SIAM J. SCI. COMPUT.
Vol. 43, No. 6, pp. A3713–A3732

© 2021 Society for Industrial and Applied Mathematics

STRUCTURE-PRESERVING NONLINEAR FILTERING FOR CONTINUOUS AND DISCONTINUOUS GALERKIN SPECTRAL/HP ELEMENT METHODS*

VIDHI ZALA†, ROBERT M. KIRBY†, AND AKIL NARAYAN‡

Abstract. Finite element simulations have been used to solve a variety of partial differential equations (PDEs) that model physical, chemical, and biological phenomena. The resulting discretized solutions to PDEs often do not satisfy requisite physical properties, such as positivity or monotonicity. Such invalid solutions pose both modeling challenges, since the physical interpretation of simulation results is not possible, and computational challenges, since such properties may be required to advance the scheme. We, therefore, consider the problem of computing solutions that preserve these structural solution properties, which we enforce as additional constraints on the solution. We consider in particular the class of convex constraints, which includes positivity and monotonicity. By embedding such constraints as a postprocessing convex optimization procedure, we are able to compute solutions that satisfy general types of convex constraints. For certain types of constraints (including positivity and monotonicity), the optimization is a filter, i.e., a norm-decreasing operation. We provide a variety of tests on one-dimensional time-dependent PDEs that demonstrate the efficacy of the method, and we empirically show that rates of convergence are unaffected by the inclusion of the constraints.

Key words. structure-preserving approximation, high-order accuracy, convex optimization

AMS subject classifications. 41A25, 41A36, 65D05, 65N30, 65M08, 65M60

DOI. 10.1137/20M1337223

1. Introduction.

Since the advent of numerical computing methods such as the finite element method (FEM) and the finite volume method (FVM) that solve partial differential equations (PDEs), the scientific computing community has advanced these methods with the goal of having computed solutions that emulate real-world phenomena. Many such numerical methods rely on piecewise polynomial approximations of fields. For example, we frequently numerically solve a system of PDEs to predict a state variable u that advects with wave-like motion. When computationally representing u using a piecewise polynomial, some physical properties of this variable can be lost sometimes. If u denotes the concentration of some quantity, piecewise polynomial representations often result in nonnegative values u, yet negative values of u are not physically interpretable. Other examples of properties that may be lost are minimum/maximum bound on values or monotonically increasing or decreasing

*Submitted to the journal's Methods and Algorithms for Scientific Computing section May 18, 2020; accepted for publication (in revised form) June 14, 2021; published electronically November 4, 2021.

https://doi.org/10.1137/20M1337223

Funding: The first and second authors acknowledge that their part of this research was sponsored by ARL under cooperative agreement number W911NF-12-2-0023. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of ARL or the US Government. The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. The third author was partially supported by NSF DMS-1848508. This material is based upon work supported by both the National Science Foundation under grant DMS-1439786 and the Simons Foundation Institute grant award ID 507536 while the third author was in residence at the Institute for Computational and Experimental Research in Mathematics in Providence, RI, during the Spring 2020 semester.

†Scientific Computing and Imaging Institute and School of Computing, University of Utah, Salt Lake City, UT 84112 USA (vidhi.zala@utah.edu, kirby@cs.utah.edu).

‡Scientific Computing and Imaging Institute and Department of Mathematics, University of Utah, Salt Lake City, UT 84112 USA (akil@sci.utah.edu).

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

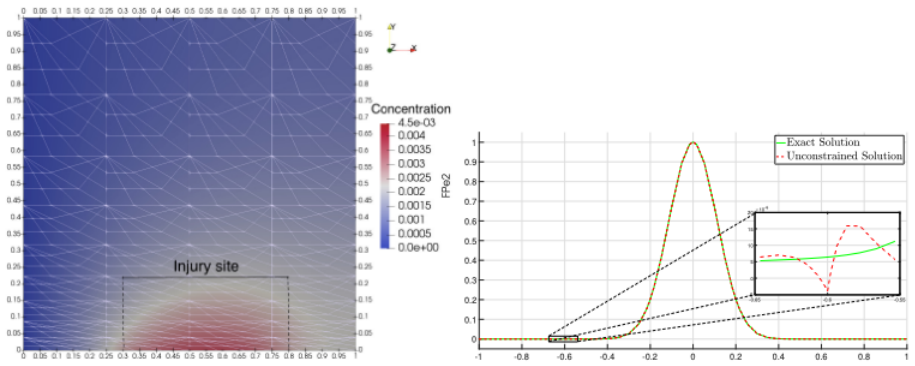

FIG. 1. *Left: Snapshot in time of an FEM method solution for the platelet aggregation and blood coagulation model* [5] *that shows the evolution of the fluid phase chemical Thrombin (FPe2) near the injury site of the vessel. The value of the concentration of FPe2 at a particular point in time is observed to be negative because of the concentration profile near the injury site, which, in turn, causes the simulation to fail. Right: A simple implementation of the sharp change in concentration in a combustion-like scenario using polynomial projection on a one-dimensional domain. Inset shows the concentration at point* $x = -0.6$ *is negative (not physically meaningful), which can lead to failure of the simulation.*

behavior. In this paper, we consider a general class of convex *structural* properties of the state, which includes the previous examples.

PDE models in which the importance of structure preservation can be observed include combustion problems, fluid flow problems, atmospheric predictions, and modeling of biochemical processes, such as platelet aggregation and blood coagulation.

In many cases, the theoretical and practical feasibility of numerical methods depends on how closely the computed approximation to u follows the requisite physical structure. If this underlying structure is violated, the resulting computation may produce unphysical predictions, and/or may cause solvability issues in numerical schemes that approximate PDE solutions. A violation in structure may arise from seemingly benign approximation properties; for example, polynomial approximations that yield Gibbs' oscillations often still converge in the mean-square sense, but these oscillations can cause a violation of positivity.

Figure 1 shows examples depicting modeling issues resulting from a nontrivial application. We see that the phenomenon of invalid solutions occurs frequently in many fields, which employ polynomial-based methods for simulations. These simulations are prone to instability or failure as a result of the feedback of an invalid solution from one step to another. It is therefore essential to carefully address the issue of invalid solutions in a generic, domain independent and robust manner without a loss of stability or convergence of the numerical solutions.

1.1. Contribution. In this paper, we present a general framework for preserving structure in piecewise polynomial-based time-dependent PDE solvers. The procedure, which is agnostic to timestepper by design, is applied at the end of each timestep. We propose a nonintrusive procedure to address the structure-preservation problem. We apply a postprocessing optimization procedure after time steps in the numerical PDE solver that enforces structure as a convex constraint. For certain constraints this optimization is norm-contractive so that our procedure can be interpreted as an application of a (nonlinear) *filter*.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Although the solution we propose corresponds to a conceptually simple convex feasibility problem, the convex feasible set is "complicated"; in particular, the feasible set is not a finite intersection of simple convex sets, such as Euclidean halfspaces. Therefore, many standard convex optimization algorithms cannot be used to directly solve this problem. However, recent work in [13] proposes algorithms to solve this optimization problem for the approximation problem. Our contribution in this paper is to investigate and extend this procedure applied to the numerical solution of PDEs. Our investigations require us to make additional algorithmic advances that are of interest to some PDE solutions: conservation of mass and preservation of boundary conditions and interelement fluxes. We investigate the efficacy of our filter to efficiently and accurately compute solutions to PDEs while preserving physical structure. All our investigations in this paper are limited to time-dependent PDEs in one spatial dimension. We explore a combination of implicit and explicit approaches to solving the PDEs.

The outline of this paper is as follows. In subsection 1.2, we survey existing solutions to the problem considered in this paper and briefly discuss their limitations. A brief, high-level description of the proposed optimization procedure/filter is given in section 2, but a more detailed description of the algorithm including its application to function approximation can be found in [13]. In section 2, we also summarize notation used throughout the paper and describe the types of constraints we consider. Section 3 details the application to PDEs and some additional discussion of the filtering procedure using geometrical interpretations. We also discuss how the filtering procedure changes some quantities of interest, such as values on element boundaries and total mass. We subsequently formulate a remedy to conserve these quantities by incorporating extra constraints into the filter. Section 4 summarizes the proposed algorithm as formulated in section 3. Finally, section 5 presents numerical examples with solutions to PDEs using both discontinuous Galerkin (dG) and continuous Galerkin (cG) formulations. We provide empirical evidence that this optimization procedure has negligible impact on convergence rates and absolute accuracy for these solvers.

1.2. Existing techniques for structure preservation. A number of strategies have been proposed in the literature to preserve structure of polynomial approximations. Many of these strategies can successfully guarantee the preservation of structure in special cases, or with special discretizations. For narrative purposes, we partition the existing methods into two broad categories: nonintrusive and intrusive.

