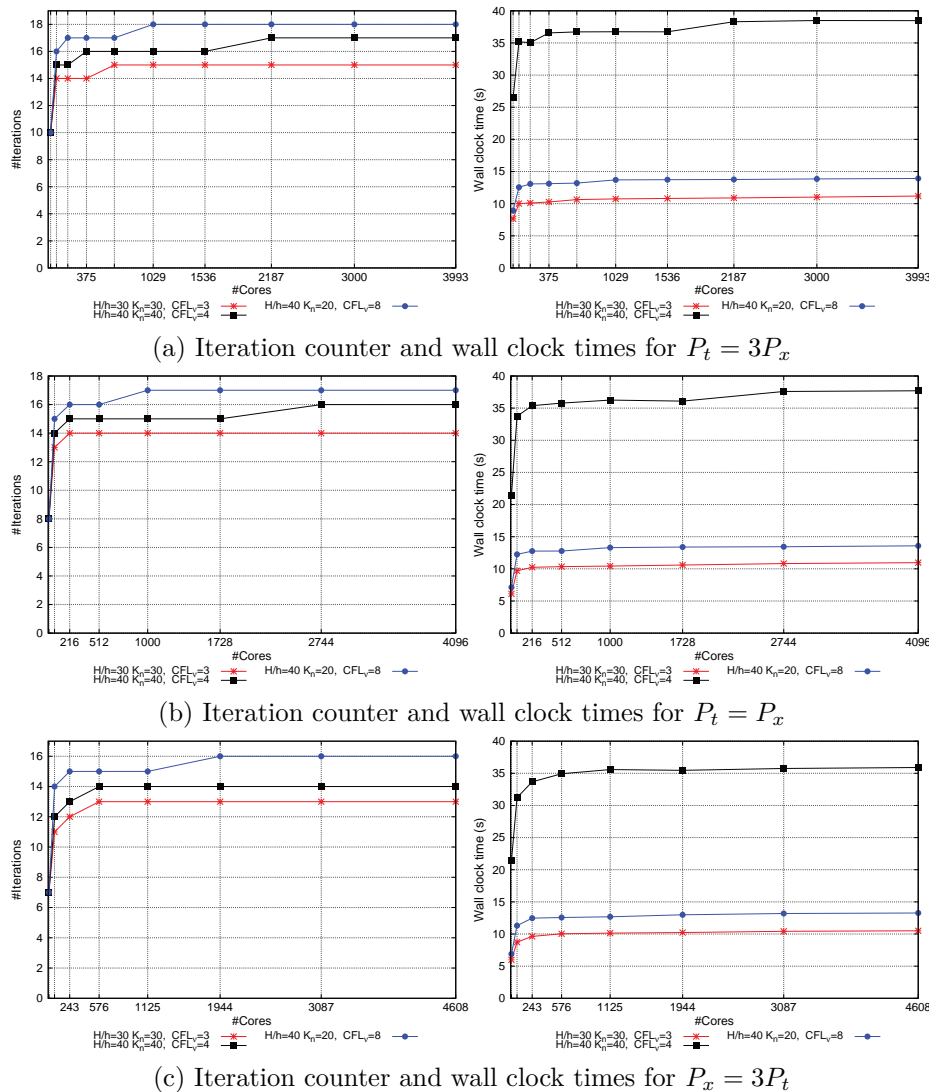
(a) Iteration counter and wall clock times for $P_t = 3P_x$



(b) Iteration counter and wall clock times for $P_t = P_x$



(c) Iteration counter and wall clock times for $P_x = 3P_t$

FIG. 5. *Weak scalability for the total elapsed time (right) and number of GMRES iterations (left) of the STBDDC solver in the solution of the* 2D *Poisson problem on HLRN.*

in Figure 6. In this case, a larger diffusive CFL leads to more iterations but weak scalability is also achieved.

