

# Journal Pre-proof

Convex Optimization-Based Structure-Preserving Filter for Multidimensional Finite Element Simulations

Vidhi Zala, Akil Narayan and Robert M. Kirby

PII: S0021-9991(23)00459-X  
DOI: <https://doi.org/10.1016/j.jcp.2023.112364>  
Reference: YJCPH 112364

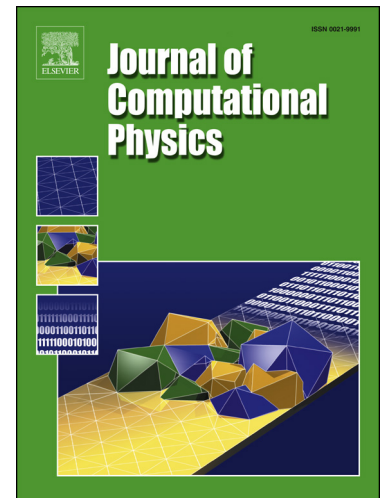
To appear in: *Journal of Computational Physics*

Received date: 15 March 2022  
Revised date: 7 July 2023  
Accepted date: 8 July 2023

Please cite this article as: V. Zala, A. Narayan and R.M. Kirby, Convex Optimization-Based Structure-Preserving Filter for Multidimensional Finite Element Simulations, *Journal of Computational Physics*, 112364, doi: <https://doi.org/10.1016/j.jcp.2023.112364>.

This is a PDF file of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive version of record. This version will undergo additional copyediting, typesetting and review before it is published in its final form, but we are providing this version to give early visibility of the article. Please note that, during the production process, errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

© 2023 Published by Elsevier.



# Convex Optimization-Based Structure-Preserving Filter for Multidimensional Finite Element Simulations

Vidhi Zala<sup>a,b,\*</sup>, Akil Narayan<sup>a,c</sup> and Robert M. Kirby<sup>a,b</sup>

<sup>a</sup> Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, UT 84112

<sup>b</sup> School of Computing, University of Utah, Salt Lake City, UT 84112

<sup>c</sup> Department of Mathematics, University of Utah, Salt Lake City, UT 84112

## ARTICLE INFO

### Keywords:

Structure Preservation  
Convex Optimization  
Finite Element Methods  
Partial Differential Equations

## ABSTRACT


In simulation sciences, capturing the real-world problem features as accurately as possible is desirable. Methods popular for scientific simulations such as the finite element method (FEM) and finite volume method (FVM) use piecewise polynomials to approximate various characteristics of a problem, such as the concentration profile and the temperature distribution across the domain. Polynomials are prone to creating artifacts such as Gibbs oscillations while capturing a complex profile. An efficient and accurate approach must be applied to deal with such inconsistencies to obtain accurate simulations. This often entails dealing with negative values for the concentration of chemicals, exceeding a percentage value over 100, and other such problems. We consider these inconsistencies in the context of partial differential equations (PDEs). We propose an innovative filter based on convex optimization to deal with the inconsistencies observed in polynomial-based simulations. In two or three spatial dimensions, additional complexities are involved in solving the problems related to structure preservation. We present the construction and application of a structure-preserving filter with a focus on multidimensional PDEs. Methods used such as the Barycentric interpolation for polynomial evaluation at arbitrary points in the domain and an optimized root-finder to identify points of interest, improve the filter efficiency, usability, and robustness. Lastly, we present numerical experiments in 2D and 3D using discontinuous Galerkin formulation and demonstrate the filter's efficacy to preserve the desired structure. As a real-world application, implementation of the mathematical biology model involving platelet aggregation and blood coagulation has been reviewed and the issues around FEM implementation of the model are resolved by applying the proposed structure-preserving filter.

## 1. Introduction

A widely used application of mathematical modeling is creating simulations used to predict and analyze different processes. Simulations obtained from numerical solutions provide physically meaningful results if the values of simulation variables at each timestep follow the *structure* of the exact solution. Structure in this context refers to properties such as non-negative values for a chemical concentration or simulating a quantity requiring an upper bound, e.g.,  $[0,1]$ , and other such aspects of the numerical solution. If the solution fails to comply with these structural restrictions, the resulting solution is not meaningful. Therefore, designing an approach to regulating the solution is crucial for simulations in many applications. Examples include thermodynamics, numerical weather predictions, biochemical processes such as platelet aggregation and blood coagulation, and fluid flow problems.