Methods are considered to be nonintrusive if they constrain the solution obtained from the solver with minimal, often superficial, change to the numerical scheme. Prominent nonintrusive methods include limiters applied at each timestep [7, 16, 10, 8, 11]. These limiters typically affect only the design of numerical fluxes and not the underlying scheme. However, they must be specially designed for different kinds of constraints and approximation spaces. These limiters therefore lack flexibility with respect to discretization, spatial dimension, and type of structure/constraint.

We also note that many different types of structure-preserving constraints can be considered. For example, the technique from [16, 3] can successfully impose maximum principle-based constraints on a scalar or vector field. The mass-variable mass-target optimization-based remap approach described in [2] prescribes mass and maximum principle conservation successfully on a discrete set of points in the domain. However, some applications require that although two fields u and v need not individually obey a maximum principle, the sum $u+v$ must obey such a principle [3]. In such scenarios, standard maximum principle approaches cannot be employed.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

The solution we propose belongs to this nonintrusive category, but attempts to mitigate the previously mentioned flexibility issues. We consider a continuous version of the constraint satisfaction problem that ensures the constraints are satisfied on all the points in the domain, and not on just a discrete set of points. Furthermore, the proposed method can be used for any arbitrarily high polynomial orders without changes to the algorithm.

The second class of methods are those that are intrusive. An approach is considered to be intrusive if it needs to substantially change the underlying numerical scheme or properties of the solution domain. The intrusive methods look at the structure-preservation problem as a PDE-constrained optimization problem. Some of these methods modify the spatial discretization [15], and others use limiters derived from Karush–Kahan–Tucker (KKT) optimality conditions [12]. The strategies proposed for constraint satisfaction in [2] consider a version of the problem that is a subset of the one solved by the proposed solution. In [1] and its extensions [2, 9], the authors explore approaches for positivity preservation that are based on basis functions derived from Bernstein polynomials such that they are nonnegative and possess partition of unity property. Therefore the interpolated solution respects the original bounds at any point in the domain. While successful in imposing structure as part of the PDE discretization, intrusive approaches often suffer from the limitation of having to modify the numerical scheme, and the type of modification is both problem- and constraint-dependent. It requires substantial human intervention to change the discrete PDE solver. Methods involving changes in domain to solve this problem are also largely problem-specific and therefore lack flexibility. For example, the authors in [15] use an optimization problem incorporated in the scheme, and the solution is computed on a curved mesh that tracks discrepancies.

To summarize, existing strategies in the literature to preserve structure typically come in the form of intrusive methods, requiring nontrivial modification to numerical schemes, or nonintrusive methods, which typically affect existing numerical implementations in benign ways. The procedure we consider in this paper falls into the latter category, and our framework handles very general constraints. Furthermore, the mathematical formulation is agnostic to the type of (linear, convex) constraint and the spatial dimension of the problem. The price we pay for this generality is that some nontrivial (but convex) optimization must be performed. We describe this optimization in more detail in the next section.

2. Structure-preserving function approximation. In this section, we summarize the main algorithmic ideas from [13], which is a major ingredient for our approach. This approach is a map M from a given function u (that may or may not satisfy structural constraints) in a finite-dimensional space V to a unique function $M(u) \in V$, which does satisfy these constraints. The work in [13] defines a general class of constraints, corresponding to a feasible set in V that is an affine convex cone, and shows that $M(u)$ is the projection of u onto this feasible set, which is an affine convex cone in V. Unfortunately, this cone is not a polytope, and convenient parameterizations of V result in representation as an intersection of an (uncountably) infinite number of halfspaces. This formulation and parameterization does not easily lend itself to existing algorithms, so [13] develops some novel algorithms, based on seminal convex feasibility algorithms [14]. We now briefly discuss the types of constraints considered in [13] and some algorithms to implement them.

Let $\Omega \subset \mathbb{R}^d$ be a physical domain. In this paper, we are interested in $d = 1$, but this restriction is not necessary for the general approach. Let V be a

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

finite-dimensional Hilbert space of real-valued functions on Ω (for example, polynomials up to some fixed, finite degree), and suppose $u \in V$ is a given function. The approach in [13] considers families of constraints, each of the form

$$\mathcal{L}_x(u) \leq \ell(x), \qquad x \in \Omega, \tag{2.1}$$

where $\mathcal{L}_x$ is a linear operator that is bounded on V, and ℓ is a function on Ω. The feasible set in V adhering to such a constraint family is convex and includes the following examples:

- Positivity: $u(x) \geq 0$ for all x in Ω.
- Monotonicity: $u'(x) \geq 0$ for all x in Ω.

Note that the framework in [13] allows a finite number of such families to be considered simultaneously, so that boundedness, e.g., enforcing $0 \leq u(x) \leq 1$, is also a valid constraint. The families of constraints correspond to a feasible set $K \subset V$ corresponding to the elements of V that satisfy the constraints. The strategy in [13] is to solve the convex feasibility problem

$$M(u) \coloneqq \operatorname*{argmin}_{k \in K} \|u - k\|_V, \tag{2.2}$$

which is well posed. If $0 \in K$, then this optimization problem is norm-contractive [13, Proposition 5.1], and therefore can be interpreted as a filter.

We define more notations to describe the algorithm. Suppose V is an N-dimensional subspace of a Hilbert space H, with $\{\psi_j\}_{j=0}^{N-1}$ a collection of orthonormal basis functions,

$$V = \operatorname{span}\{\psi_0, \ldots, \psi_{N-1}\}, \qquad \langle \psi_i, \psi_j \rangle = \delta_{ij}, \qquad i, j = 0, \ldots, N-1,$$

where $\langle \cdot, \cdot \rangle$ is the inner product on V, and δ_{ij}, the Kronecker delta function. For a particular constraint $k \in K$, we can represent $u \in V$ in its coordinates $\{\widehat{v}_j\}_{j=0}^{N-1}$ collected in a vector $\boldsymbol{v} \in \mathbb{R}^N$. Any $u \in V$ that does not satisfy the desired constraints can be represented as $\sum_{j=0}^{N-1} \tilde{v}_j \psi_j = \tilde{\boldsymbol{v}} \boldsymbol{\psi}$. We collect the coefficients of expansion in a vector $\tilde{\boldsymbol{v}} \in \mathbb{R}^N$. The optimization problem (2.2) is therefore equivalent to

$$\operatorname*{argmin}_{\boldsymbol{v} \in C} \|\tilde{\boldsymbol{v}} - \boldsymbol{v}\|_2, \tag{2.3}$$

where $\boldsymbol{v}$ is the filtered version of $\tilde{\boldsymbol{v}}$ and obeys the constraints, $\|\cdot\|_2$ is the Euclidean 2-norm on vectors, and C is the affine conic region in $\mathbb{R}^N$ corresponding to $K \subset V$. Whereas the basis function $\boldsymbol{\psi}$ represents any orthonormal basis function, in general, use of any basis function is generally possible as long as a transformation to the orthonormal basis is done prior to the application of the filter. Note that the filter operates on the coefficients of expansion $\tilde{\boldsymbol{v}}$ while ensuring that the filtered $\boldsymbol{v}$ can be mapped back to the baseline set of basis function, thus maintaining the constraint satisfaction on all the points in the domain. If K contains a single family of the form (2.1), then the set C can be written as

$$C = \bigcap_{x \in \Omega} H_x,$$

where H_x are halfspaces in $\mathbb{R}^N$. In other words, for a fixed x, H_x is an $(N-1)$-dimensional planar surface in $\mathbb{R}^N$ defined by the single linear constraint (2.1). The algorithms in [13] proceed by computationally inspecting the signed distance function,

$$s(x) \coloneqq \operatorname{sdist}(\tilde{\boldsymbol{v}}, C_x) = \begin{cases} -\operatorname{dist}(\tilde{\boldsymbol{v}}, H_x), & x \notin C_x, \\ +\operatorname{dist}(\tilde{\boldsymbol{v}}, H_x), & x \in C_x. \end{cases}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