Next, we want to compare the space-time solver against a sequential-in-time approach. We fix the time step size to $|\delta_k| = 10^{-3}$ and $\frac{H}{h} = 30$. Thus, the time interval is $T = K|\delta_k|$, when considering $K$ time steps. The sequential approach makes use of $P_s = P_x \times P_y = 4^2$ processors (space-parallelism only) to solve recursively the spatial problem for increasing values of $K$. The space-time approach is using $P_s \times P_t = 4^2 P_t$ processors to solve the same problem, with a local number of time steps $K_n = 10$. The motivation for such analysis is to assess the benefit of time parallelism in linear problems when spatial parallelism cannot be further exploited efficiently due to very low load per processor.
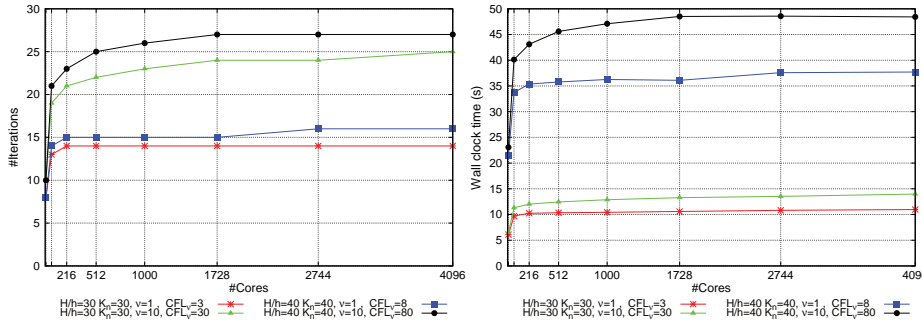
FIG. 6. *Weak scalability for the total elapsed time (right) and number of GMRES iterations (left) of the STBDDC solver in the solution of the 2D Poisson problem on HLRN. Partition is done equally in time and space, i.e., $P_x = P_t$.*
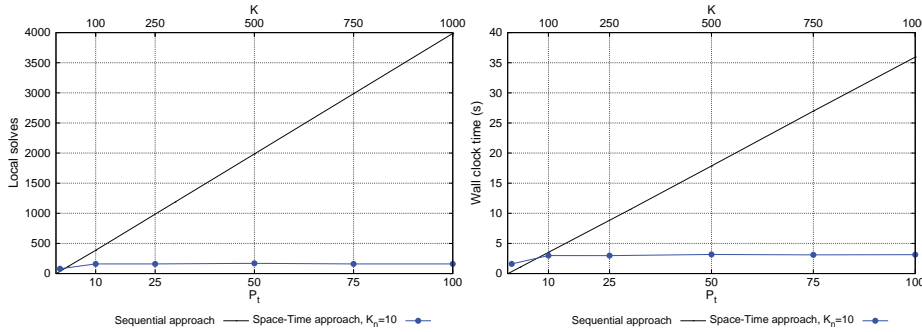


FIG. 7. *Comparison between the sequential and space-time solvers for the transient Poisson problem on MareNostrum supercomputer. Spatial partition is fixed to $P_x = P_y = 4$. The space-time approach is using $P = 4^2 P_t$ MPI fine-level tasks.*

Figure 7 shows the comparison between the sequential approach and the STBDDC preconditioned space-time solver up to $K = 1000$ time steps. The theoretical cost of the sequential approach is proportional to $K$ steps times the elapsed time of the $A_\omega^{-1}$ local solve (a preconditioned GMRES iteration). On the other hand, the theoretical cost of the space-time solver is proportional to the cost of the $A_{\omega_n}^{-1}$ local space-time solves, and in turn, $K_n$ times spatial $A_\omega^{-1}$ local solves (exploiting locally sequentiality in time). The number of local solves is plotted for both sequential and space-time approaches in Figure 7(a). The sequential approach shows a linear growth of the computing time, as expected, since it is solving $K$ times problems in a sequential fashion. Since the current implementation of space-time preconditioners in FEMPAR does not exploit local sequentiality in time, we observe a discrepancy between the intersection of curves in terms of local solves and elapsed time, due to the quadratic complexity of sparse direct methods. In any case, the space-time approach starts to be competitive with less than 10 time partitions. Out of these plots, we are able to reduce the time to solution of simulations with the space-time approach by adding more processors to exploit time parallelism, both for the linear and nonliner problems considered herein. Besides, the method shows excellent weak scalability properties.