Computing solutions to mathematical models described in multiple dimensions further adds to the complexity of solving a structure-preservation problem. To implement multidimensional models using numerical methods, we require approximations of the quantities of interest in the domain. Popular choices for numerical methods such as the finite element method (FEM) and the finite volume method (FVM) employ piecewise polynomial-based approximations. These approximations are prone to artifacts such as the Gibbs phenomenon, leading to a violation of structure. We focus on the FEM implementation for the advection-diffusion-reaction (ADR) problem and design an algorithm that computes a solution at each timestep adhering to a particular structure (e.g., positivity and monotonicity). The nonphysical values from such simulations may result in a propagation of nonphysical values, which often cause a blow

\*Corresponding author

 vidhi.zala@utah.edu (V. Zala); akil@sci.utah.edu (A. Narayan); kirby@cs.utah.edu (R.M. Kirby)

ORCID(s):

up, resulting in the failure of the entire simulation. For example, when the concentration values are expected to be bound within  $[0, 1]$  but the numerical solution produces a value greater than one, even a small change in the next timestep causes a rapid compounding of the solution.

The problem of structure preservation has been well studied in different fields for applications to various partial differential equations (PDEs) and using different numerical approaches. The next section provides a brief overview of some existing methods proposed to solve the structure-preservation problem.

### 1.1. Review

Many sophisticated approaches have been considered in the literature to tackle different aspects of the structure-preservation problem. We make a broad classification into ‘intrusive’ and ‘nonintrusive’ classes of solutions. Many popular approaches generally involve changing either the solution or the PDE and require the imposition of stringent conditions on solver parameters, such as the step-size of the numerical method or the domain shape and granularity. Such methods can be classified as intrusive. On the other hand, some methods constrain the solution obtained from the solver with minimal, often superficial, change to the numerical scheme, i.e., in a nonintrusive way.

Prominent among the intrusive class of approaches is the transformation of structure preservation into a PDE-constrained optimization problem, methods that modify the spatial discretization [1], and limiters derived from Karush-Kahan-Tucker (KKT) optimality conditions [2]. Another example of the intrusive approach [3] proposes an operator-splitting positivity-preservation method based on the energy dissipation law.

The strategies for constraint satisfaction in [4] consider a version of the problem that is a specialization of the one solved by the solution proposed in this manuscript, which falls under the nonintrusive class. In [5, 6] and extensions [4, 7], the authors explore approaches for positivity preservation on the basis functions derived from Bernstein polynomials such that they are non-negative and possess a partition of unity property. Therefore, the interpolated solution respects the original bounds at any point in the domain, nonintrusively. Other nonintrusive methods prescribe solutions such as limiters or truncation and modification of the domain (e.g., curvilinear mesh adaptation) to adhere to the desired solution structures. One such method is established in [8] where the authors present a new approach for multi-material arbitrary Lagrangian–Eulerian (ALE) hydrodynamics simulations based on high-order finite elements posed on high-order curvilinear meshes in which conservative fields are remapped and additional synchronization steps are introduced for structural preservation. Additionally, many nonintrusive methods perform pre- or postprocessing of intermediate solutions without modifying the underlying problem to incorporate the constraints or making changes to the domain. Some nonintrusive approaches to structure preservation can be found in [9, 10, 11, 12, 13]. In [14], the authors present a positivity-preserving limiter by maintaining cell averages within a certain tolerance. The work in [15] imposes positivity on a discrete grid for kinetic equations via a nonlinear filtering procedure. Satisfying constraints up to a numerical tolerance is sufficient for many applications. However some applications require the solution to adhere to a strict non-negative structure. For example, in [16], the authors enforce strict, global non-negativity of the diffusion propagator by formulating constraints specific to the propagator model. The algorithm in [17] employs alternating least squares procedures to implement general linear equality and inequality constraints by reducing it to a non-negativity-constrained least squares (NNLS) problem. The non-negativity constraints based on square root representations have also been of interest in many other works [18, 19]. We propose a nonintrusive method based on a general filtering approach in [20, 21].

Many of the existing techniques prescribe approaches to preserve the structure that work only for low-order problems. Additionally, an increase in the dimensionality of the problem further exacerbates the issue of accuracy and convergence. Many approaches struggle to maintain high-order accuracy and convergence robustly, including the filter’s ability to preserve structure without any additional dependence or restrictions imposed on a per-case basis. This includes problem-specific modifications as well as varying solutions based on other aspects of the problem. For example, [22, 23] presents redistribution-based local and global repair methods depending on a sweep through cells to locate and rectify the violating structures. The final solution depends on the sweep order, producing different solutions for different sweep orders. In addition, this repair procedure preserves element-averaged bound/positivity constraints at particular finite difference nodes. It does not preserve them pointwise, e.g., at every spatial location. The proposed method tries to preserve the constraints at every spatial location. An ideal approach would achieve a balance between

robust structure preservation, maintaining accuracy as well as convergence, and providing usability by deterministic termination within an acceptable time.