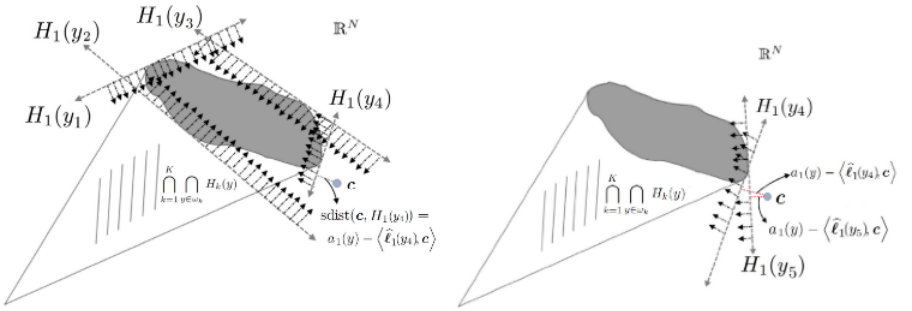

FIG. 2. *The steps in the procedure that find the distance between $\tilde{\boldsymbol{v}}$ and the hyperspace boundary representing each violating constraint. Subsequently, the algorithm greedily calculates the correction to $\tilde{\boldsymbol{v}}$. Left: A geometrical visual of the distance calculation from $\tilde{\boldsymbol{v}}$ to the hyperplanes defining boundaries of the violating constraint. Right: Selection of H_{y4} over H_{y5} since it defines the hyperplane farther away compared to other violating constraints. Projection of $\tilde{\boldsymbol{v}}$ on to H_{y4}.*

Here, "inspection" means, for example, the ability to compute the global minimum of $s(x)$ and/or to determine regions where s is negative. Based on this inspection, the algorithms project the state vector of the current iterate $\tilde{\boldsymbol{v}}$ onto H_y for some $y \in \Omega$, or perform relaxed/averaged versions of these projections. Geometrically, the projection operation corresponds to projecting $\tilde{\boldsymbol{v}}$ onto a supporting hyperplane H_x for C, which we depict in Figure 2.

The process is repeated until numerical convergence up to a tolerance (i.e., until the minimum value of the signed distance function is numerically 0). Thus, this algorithm is a type of generalized iterative cyclic/alternating algorithm applied to the case of an infinite number of convex sets (halfspaces).

The major computational work in this iterative procedure is the manipulation/minimization of $s(x)$ at each iteration. This signed distance function has the form

$$s(x) = \lambda(x)\left(\mathcal{L}_x(u) - \ell(x)\right), \qquad \lambda^2(x) \coloneqq \frac{1}{\sum_{j=0}^{N-1} \left(\mathcal{L}_x(\psi_j)\right)^2}, \tag{2.4}$$

where, $\ell \in \Omega$, and s is a λ-weighted version of u; therefore it is easy to evaluate. However computing a global minimum for s may be difficult in general. In [13], it is shown that if V is a univariate polynomial space of degree $N-1$, then computing the minimum of s can be accomplished by computing the roots of a polynomial of degree $3N$. We accomplish this by computing the spectrum of a confederate matrix associated with a Legendre orthogonal polynomial basis. For more details on the algorithm, including computational cost, see [13].

This algorithm which seeks to solve the optimization problem (2.2) is the key ingredient in our approach to preserve structure in time-dependent PDE simulations.

3. Method. We now discuss the application of the filter introduced in section 2 to time-dependent PDEs in one spatial dimension. We will primarily focus on method-of-lines discretizations with a Galerkin-type spatial discretization. In particular, we will consider cG and dG formulations. Consider an advection-diffusion-reaction system defined as

$$u_t(x,t) + a \cdot u_x(x,t) = \gamma u_{xx}(x,t) + r(u(x,t)), \tag{3.1}$$

where $x \in \Omega$, a is the velocity for advection, γ is the diffusivity, and r is a nonlinear function representing the reaction term. For simplicity, assume a and γ to

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

be constant advection and diffusion coefficients, respectively. Assuming appropriate initial $\boldsymbol{u}_0(x, t = 0)$ and boundary conditions are defined for (3.1), we can formulate its semidiscrete form as follows.

Galerkin-type methods assume an ansatz for u as a time-varying element of a fixed N-dimensional linear subspace V, where frequently $V \subset L^2(\Omega)$:

$$u(x,t) \approx u_N(x,t) \coloneqq \sum_{i=0}^{N-1} \tilde{v}_i(t)\phi_i(x), \qquad V = \operatorname{span}\{\phi_1, \ldots, \phi_N\}. \tag{3.2}$$

Note that the basis functions ϕ_i used in (3.2) represent the traditional FEM basis functions that span the entire Ω and are not necessarily orthogonal. We denote the orthonormal basis functions by $\boldsymbol{\psi}_j$. We assume that both basis sets span the same space and thus a transformation between them exists. For example, finite element methods partition Ω into E nonoverlapping subintervals. As a next step, the method assumes that the continuous functions in V are polynomials of a fixed degree N on each $E \in \Omega$. The discontinuities in derivatives are allowed only at partition boundaries. Similarly, dG FEMs define V in a similar way except that elements are allowed to be discontinuous at partition boundaries. The semidiscrete form for (3.1) is derived in the standard Galerkin way, by using the ansatz (3.2) and forcing the residual to be L^2-orthogonal to V. Usually, integration by parts is performed in the residual orthogonalization step and often, depending on the equation and spatial discretization, a numerical flux and/or stabilization terms are included in the resulting weak formulation.

The result is a system of ordinary differential equations prescribing time-evolution of the discrete degrees of freedom represented by vector $\tilde{\boldsymbol{v}} = \{\tilde{v}_0, \ldots, \tilde{v}_{N-1}\}$:

$$\boldsymbol{M}\frac{\partial}{\partial t}\tilde{\boldsymbol{v}} + \boldsymbol{A}\tilde{\boldsymbol{v}} = -\gamma\boldsymbol{L}\tilde{\boldsymbol{v}} + \boldsymbol{r} + \boldsymbol{F}(\tilde{\boldsymbol{v}}), \tag{3.3}$$

where $\boldsymbol{M}$, $\boldsymbol{A}$, and $\boldsymbol{L}$ are the $N \times N$ mass, advection, and Laplacian (stiffness) matrices, respectively, defined as

$$(M)_{i,j} = \langle \phi_i, \phi_j \rangle, \qquad (A)_{i,j} = \left\langle \phi_i, a\frac{\partial}{\partial x}\phi_j(x) \right\rangle, \qquad (L)_{i,j} = \left\langle \frac{\partial}{\partial x}\phi_i, \frac{\partial}{\partial x}\phi_j \right\rangle.$$

The N-vector $\boldsymbol{r}$, defined as

$$\boldsymbol{r} = \sum_{j=0}^{N-1} \widehat{r}_i(t)\phi_i(x), V = \operatorname{span}\{\phi_1, \ldots, \phi_N\},$$

has entries that are numerical approximations to the integral of the nonlinear reaction term,

$$\widehat{r}_i(\tilde{\boldsymbol{v}}, t) \approx \int \boldsymbol{r}(u_N, t)\phi_i(x)\mathrm{d}x,$$

which in this paper we compute with a collocation-based approach. Finally, the term $\boldsymbol{F}(\tilde{\boldsymbol{v}})$ is a generic term for any numerical fluxes or stabilization terms. For example, in an advection-dominated problem with a dG formulation, $\boldsymbol{F}$ might be the upwind flux corresponding to the continuous advection term $a\boldsymbol{u}_x$. Finally, a fully discrete scheme is derived from (3.3) using an appropriate time-integration method.

Let $\boldsymbol{v}^n$ represent the solution of (3.3) at time step n, and let us call $\tilde{\boldsymbol{v}}^{n+1}$ the solution at the time step $n+1$. For simplicity, assume that the solutions $\boldsymbol{v}^n$ and $\tilde{\boldsymbol{v}}^{n+1}$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

are transformed into orthonormal basis $\boldsymbol{\psi_j}$ and back to original basis ϕ_I inside the filter as the first and last steps. This discrete scheme does not enforce the structural properties that we desire. As discussed in section 2, given a constraint set $C \subset \mathbb{R}^N$, if $\boldsymbol{v}^n \in C$, then it is not necessarily true that $\tilde{\boldsymbol{v}}^{n+1} \in C$. To rectify this situation, we employ the optimization outlined in section 2 as a postprocessing step. We will hereafter refer to this optimization as a nonlinear "filter" due its norm-contractivity properties for the types of constraints we consider [13, Proposition 5.1]. Thus, our proposed procedure is a simple, nonintrusive augmentation of the standard fully discrete scheme (3.3):

$$\boldsymbol{v}^n \xrightarrow{\text{Timestepper for (3.3)}} \tilde{\boldsymbol{v}}^{n+1} \xrightarrow{\text{Filter}} \boldsymbol{v}^{n+1}. \tag{3.4}$$

Thus, we obtain the filtered solution $\boldsymbol{v}^{n+1} = \sum_{j=0}^{N-1} \widehat{v}_j(t)\phi_j(x)$ and introduce it as the state in the next step of the fully discrete scheme. The effect of the filter for an example of constraint: positivity preservation for $\tilde{\boldsymbol{v}}^n$ is demonstrated in Figure 3, which shows a snapshot in time of an advecting wave. The property of interest here is the positivity, and in this particular example, a dG formulation is used as the PDE solver. The process of enforcing positivity (see inset) leads to changes in solution properties; in particular, elementwise boundary values and the mean of the discrete solution are not preserved. A change in element boundary values also changes the corresponding (say, upwind) fluxes at the next time step. Hence, in this particular case, the filter changes physical behavior (numerical fluxes and mean value).