**5.4.2. Space-time CDR solver.** Consider the CDR equation (1) with $\sigma = 10^{-4}$, $\beta = (1, 0)$, and $f = 1$, on an original domain $\Omega \times (0, T] = [0, 0.3]^2 \times (0, 0.3]$,

TABLE 1
*Iteration count for CDR equation. Convective CFL equal to* 1.0 *in all cases. CFL$_\nu$ represents the diffusive CFL.*

| $\nu =$ | | 1 | $10^{-1}$ | $10^{-2}$ | $10^{-3}$ | $10^{-4}$ | $10^{-6}$ |
|---|---|---|---|---|---|---|---|
| CFL$_\nu=$ | | $10^2$ | 10 | 1 | $10^{-1}$ | $10^{-2}$ | $10^{-4}$ |
| Péclet= | | $5 \cdot 10^{-3}$ | $5 \cdot 10^{-2}$ | 0.5 | 5 | 50 | $5 \cdot 10^3$ |
| $(P_x \times P_y) \times P_t$ | #Sbd | | | | | | |
| $(3 \times 3) \times 1$ | 9 | 18 | 11 | 7 | 5 | 5 | 5 |
| $(6 \times 6) \times 2$ | 72 | 28 | 16 | 11 | 11 | 11 | 11 |
| $(9 \times 9) \times 3$ | 243 | 35 | 17 | 11 | 12 | 12 | 13 |
| $(12 \times 12) \times 4$ | 576 | 37 | 17 | 12 | 13 | 14 | 15 |
| $(15 \times 15) \times 5$ | 1125 | 38 | 17 | 13 | 14 | 15 | 17 |
| $(18 \times 18) \times 6$ | 1944 | 39 | 17 | 13 | 15 | 15 | 18 |
| $(21 \times 21) \times 7$ | 3087 | 40 | 17 | 14 | 16 | 16 | 19 |
| $(24 \times 24) \times 8$ | 4608 | 41 | 18 | 14 | 17 | 17 | 21 |
| $(27 \times 27) \times 9$ | 6561 | 41 | 18 | 15 | 18 | 18 | 22 |
| $(30 \times 30) \times 10$ | 9000 | 42 | 18 | 16 | 19 | 19 | 24 |

and scaled through the weak scalability analysis by $P_x, P_t$, respectively. Homogeneous Dirichlet and initial conditions are enforced. Local problem size is fixed with $K_n = 30$ and $\frac{H}{h} = 30$. The ratio between spatial and time partitions is $\frac{P_x}{P_t} = 3$. Several diffusion parameters are considered in order to present different scenarios: from a diffusion-dominated case ($\nu = 1.0$) to a convection-dominated one ($\nu = 10^{-6}$). We have the convective CFL equal to 1.0 in all cases. SUPG stabilization is again used.

Table 1 presents the iteration count for different diffusion values that lead to different scenarios. In the diffusion-dominated case the STBDDC preconditioned GMRES tends to an asymptotically constant number of iterations, thus is independent of the number of subdomains. Moving to the convective case, the number of iterations slightly grows with the decrease of the diffusive CFL number.

**5.5. Nonlinear space-time $p$-Laplacian solver.** In this experiment we compare the sequential-in-time method (solving the spatial problem with a BDDC approach for every time step) against the proposed STBDDC solver. We consider the $p$-Laplacian problem (with $p = 1$), i.e., (1) with $\nu = |\nabla u|$, $\beta(0,0)$, and $\sigma = 0$, on $\Omega = [0,1]^2 \times (0,T)$ with the initial solution $u^0 = x + y$, Dirichlet data $g = x + y$, and the forcing term $f = 1$. We fix the time step size to $|\delta_k| = 10^{-3}$ and $\frac{H}{h} = 30$.

We consider the same setting as the experiment reported in Figure 7 to compare the sequential-in-time and space-time parallel solvers for a nonlinear problem. Thus, we consider an increasing number of time steps, and $P_s = P_x \times P_y = 4^2$ processors for the space-time solver. The sequential solver exploits $P_s$ processors to extract space parallelism only. Figure 8 shows the comparison between the sequential approach up to $T = 1.5$ and $K = 1500$ time steps. In Figure 8(a) we plot the number of space solves, which is now proportional to the number of accumulated linear iterations through the nonlinear iterations. Remarkable algorithmic scalability is also obtained in the nonlinear setting. Elapsed time plots in 8(b) show a similar behavior in the nonlinear case to the linear one. The nonlinear case also exhibits excellent weak scalability properties. The nonlinear space-time solver is competitive in terms of number of local problems to be computed at about 20 processors, whereas it requires slightly more than 50 processors in order to be superior in terms of elapsed time. Discrepancies should be substantially reduced exploiting causality for the local problems, as commented upon above. Out of these plots, we are able to reduce the
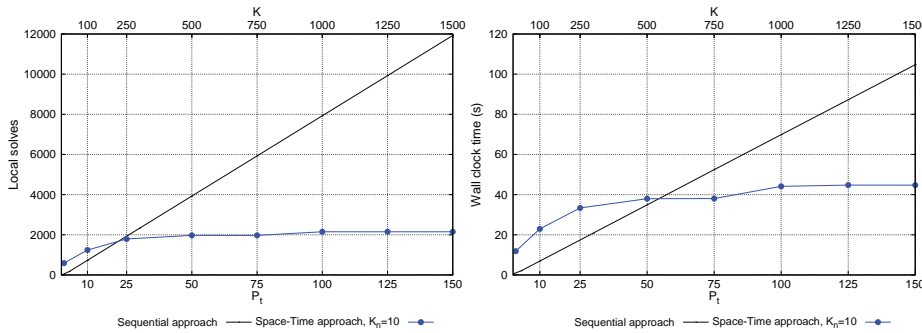
FIG. 8. *Comparison between the sequential and space-time solvers for the p-Laplacian transient problem on MareNostrum supercomputer. Spatial partition is fixed to $P_x = P_y = 4$. The space-time approach is using $P = 4^2 P_t$ MPI fine-level tasks.*