We will assume hereafter that numerical fluxes are computed as explicit functions of left- and right-boundary values. Hence enforcing element boundary value conservation for a dG solver becomes necessary to ensure that numerical fluxes at future times are unchanged by this procedure. One remedy for this particular issue is to impose additional equality constraints (i.e., function values at element boundaries) in the filter. The preservation of the mean value (e.g., total mass) can also be enforced in a similar way. Later in this section we elaborate on the construction of the filter for these additional constraints. Note that, between cG and dG discretizations, element boundary value constraints are relevant (for the purposes of conserving numerical fluxes) only for dG-type discretizations.

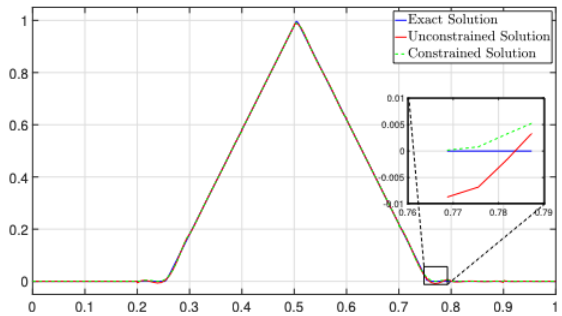

FIG. 3. *Numerical solution to a PDE with and without application of the positivity-constraining filter. Shown are the exact solution, the unconstrained solution (i.e., the solution via the scheme* (3.3)*), and the constrained solution (the solution via* (3.4)*).*

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

3.1. Degrees of freedom in cG and dG simulations. In order to describe our enforcement of equality constraints, we need to introduce some new notation that is specific to the physical discretization. Since dG discretizations have degrees of freedom that are decoupled across elements, the N degrees of freedom in the optimization described in section 2 only need to be over element-local degrees of freedom. In constrast, standard degrees of freedom in cG simulations (i.e., coefficients of "hat" and "bubble" functions) are coupled in L^2 across elements.

3.1.1. dG. With a partition consisting of E subintervals of Ω, a dG discretization allows us to convert the optimization (2.3) in an N-dimensional space into a much more efficient and parallelizable set of E independent optimizations each in $n := N/E \ll N$ dimensions. First, we recognize that a dG expansion of the form (3.2) can be written as

$$u_N(x,t) = \sum_{e=1}^{E} \sum_{k=0}^{n-1} \tilde{v}_{e,k} \psi_{e,k}(x), \qquad\qquad N = E\,n,$$

where $\{\psi_{e,k}(\cdot)\}_{k=0}^{n-1}$ for a fixed e are polynomials whose support is *only* on element e. In particular, these element-local polynomials can be chosen as $L^2(\Omega)$-orthonormal (e.g., mapped Legendre) polynomials. Therefore,

$$\langle \psi_{e,k}, \psi_{f,\ell} \rangle = 0, \qquad\qquad e \neq f, \quad k,\ell \in \{0,\dots,n-1\}. \tag{3.5a}$$

The second observation we make is that the linear constraint operator $\mathcal{L}_x$ defined in (2.1), for common examples we consider, also obeys this type of decoupling. If $\{\Omega_e\}_{e=1}^{E}$ is the partition of Ω, then we assume that

$$\mathcal{L}_x(\psi_{e,k}(x)) = 0, \qquad\qquad x \notin \Omega_e. \tag{3.5b}$$

This assumption is satisfied for point and derivative evaluation, i.e., for $\mathcal{L}_x(u) = u(x)$ and $\mathcal{L}_x(u) = u'(x)$. Under conditions (3.5), both the constraint and the L^2 norm are decoupled across elements, and so we have the following result.

PROPOSITION 3.1. *Assume that conditions* (3.5) *hold. Given any* $\tilde{\boldsymbol{v}} \in \mathbb{R}^N$*, partition it into* E *n-dimensional subvectors, each containing element-local degrees of freedom:*

$$\tilde{\boldsymbol{v}} = \begin{pmatrix} \tilde{\boldsymbol{v}}_1 \\ \tilde{\boldsymbol{v}}_2 \\ \vdots \\ \tilde{\boldsymbol{v}}_E \end{pmatrix}, \qquad\qquad \tilde{\boldsymbol{v}}_e = \begin{pmatrix} \tilde{v}_{e,0} \\ \vdots \\ \tilde{v}_{e,n-1} \end{pmatrix}. \tag{3.6}$$

Then the solution to (2.3) *is given by*

$$\operatorname*{argmin}_{\boldsymbol{v}\in C} \|\boldsymbol{v} - \tilde{\boldsymbol{v}}\|_2 = \begin{pmatrix} \operatorname{argmin}_{\boldsymbol{w}\in C^1} \|\boldsymbol{w} - \tilde{\boldsymbol{v}}_1\|_2 \\ \operatorname{argmin}_{\boldsymbol{w}\in C^2} \|\boldsymbol{w} - \tilde{\boldsymbol{v}}_2\|_2 \\ \vdots \\ \operatorname{argmin}_{\boldsymbol{w}\in C^E} \|\boldsymbol{w} - \tilde{\boldsymbol{v}}_E\|_2 \end{pmatrix},$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

where the n-dimensional sets C^e are defined as

$$C^e \coloneqq \left\{ \boldsymbol{w} = (w_0, \ldots, w_{n-1})^T \in \mathbb{R}^n \;\middle|\; \sum_{k=0}^{n-1} w_k \psi_{e,k}(x) \le r(x) \;\; \textit{for all } x \in \Omega_e \right\}.$$

We emphasize again that dG discretizations satisfy (3.5) so that Propostion 3.1 allows us to conclude that our optimization (for example, constraining positivity and/or monotonicity) can be decoupled to operate on individual elements. The decoupling can result in substantial computational savings since frequently one N-dimensional optimization is much more expensive than E N/E-dimensional optimizations.

Remark 3.2. The elementwise decoupling conclusion of Proposition 3.1 holds under slightly more general conditions. For example, in so-called p-adaptive simulations, the number of local degrees of freedom may depend on the element index, i.e., $n = n(e)$. However, we do not pursue p-adaptive simulations in this paper, so this level of generality is not needed.

3.1.2. cG. In contrast to the dG framework, in cG discretizations, typically both the constraint operator (e.g., point evaluation operator) and the L^2 norm are coupled across elements. Thus, there is no straightforward decoupling procedure that can be leveraged. Instead, the full optimization problem (2.3) in N-dimensional space must be solved.

An additional difficulty with this optimization in cG simulations is that the discussion and algorithms presented in section 2 essentially require an expansion in *orthonormal* basis functions; i.e., (3.2) must be expressed as

$$u_N(x,t) = \sum_{j=0}^{N-1} \tilde{v}_j(t)\phi_j(x) = \sum_{j=0}^{N-1} \tilde{w}_j(t)\psi_j(x), \qquad \langle \psi_j, \psi_k \rangle = \delta_{k,j}. \tag{3.7}$$

Typically, cG simulations utilize nonorthonormal hat and bubble functions as degrees of freedom [4], and therefore the use of this optimizer requires transformation of the hat-bubble coordinates into an orthonormal set of coordinates. To accomplish this, (a Cholesky factor of) the mass matrix $\boldsymbol{M}$ must be inverted. Typically $\boldsymbol{M}$ is sparse, but its inverse Cholesky factor is usually not, which poses a challenge for large-N simulations when size-N dense matrix linear algebra is computationally infeasible.

In this paper, we consider simulations only in one spatial dimension where N is small enough so that direct inversion of $\boldsymbol{M}$ is feasible. However, more sophisticated procedures would be needed to extend this approach to two or three spatial dimensions.