time to solution of simulations with the space-time approach by adding more processors to exploit time parallelism for the nonliner problem considered herein. In any case, there is still room for improvement when considering nonlinear problems. In this sense, nonlinear space-time preconditioning [15, 7] and more elaborated linearization strategies have the potential to lead to better performance.

**6. Conclusions.** In this work, we have considered a space-time iterative solver based on DD techniques. In particular, we have considered the GMRES iterative solver with space-time preconditioning obtained by extending the BDDC framework to space time for parabolic problems discretized with FEs. Since the time direction has a very different nature than the spatial one, i.e., it is a transport-type operator, a particular definition of the coarse DOFs and transfer operators is considered, taking into account time causality. Further, perturbation terms must be included to lead to a well-posed system. The exposition has been carried out for a backward-Euler time integrator, but the extension to $\theta$-methods and Runge-Kutta methods is straightforward. Further, the well-posedness of the proposed space-time preconditioner has been checked.

On the other hand, we have carried out a detailed set of numerical experiments on parallel platforms. Out of these results, the proposed methodology is observed to exhibit excellent scalability properties. The methods are weakly scalable in time, i.e., increasing X times the number of MPI tasks one can solve X times more time steps, in approximately the same amount of time, which is a key property to reduce time to solution in transient simulations with heavy time stepping. We have also shown weak scalability in space time, where one is not only facing larger problems in time but also in space. Further, we have applied the STBDDC preconditioner to nonlinear problems by considering a linearization of the full space-time system, and applying the proposed space-time solver at every nonlinear iteration. The use of the space-time solvers proposed becomes faster than a sequential-in-time approach for a modest number of processors.

Future work will include the development of nonlinear space-time preconditioners, extending the concept of nonlinear preconditioning (see, e.g., the recent work in [15] for nonlinear FE tearing and interconnecting (FETI) preconditioners in space) to space time. As it has already been observed in space [7], the use of nonlinear preconditioning should make space-time preconditioning more effective for nonlinear problems. Further extensions of this work will involve the extension to multilevel

space-time algorithms (to keep perfect weak scalability at larger scales), and their application to solid mechanics and incompressible fluid dynamics simulations.

## REFERENCES

[1] S. Badia, A. F. Martín, and J. Principe, *Implementation and scalability analysis of balancing domain decomposition methods*, Arch. Comput. Methods Eng., 20 (2013), pp. 239–262, http://dx.doi.org/10.1007/s11831-013-9086-4.

[2] S. Badia, A. F. Martín, and J. Principe, *A highly scalable parallel implementation of balancing domain decomposition by constraints*, SIAM J. Sci. Comput., 36 (2014), pp. C190–C218, http://dx.doi.org/10.1137/130931989.

[3] S. Badia, A. F. Martín, and J. Principe, *On the scalability of inexact balancing domain decomposition by constraints with overlapped coarse/fine corrections*, Parallel Comput., 50 (2015), pp. 1–24, http://dx.doi.org/10.1016/j.parco.2015.09.004.

[4] S. Badia, A. F. Martín, and J. Principe, *Multilevel balancing domain decomposition at extreme scales*, SIAM J. Sci. Comput., 38 (2016), pp. C22–C52, http://dx.doi.org/10.1137/15M1013511.

[5] S. C. Brenner and R. Scott, *The Mathematical Theory of Finite Element Methods*, 3rd ed., Springer, New York, 2010.