3.2. Elementwise boundary value conservation (dG). The goal in this section is to enforce the element boundary constraints explained at the end of section 3 for dG discretizations. For simplicity of exposition, we assume in what follows that $\{\psi_{e,k}\}_{k=0}^{n-1}$ are L^2-orthonormal. The procedure we describe can be extended to the case when this assumption is not satisfied. With $\{\Omega_e\}_{e=1}^{E}$ the subinterval partition of Ω, we seek to solve the equality-constrained optimization problem,

$$\min_{\boldsymbol{v} \in C} \|\boldsymbol{v} - \tilde{\boldsymbol{v}}\|_2 \;\; \text{such that} \;\; \sum_{k=0}^{n-1} \tilde{v}_{e,k}\psi_{e,k}(x_e^{\pm}) = \sum_{k=0}^{n-1} v_{e,k}\psi_{e,k}(x_e^{\pm}) \;\; \textit{for all } e \in \{1, \ldots, E\}, \tag{3.8}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

where $\tilde{v}_{e,k}$ and $v_{e,k}$ are components of the elementwise subvector partition of $\tilde{\boldsymbol{v}}$ and $\boldsymbol{v}$, respectively, that was introduced in (3.6). The point values $x_e^{\pm}$ are left- and right-hand side values of subinterval e, $\Omega_e = [x_e^-, x_e^+]$. As described in subsection 3.1.1 and Proposition 3.1, the (nonequality-constrained) dG optimization problem can be decoupled into elementwise operations. Adding in the elementwise constraints described above does not change this decoupling property (since the boundary values on element e are independent of the degrees of freedom on any other element). The resulting optimization on element e has the form,

$$\min_{\boldsymbol{w}\in C^e} \|\boldsymbol{w} - \tilde{\boldsymbol{v}}_e\|_2 \;\text{ such that }\; \boldsymbol{Q}^T(\boldsymbol{w} - \tilde{\boldsymbol{v}}_e) = \boldsymbol{0}, \tag{3.9}$$

where $\boldsymbol{Q}$ is an $n \times 2$ matrix with orthonormal columns $\boldsymbol{q}_1, \boldsymbol{q}_2$ satisfying

$$\operatorname{span}\{\boldsymbol{q}_1, \boldsymbol{q}_2\} = \operatorname{span}\left\{\boldsymbol{\psi}_e(x_e^-), \boldsymbol{\psi}_e(x_e^+)\right\}, \quad \boldsymbol{\psi}_e(x) \coloneqq (\psi_{e,0}(x),\ \dots\ \psi_{e,n-1}(x))^T.$$

In order to solve this n-dimensional linear-equality-constrained optimization problem, we reduce it to an $(n-2)$-dimensional optimization problem of the standard form (2.3) by working in the $n-2$ coordinates corresponding to $\mathcal{R}(\boldsymbol{Q})^\perp$. To set up for this procedure, let $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \dots, \boldsymbol{p}_{n-2}\}$ be an(y) orthonormal completion of $\{\boldsymbol{q}_1, \boldsymbol{q}_2\}$ in $\mathbb{R}^n$, and introduce $\boldsymbol{P} \in \mathbb{R}^{n\times(n-2)}$,

$$\boldsymbol{P} = \begin{pmatrix} \boldsymbol{p}_1 & \boldsymbol{p}_2 & \cdots & \boldsymbol{p}_{n-2} \end{pmatrix}.$$

Then, our equality constrained optimization problem (3.9) is equivalent to the non-equality-constrained problem

$$\min_{\boldsymbol{z}\in\widehat{C}} \left\|\boldsymbol{z} - \boldsymbol{P}^T\tilde{\boldsymbol{v}}_e\right\|_2, \tag{3.10}$$

where $\widehat{C}$ is the $\mathbb{R}^{n-2}$-dimensional set,

$$\widehat{C} \coloneqq \left\{ \boldsymbol{z} \in \mathbb{R}^{n-2} \;\middle|\; \sum_{j=0}^{n-3} z_j \widehat{\psi}_j(x) \le \widehat{\ell}(x) \right\}, \qquad \widehat{\ell}(x) \coloneqq \ell(x) - \boldsymbol{\psi}_{\boldsymbol{e}}(x)^T \boldsymbol{Q}\boldsymbol{Q}^T\tilde{\boldsymbol{v}}_e,$$

and $\{\widehat{\psi}_j\}_{j=0}^{n-3}$ are the $\boldsymbol{P}$-projected functions,

$$\boldsymbol{\psi}_e^T(x)\boldsymbol{P} =: \begin{pmatrix} \widehat{\psi}_0(x) & \cdots & \widehat{\psi}_{n-3}(x) \end{pmatrix}.$$

Therefore, the reduced problem (3.10) is precisely a special case of our problem (2.3), and all the same tools described in section 2 are applicable. The signed distance function in (2.4) requires knowledge only of the appropriate $n-2$ projected orthonormal functions. The formula precisely is

$$s(x) = \lambda(x)\left(\mathcal{L}_x(z) - \widehat{\ell}(x)\right), \qquad \lambda^2(x) = \frac{1}{\sum_{j=0}^{n-3}(\mathcal{L}_x(\widehat{\psi}_j(x))^2},$$

and $z = \sum_{j=0}^{n-3} z_j\widehat{\psi}_j(x)$. Once this solution, say $\boldsymbol{z}^*$, is computed, then the solution to (3.8) is the n-dimensional vector $\boldsymbol{Q}\boldsymbol{Q}^T\tilde{\boldsymbol{v}}_e + \boldsymbol{P}\boldsymbol{z}^*$.

To summarize, we wish to solve the N-dimensional problem (3.8), which simultaneously enforces inequality constraints (such as positivity) and preserves element

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

boundary values (which we take as a proxy for preservation of numerical fluxes). This problem is equivalent to solving (3.9) for every element index e. For a fixed e, this latter problem, in turn, is equivalent to the $(n-2)$-dimensional problem (3.10), which can be solved with the techniques outlined in section 2. Note in particular that all the matrices involved in this section can be precomputed and stored for use in online time-dependent simulations. Furthermore, if all physical elements are templated on a standard (reference) element, as is common in finite element code, then only one copy of all these size-n matrices is required for the entire simulation.

3.3. Mass conservation (cG and dG). A similar approach for mass (integral) conservation can be obtained for both cG and dG discretizations using a procedure that is essentially identical to the one described in the previous section. For example, in a cG discretization with orthonormal coordinates $\tilde{\boldsymbol{w}}$, we might wish to perform the optimization (2.3) for $\boldsymbol{w}$ subject to the equality constraint,

$$\int_\Omega \sum_{i=0}^{N-1} w_i \psi_i(x) dx = M \coloneqq \int_\Omega \sum_{i=0}^{N-1} \tilde{w}_i \psi_i(x) \mathrm{d}x,$$

where we recall that $\{\psi_i\}_{j=0}^{N-1}$ are the orthonormal basis functions in (3.7). This is equivalent to the equality constraint

$$\boldsymbol{q}^T \boldsymbol{w} = M, \qquad\qquad q_j \coloneqq \int_\Omega \psi_j(x) \mathrm{d}x,$$

and therefore the procedure of subsection 3.2 can be leveraged (with only one equality constraint and with an N-dimensional state vector). This process leads to the global conservation of mass. In many cases, having local mass conservation and constraint satisfaction across individual elements is desirable. This type of problem is possible to formulate by careful reconstruction of constraints without any changes to the algorithm. The constraint on the entire domain can be broken down per element, resulting in "E" constraints, where E = number of elements $\{e_0, e_1, \ldots, e_{E-1}\} \in \Omega$. Therefore, the constraint set looks like the one described in (3.8). The only difference in the problem formulation will be the definition of the constraint set, which will be a matrix of size $N \times E$, with each column representing a mass conservation condition for each element, respectively. Since we use a linear programming solver in the algorithm, with the increase in number of constraints, the complexity of the solver increases proportionally. The complexity can be estimated by following the constraint set formulation and its properties in section 3.1 of [13].

For dG discretizations, the same idea can be applied, except that straightforward mass conservation couples all elements. To retain the elementwise decoupling efficiency, we impose a stronger condition that the mass *per element* remain unchanged. This allows us to use the same procedure as in subsection 3.2, performing E size-n optimizations. In particular, we can simultaneously enforce both element boundary preservation and elementwise mass conservation with three equality constraints.

Note that these procedures are generalizable for arbitrary linear constraints. In particular, if we have K linear constraints, then the dimension of the optimization problem can be reduced to an $(n-K)$-dimensional problem using the procedure described in section 3.2. This, in turn, means that this optimization is meaningful only if the original size of the problem is more than K. For example, for dG simulations we can only accommodate $n-1$ linear constraints per element if there are n degrees of freedom per element.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

4. Algorithm. This section describes some algorithmic considerations of our approach. We emphasize that although in previous sections we have described details in different ways for cG versus dG discretizations, the fundamental tool, i.e., the algorithms in section 2, are identical. The differences manifest only when we need to fit these types of discretrizations into the general framework of that section. This section will also highlight some algorithmic differences between the cG and dG discretizations that surface in the use of our optimization filter.