[6] A. N. Brooks and T. J. Hughes, *Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations*, Comput. Methods Appl. Mech. Engrg, 32 (1982), pp. 199–259, http://dx.doi.org/10.1016/0045-7825(82)90071-8.

[7] X.-C. Cai and D. E. Keyes, *Nonlinearly preconditioned inexact Newton algorithms*, SIAM J. Sci. Comput., 24 (2002), pp. 183–200, http://dx.doi.org/10.1137/S106482750037620X.

[8] E. C. Cyr, J. N. Shadid, and R. S. Tuminaro, *Stabilization and scalable block preconditioning for the Navier–Stokes equations*, J. Comput. Phys., 231 (2012), pp. 345–363, http://dx.doi.org/10.1016/j.jcp.2011.09.001.

[9] C. R. Dohrmann, *A preconditioner for substructuring based on constrained energy minimization*, SIAM J. Sci. Comput., 25 (2003), pp. 246–258, http://dx.doi.org/10.1137/S1064827502412887.

[10] M. Dryja, J. Galvis, and M. Sarkis, *BDDC methods for discontinuous Galerkin discretization of elliptic problems*, J. Complexity, 23 (2007), pp. 715–739, http://dx.doi.org/10.1016/j.jco.2007.02.003.

[11] M. Emmett and M. J. Minion, *Toward an efficient parallel in time method for partial differential equations*, Commun. App. Math. Comput. Sci., 7 (2012), pp. 105–132.

[12] R. D. Falgout, S. Friedhoff, TZ. V. Kolev, S. P. MacLachlan, and J. B. Schroder, *Parallel time integration with multigrid*, SIAM J. Sci. Comput., 36 (2014), pp. C635–C661, http://dx.doi.org/10.1137/130944230.

[13] M. J. Gander, R.-J. Li, and Y.-L. Jiang, *Parareal Schwarz waveform relaxation methods*, in Domain Decomposition Methods in Science and Engineering XX, Lecture Notes Comput. Sci. Eng. 91, Springer, Berlin, 2013, pp. 451–458.

[14] M. J. Gander and M. Neumüller, *Analysis of a new space-time parallel multigrid algorithm for parabolic problems*, SIAM J. Sci. Comput., 38 (2016), pp. A2173–A2208, http://dx.doi.org/10.1137/15M1046605.

[15] A. Klawonn, M. Lanser, and O. Rheinbach, *Nonlinear FETI-DP and BDDC Methods*, SIAM J. Sci. Comput., 36 (2014), pp. A737–A765, http://dx.doi.org/10.1137/130920563.

[16] J. Lions, Y. Maday, and A. Turinici, *A parareal in time discretization of PDEs*, C. R. Math. Acad. Sci. Paris, 332 (2001), pp. 661–668.

[17] J. Mandel and C. R. Dohrmann, *Convergence of a balancing domain decomposition by constraints and energy minimization*, Numer. Linear Algebra Appl., 10 (2003), pp. 639–659, http://dx.doi.org/10.1002/nla.341.

[18] J. Mandel, B. Sousedík, and C. Dohrmann, *Multispace and multilevel BDDC*, Computing, 83 (2008), pp. 55–85, http://dx.doi.org/10.1007/s00607-008-0014-7.

[19] M. L. Minion, R. Speck, M. Bolten, M. Emmett, and D. Ruprecht, *Interweaving PFASST and Parallel Multigrid*, SIAM J. Sci. Comput., 37 (2015), pp. S244–S263, http://dx.doi.org/10.1137/14097536X.

[20] R. Speck, D. Ruprecht, M. Emmett, M. Bolten, and R. Krause, *A space-time parallel solver for the three-dimensional heat equation*, in Parallel Computing: Accelerating Computational Science and Engineering (CSE), IOS press, Amsterdam, 2014, pp. 263–272. http://dx.doi.org/10.3233/978-1-61499-381-0-263.

[21] A. TOSELLI AND O. WIDLUND, *Domain Decomposition Methods - Algorithms and Theory*, Springer, Berlin, 2005.

[22] X. TU, *Three-level BDDC in three dimensions*, SIAM J. Sci. Comput., 29 (2007), pp. 1759–1780, http://dx.doi.org/10.1137/050629902.

[23] T. WEINZIERL AND T. KÖPPL, *A geometric space-time multigrid algorithm for the heat equation*, Numerical Mathematics Theory Methods Appl., 5 (2012), pp. 110–130, http://journals. cambridge.org/abstract_S1004897900000258.