Note that, regardless of the discretization used, this filter preserves L^2 stability properties since it is norm-contractive. For example, if a CFL condition for L^2 stabilty of an unconstrained solver is used to determine a stable time-step value, then a filtered version of this solver will not require a change in time-step value to maintain this stability property.

4.1. Algorithmic ingredients. As a first step of the optimization procedure, we must transform any given basis $\{\phi\}_{j=0}^{N-1}$ that is assumed by the PDE solver to an orthonormal basis $\{\psi\}_{j=0}^{N-1}$. We use the standard procedure that requires application of the inverse of Cholesky decomposition of the mass matrix. In dG simulations, we can leverage elementwise decoupling as described in section 3.1.1 so that the appropriate matrix algebra is performed locally on a local mass matrix associated with only a single element. However, when a cG formulation is used, the global mass matrix must be inverted, and this cost significantly slows down the optimization process.

In cG, the structure of the global mass matrix $\boldsymbol{M}$ when using standard hat and bubble basis functions is "nearly" diagonal, but the inverse of $\boldsymbol{M}$ (or its Cholesky factor) is a dense matrix. In our simulations, we precompute the required global matrices for orthonormalization of the basis elements and coordinates. This procedure is feasible for our problems in one spatial dimension. However, this process is no longer feasible if the global number of degrees of freedom N becomes too large, as would happen with problems in two or three spatial dimensions.

4.1.1. dG specializations. For dG discretizations, the global filtering operation is decoupled to act on individual elements. This element-by-element application lends itself to parallel implementation. In addition, we need not call the filter on elements where the constraints are already satisfied. For example, if positivity is the constraint, then at every time step we can flag elements where positivity may be violated and run the optimizer only over those elements. In our implementation, we use confederate linearization to check for the optima of function values on each element. This approach guarantees that all the optima are calculated accurately. We use the optima to determine whether there is a violation of positivity constraint on that element. Although a computational cost is associated with the eigenvalue solve on each element, the overall speed-up obtained by not having to calculate the signed distance function described in (4.1) on each element certainly justifies the extra computation.

With this flagging scheme, we observe that the computational time spent on the filter is a fraction of the cost of the unconstrained solver (cf. our numerical results in section 5).

4.2. Algorithm summary. For completeness, we now summarize one procedure proposed in [13] to perform the optimization in section 3. Consider a stable implementation of (3.1). As discussed in section 3, the optimization that enforces a constrained solution is performed every time step.

Recalling some notation from section 2, we assume K families of linear constraints, and for a fixed $k \in \{1, \ldots, K\}$ each constraint is of the form (2.1). This results in

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

K feasible sets $\{C_k\}_{k=1}^K$, and K signed distance functions $\{s_k\}_{k=1}^k$ each of the form (2.4). The full feasible set C is the intersection of the C_k.

Computation of this signed distance function requires orthonormalization of a given basis for V, which in turn requires the (inverse) Cholesky factor of the mass matrix, which can be precomputed before the simulation begins. A conversion from an input coordinate vector to coordinates $\tilde{\boldsymbol{v}}$ in an orthonormal basis is the first stage of the filtering procedure.

The second stage of the filter is an iterative procedure that attempts to project $\tilde{\boldsymbol{v}}$ onto the feasible set C_k. At each iteration, we find the constraint index k and the spatial point x that minimizes the signed distance function,

$$(x^*, k^*) \coloneqq \operatorname*{argmin}_{x\in\Omega, k\in\{1,\dots,K\}} s_k(x). \tag{4.1}$$

We then update $\tilde{\boldsymbol{v}}$ by projecting it onto the hyperplace corresponding to (x^*, k^*):

$$\tilde{\boldsymbol{v}} \leftarrow \tilde{\boldsymbol{v}} + \boldsymbol{h}(x^*, k^*) \min\left\{0, s_{k^*}(x^*)\right\}, \tag{4.2}$$

where $\boldsymbol{h}(x, k)$ is the normal vector corresponding to hyperplane H_x of constraint family k that points toward C_k. This vector is readily computable from the orthonormal basis; see [13] for details. This procedure is repeated until $s_{k^*}(x^*)$ vanishes to within a numerical tolerance. Upon termination of the iterations, the output of the filtering procedure is the (updated) $\tilde{\boldsymbol{v}}$.

A brief summary of steps taken by a filtered PDE solver is presented in Algorithm 4.1.

Algorithm 4.1. Constrained PDE timestepping

Input: Terminal time T, time step size Δt, PDE solver spatial basis $\boldsymbol{\phi}$
Orthonormalize the basis function $\boldsymbol{\phi}$ to $\boldsymbol{\psi} \in \mathbb{R}^N$
Define nsteps $= \frac{T}{\Delta t}$
for $i = 0, \dots,$ nsteps **do**
  Solve PDE to obtain the coefficients $\tilde{\boldsymbol{v}}^i \in \mathbb{R}^N$
  Input: constraints $(L_k, \ell_k)_{k=1}^K$
  while True **do**
    Compute (x^*, k^*) via (4.1).
    If $s_{k^*}(x^*) \geq 0$, **break**
    Update $\tilde{\boldsymbol{v}}^{i+1}(t)$ via (4.2).
  end while
  $\boldsymbol{v}^{i+1} = \tilde{\boldsymbol{v}}^{i+1}$
end for

5. Numerical results. We numerically investigate the proposed procedure for preserving convex constraints in solutions to PDEs. We consider a bounded one-dimensional spatial interval $\Omega \subset \mathbb{R}$. We will consider positivity constraints (i.e., $u(x) \geq 0$ over all Ω), and will also investigate cell boundary value ("flux") preservation and mass conservation (for dG simulations). We are primarily interested in the effect that the filter has on convergence rates and in quantifying the computational efficiency of the procedure. All simulations perform the procedure summarized in Algorithm 4.1. We use the following machine to report all the performance and error numbers in this section: 256 Intel(R) Xeon(R) CPU E7-4850 v4 @ 2.10 GHz cores (HT) with 1024 GB of RAM running Redhat Enterprise 7.5 (Maipo).

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

5.1. dG. Consider the one-dimensional advection equation,

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0,$$

where a is a fixed constant. For $a = 1$, we investigate this problem for the exact solution $u(x,t) = 0.5\sin(2\pi x - 2\pi t - 0.5\pi) + 0.5$. We use a dG formulation with periodic boundary conditions over the domain $\Omega = [-1,1]$, which results in a system of ordinary differential equations prescribing the evolution of the Galerkin coefficient. We employ Runge-Kutta-2 to integrate in time and compute up to a final time $T = 1$ using upwind flux calculation.

Once fully discretized, the numerical solution can correspond to a function that is negative in some parts of the spatial domain, and hence we apply the filtering procedure in Algorithm 4.1 to enforce positivity on all Ω. This results in two numerical solutions:

- $\tilde{\boldsymbol{v}}$ is the solution resulting from a *standard* dG solver, i.e., one that does *not* employ our filter.
- $\boldsymbol{v}$ is the numerical solution resulting from application of the filter as specified in Algorithm 4.1.

As discussed earlier, this filtered solution does not respect mass conservation, and it in general changes values on the boundary, which changes numerical fluxes. Therefore, we use Algorithm 4.1 to generate two more numerical solutions:

- $\boldsymbol{v}_F$ is the positivity-constrained solution that adds in elementwise equality constraints to preserve interelement boundary (flux) values.
- $\boldsymbol{v}_{I+F}$ is the positivity-constrained solution that includes both interelement flux preservation and elementwise mass conservation.

Figure 4 investigates both h- and p-convergence of the numerical solution to (5.1) for all four numerical solutions. We observe that, regardless of the constraint that we impose, the convergence rates are unchanged, and even the error values are nearly identical for all numerical solutions. Thus, for this example, our proposed procedure can guarantee positivity (and flux/mass conservation as well, if desired) without a notable impact on the accuracy of the solver.

We repeat the same convergence study for the same PDE, but instead for a triangular hat function with nonnegative values as shown in Figure 5; also shown in the figure is the unconstrained numerical solution at $T = 1$, which violates

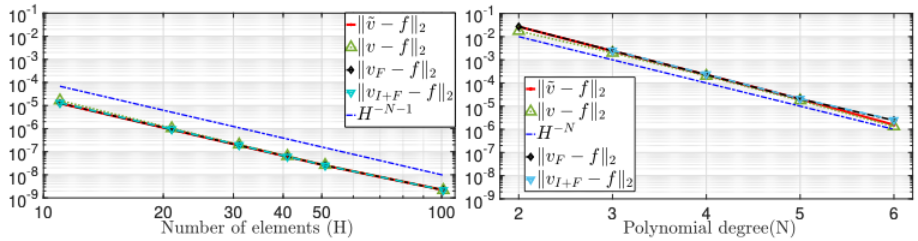

FIG. 4. *Convergence study for dG solution to the one-dimensional advection PDE applied to* $u(x,t) = 0.5\sin(2\pi x - 2\pi t - 0.5\pi) + 1$ *for a time period of* $T = 1$ *second.* $\tilde{\boldsymbol{v}}$ *refers to the unfiltered solution,* $\boldsymbol{v}$ *refers to the positive solution,* $\boldsymbol{v}_F$ *refers to the positive solution with flux (boundary values) conserved for each element, and* $\boldsymbol{v}_{I+F}$ *refers to the positive solution with flux and mass conserved for each element . Left: h-convergence using constant polynomial order* $N = 3$, $\Delta t = 10^{-5}$. *Right: p-convergence at constant number of elements* $H = 3$, $\Delta t = 10^{-4}$.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

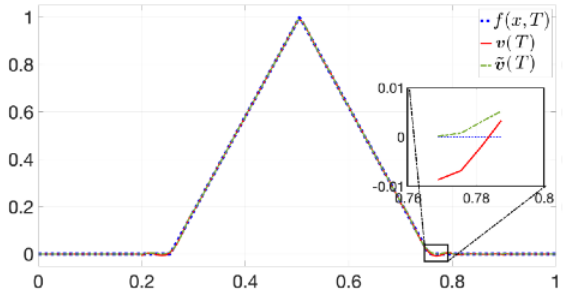

FIG. 5. *Periodic function f for convergence study of filtered one-dimensional dG solution to (5.1). Note that the $f(x,0)$ and $\tilde{\boldsymbol{v}}(T)$ overlap because of the periodic nature of f; however $\tilde{\boldsymbol{v}}(T)$ does not comply to the positivity structure of $f(x,0)$ as expected. After the application of filter, we obtain $\boldsymbol{v}(T)$, which changes $\tilde{\boldsymbol{v}}(T)$ to preserves the positive structure of f.*

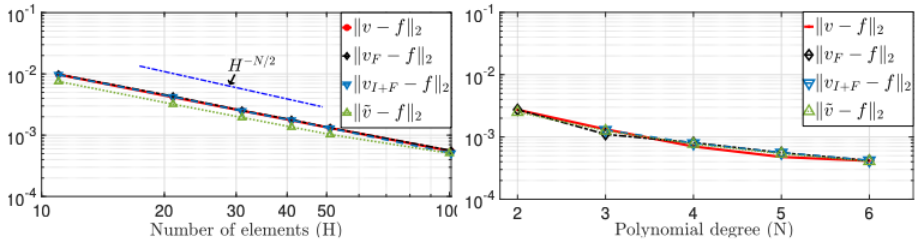

FIG. 6. *Convergence study for dG formulation of one-dimensional advection PDE applied to Figure 5, for a simulation run to $T=1$. Vector $\tilde{\boldsymbol{v}}$ represents the unfiltered solution coefficients, $\boldsymbol{v}$ refers to the positive solution coefficients at $T=1$, $\boldsymbol{v}_F$ refers to the coefficients of the positive solution with flux (boundary values) conserved for each element, and $\boldsymbol{v}_{I+F}$ refers to the positive solution coefficients with flux and mass conserved for each element . Left: h-convergence at constant polynomial degree $N=3$, $\Delta t=10^{-5}$. Right: p-convergence at constant number of elements $H=51$, $\Delta t=10^{-4}$. Here the choice of an odd number of elements for the h-convergence study ensures that the middle discontinuity of the function's derivative is located on an element boundary.*

positivity. We run the same experiment as for the previous example and plot the h- and p-convergence results in Figure 6.

From the results shown in Figures 4 and 6 we observe that the convergence for filtered solution $\boldsymbol{v}$ and its variants remains largely unchanged/comparable to the unfiltered counterpart. Regarding the cost of the filtering procedure, there is a one-time orthonormalization cost of the basis function used as the initial setup, but since we can employ the filter over each element individually, this cost is negligible. The computational cost of filtering procedure per time step depends on the time taken by the global minimum finding step of the filter, which we investigate next.

For each simulation involving the filter, we compile the number of elements per time step where the filter is employed. (Recall that we perform a cheaper check to certify positivity before calling the filter optimization.) For each of the three filtered solutions $\boldsymbol{v}$, $\boldsymbol{v}_F$, and $\boldsymbol{v}_{I+F}$, Figure 7 plots the average number of filtered elements per time step during the convergence experiments run in Figure 6.

As seen from Figure 7, the filter is not indiscriminately applied across all elements in the domain and instead is called only for a few flagged elements per time step where constraint violations are observed. This can substantially reduce the required computational overhead compared to applying the filter across every element.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

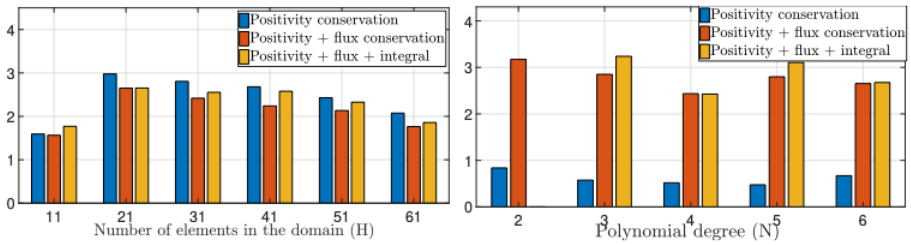

FIG. 7. *Average number of elements flagged per time step during the experiments shown in Figure 6. Left: h-convergence experiment with* $\Delta t = 10^{-5}$ *and* $N = 3$. *Right: p-convergence experiment with* $\Delta t = 10^{-4}$ *and* $H = 51$.

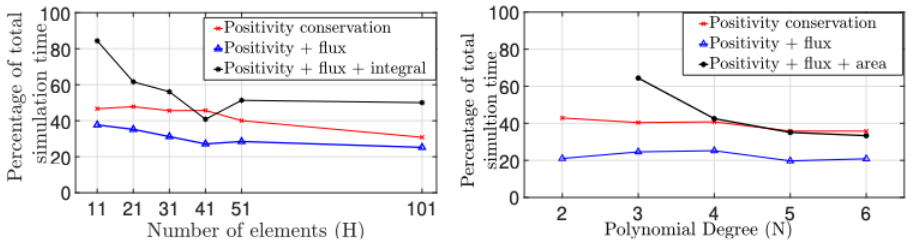

FIG. 8. *Percentage of total simulation clock time spent inside the filtering procedure during the convergence experiments shown in Figure 6. Left: h-convergence experiment with* $\Delta t = 10^{-5}$ *and* $N = 3$. *Right: p-convergence experiment with* $\Delta t = 10^{-4}$ *and* $H = 51$.

The percentage of total simulation time that is spent inside the filtering procedure for experiment Figure 6 is shown in Figure 8.

For some experiment configurations, the filtering cost can be more than half the simulation time (e.g., $H = 11$ for the black line in Figure 8 (left). However, in many cases, the filtering procedure can require less than 20% of the total computational effort. It is interesting to note that for the p-convergence experiment in Figure 8, the percentage of filtering time is *higher* for some filters with *fewer* constraints. For example, in Figure 8 (right), less simulation time is required to impose positivity and flux preservation compared to simply imposing positivity. One possible explanation for this phenomenon is that filters with a larger number of equality constraints require optimization in a lower dimensional space. Thus, the optimization problem can be less expensive to solve in this case. However, with additional constraints (e.g., the black line in Figure 8 (right), even though the dimensionality of the optimizer decreases, the number of iteration required by the filter increases. Therefore, a suitable balance must be struck from application to application.

5.2. cG. We now present the results on the cG implementation of a diffusion-reaction equation,

$$\frac{\partial}{\partial t}u(x,t) = \gamma \frac{\partial^2}{\partial x^2}u(x,t) + r(u(x,t)).$$

We consider the problem on a bounded domain $\Omega \subset \mathbb{R}$ and $\Omega = [-1, 1]$. Consider the following advecting smoothed Heaviside function as a solution to this problem:

$$u(x,t) = e^{-\gamma t}\Big(tanh(\epsilon(x + 0.4) - ct) + 1\Big),$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

where ϵ is a shape-parameter, and c refers to the speed at which the exact solution moves in space over time. The nonlinear reaction term in this experiment is set as $r(u(x,t)) = \mu u(x,t)(1-u^2(x,t))$, where μ is a constant. Our Galerkin discretization of this problem integrates r exactly, which is possible since r depends polynomially on u. A diffusion-reaction PDE with a stiff quadratic reaction term is interesting as it represents a source term with the potential to blow up and only becomes active at the discontinuity [6]. Whereas numerical schemes may converge, they may converge with approximants that violate positivity.

Using the discrete form of the continuous problem (3.1) as described in section 3, we choose the IMEX CNAB-2 scheme, which deals with the advection term using Crank–Nicolson and uses a second-order Adams–Bashforth scheme for the reaction term. The *unconstrained* cG discrete time update then is

$$\tilde{\boldsymbol{v}}^{n+1} = \boldsymbol{B}^{-1}\Big(\boldsymbol{A}\tilde{\boldsymbol{v}}^{n} + \boldsymbol{r}^{n}\Big),$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ are given by

$$\begin{aligned}\boldsymbol{A} &= \boldsymbol{M} - (0.5\gamma\Delta t\boldsymbol{L}),\\ \boldsymbol{B} &= \boldsymbol{M} + (0.5\gamma\Delta t\boldsymbol{L}),\end{aligned}$$

with $\boldsymbol{M}$ and $\boldsymbol{L}$ the mass and Laplacian matrices, respectively, and $\boldsymbol{r}^n$ the projection coefficients of the reaction function $r(\tilde{\boldsymbol{v}}^n)$.

Figure 9 shows the initial and final states of $u(x,t)$ for a simulation up to $T=1$. As described earlier, the cG formulation uses a nonorthonormal ("hat" and "bubble" function) basis, meaning that a spatially global mass matrix inversion must be performed for the filtered versions of this experiment. Advances that make this procedure efficient are currently being investigated, but Figure 10 demonstrates that, like the dG formulation, error and convergence rates are largely unaffected for this problem by inclusion of the filter. If the constraints are to be satisfied locally in the cG formulation, the order of accuracy we observe in this example may not hold. This problem is widely known in the context of flux limiters, and many fixes have been proposed, e.g., [9].

This example also demonstrates the flexibility of our approach: Although cG and dG discretizations can be very different mathematically and computationally, Algorithm 4.1 is the generic template for applying our procedure, independent of most PDE discretization details. Note that we consider global constraint satisfaction in the current example. In the case in which an additional advection term is present

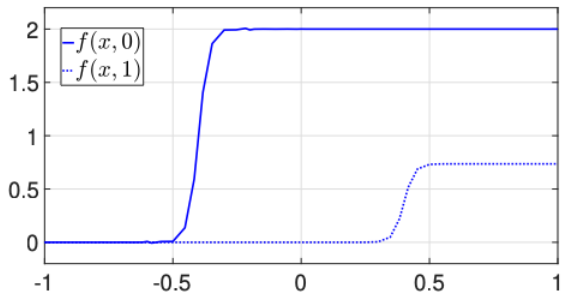

FIG. 9. *Initial and final state of the function u for $\epsilon = 25$, $c = 20$, and $\gamma = 1$, $\mu = 1$.*

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

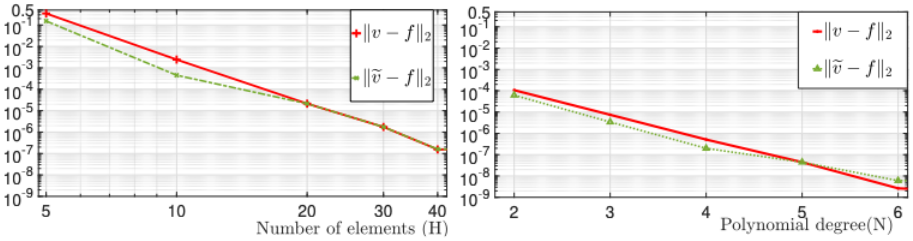

FIG. 10. *Convergence study for one-dimensional cG diffusion-reaction PDE applied to the function f shown in Figure 9 with constant time step $\Delta t = 10^{-4}$ until final time $t = 1$. Left: h-convergence at constant polynomial degrees $N = 7$. Right: p-convergence at constant number of elements $H = 100$.*

in the PDE, it conserving the structure of the solution locally is often desirable. Such conservation can be achieved by the current method after reformulating the optimization problem such that E constraints are in the constraint set, where E is the number of elements in the domain. Each constraint represents the property to be conserved in each element.

6. Conclusions. We have proposed a filtering approach applied to standard finite element time-stepping discretizations of PDE, using both cG and dG formulations in one spatial dimension. The goal of the filter is to enforce one or more constraints from a general class of convex inequalities that can apply over the entire spatial domain. Thus, we pose the filtering problem as a nonlinear optimization problem, which we propose to perform after every time step. We focus mainly on enforcing positivity, but enforcing a maximum principle or monotonicity also falls into our framework. Linear equality constraints, such as total mass preservation or elementwise boundary value preservation, are possible in our framework. Our numerical results show that the filtering (optimization) does increase in the computational cost of PDE solvers, but it often requires less than 50% of the total simulation time.

REFERENCES

[1] R. ANDERSON, V. DOBREV, T. KOLEV, D. KUZMIN, M. Q. DE LUNA, R. RIEBEN, AND V. TOMOV, *High-order local maximum principle preserving (mpp) discontinuous galerkin finite element method for the transport equation*, J. Comput. Phys., 334 (2017), pp. 102–124.

[2] R. W. ANDERSON, V. A. DOBREV, T. V. KOLEV, AND R. N. RIEBEN, *Monotonicity in high-order curvilinear finite element arbitrary Lagrangian–Eulerian remap*, Int. J. Numer. Methods Fluids, 77 (2015), pp. 249–273.

[3] H. GUO AND Y. YANG, *Bound-preserving discontinuous Galerkin method for compressible miscible displacement in porous media*, SIAM J. Sci. Comput., 39 (2017), pp. A1969 A1990.

[4] G. KARNIADAKIS AND S. SHERWIN, *Spectral/hp Element Methods for Computational Fluid Dynamics*, Oxford University Press, Oxford, UK, 2013.

[5] K. LEIDERMAN AND A. L. FOGELSON, *Grow with the flow: A spatial–temporal model of platelet deposition and blood coagulation under flow*, Math. Med. Biol., 28 (2011), pp. 47–84.

[6] R. J. LEVEQUE AND H. C. YEE, *A study of numerical methods for hyperbolic conservation laws with stiff source terms*, J. Comput. Phys., 86 (1990), pp. 187–210.

[7] D. LIGHT AND D. DURRAN, *Preserving nonnegativity in discontinuous Galerkin approximations to scalar transport via truncation and mass aware rescaling (TMAR)*, Mon. Weather Rev., 144 (2016), pp. 4771–4786.

[8] X.-D. LIU AND S. OSHER, *Nonoscillatory high order accurate self-similar maximum principle satisfying shock capturing schemes* I, SIAM J. Numer. Anal., 33 (1996), pp. 760–779.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

[9] C. Lohmann, D. Kuzmin, J. N. Shadid, and S. Mabuza, *Flux-corrected transport algorithms for continuous Galerkin methods based on high order Bernstein finite elements*, J. Comput. Phys., 344 (2017), pp. 151–186.
[10] R. Sanders, *A third-order accurate variation nonexpansive difference scheme for single nonlinear conservation laws*, Math. Comp., 51 (1988), pp. 535–558.
[11] C. W. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, J. Comput. Phys., 77 (1988), pp. 439–471.
[12] J. van der Vegt, Y. Xia, and Y. Xu, *Positivity Preserving Limiters for Time-Implicit Higher Order Accurate Discontinuous Galerkin Discretizations*, preprint, arXiv:1811.08620, 2018.
[13] Vidhi Zala, Robert M. Kirby, and Akil Narayan, *Structure-preserving function approximation*, SIAM J. Sci. Comput., to appear.
[14] J. Von Neumann, *Functional Operators (AM-22), Volume* 2: The Geometry of Orthogonal Spaces (AM-22), Ann. of Math. Stud. 22, Princeton University Press, Princeton, NJ, 1951.
[15] M. J. Zahr and P.-O. Persson, *An optimization based discontinuous Galerkin approach for high-order accurate shock tracking*, in Proceedings of the AIAA Aerospace Sciences Meeting, Kissimmee, FL, 2018.
[16] X. Zhang and C.-W. Shu, *On maximum-principle-satisfying high order schemes for scalar conservation laws*, J. Comput. Phys., 229 (2010), pp. 3091–3120, https://doi.org/10.1016/j.jcp.2009.12.030.

Downloaded 02/11/22 to 128.110.184.55 Redistribution subject to SIAM license or copyright; see https://epubs.siam.org/page/terms

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.