## 1.2. Contribution

We pose the problem of preserving the structure in polynomial-based numerical methods as a convex optimization problem. Our previous works discuss a filtering algorithm to solve this problem and demonstrate the solution for function approximations [20] and PDE solutions in 1D [21]. The robust design of the problem enables the preservation of different properties of the solution simultaneously (e.g., positivity, monotonicity, and boundedness). It transforms the problem into a composite constraint satisfaction problem that can be solved by convex minimization. To address this problem, we propose a filter that can be applied as a postprocessing step to deal with structural inconsistencies. The novel idea in the design of the filter is the efficiency achieved by applying the filter only to the violating regions in a subdomain and structure-preservation continuously *throughout* the domain, and not just a subset of points within the domain. This guarantee of structure-preservation depends on the granularity of the global minimization procedure in 2D and 3D, which is expensive and less reliable compared to its 1D counterpart. In 1D, the minimum can be found accurately using the confederate matrix approach to find zeros of the derivative of a polynomial. For finding the critical points in higher dimensions, an efficient gradient-based method is required which solves the nonconvex problem of finding the global minimum. In the case of the proposed filter, the global minimum represents a structure violation. Gradient-based methods are prone to get stuck at a local minimum, causing it to miss the point of larger structural inconsistency, thus failing to provide strong guarantees of preservation. Another important aspect that affects the outcome of the gradient-based method is the choice of a starting point. To address this aspect, we propose an investigation of structural inconsistencies on a lattice of points as the initial step in the gradient-based method. The point with the most inconsistency is chosen as a starting point. The accuracy of the minimum found depends on the pattern and granularity of the initial lattice used, which is detailed in Section 2.1.

To streamline the minima-finding stage and reduce the time and space complexity, we adopt the Barycentric interpolation approach from [24], expanded upon by [25]. Using these techniques, a considerable speed-up in time taken by minimization and overall filter computations is achieved.

This paper focuses on designing a structure-preserving filter using a convex optimization approach for 2D and 3D problems on different element types. We will present the formulation derived from [20] in Section 2 and extend the concepts to more complex problems tackled in this paper. The remainder of the paper is organized as follows: Section 2 discusses the setup for applications of the filter to time-dependent PDEs in 2D and 3D. Section 3 introduces the notations, details the design of different building blocks of the filter, and summarizes the 1D problem formulation from [20, 21]. A procedure for the filtering process developed for multidimensional applications is presented in Section 4. Section 5 describes the numerical results to demonstrate the filter's efficacy in preserving the desired structures in different application scenarios. This section is divided into subsections describing the process of choosing the parameters to run the experiments, the 2D and 3D canonical experiments, and an advection problem on different homogeneous meshes using the discontinuous Galerkin (dG) formulation. We conclude with a demonstration of the use of the proposed filter on a real-world application: the model of platelet aggregation and blood coagulation problem.

## 2. Setup

In this section, we discuss the setup for filter applications to function approximations and time-dependent PDEs to solve an advection problem using the FEM with method-of-lines discretizations with a focus on the dG formulation. The choice of advection problem is for illustration purposes only. The setup remains the same for any time-dependent PDE solution using a polynomial-based method. The filter behaves as a postprocessing step that preserves the desired structure at each timestep within a defined tolerance and is agnostic to the numerical method used to obtain the solutions. In all our experiments, we consider the numerical zero to be  $10^{-7}$ . The idea behind this choice is that in all the positivity preserving experiments the gradient-based method stagnates when the difference in minimum found by two consecutive iterations dip below  $-10^{-7}$ . In such a situation, there are no significant changes to the interpolation coefficients between iterations of the filter, and therefore, the convergence is considerably slow. Certain variations of the filter as investigated in [20, Section 5.1.2] can improve the rate of convergence below a particular tolerance.



Section 5 shows numerical examples in higher dimensions using the same setup on element types such as triangle, quadrilateral, hexahedron, tetrahedron, prism, and meshes comprised of these element types.

### 2.1. Detecting and resolving structural inconsistency in 2D and 3D

If the polynomial projections lose structural conformity, they are likely to do so at or between the quadrature points. We hope to capture the structural inconsistencies by checking the values on a lattice of quadrature points and the centroids formed by adjacent quadrature points. Here, the arbitrary choice of the centroids is one of the many possible options to choose the lattice. A quadrilateral is used as a sample element to describe the setup; however, the same applies to any canonical element type in 2D and 3D. Let us call such a lattice of points ( $X$ ). For a quadrilateral, let  $Q$  represent the number of quadrature points in one dimension.  $X$  is defined by a combination of the quadrature grid on the quadrilateral (total  $Q^2$  points and  $(Q - 1)^2$  centroids formed by the midpoints of the quadrature grid) as defined in (1).

$$X = \{Q^2 \text{ points in the quadrature grid}\} \cup \{(Q - 1)^2 \text{ points in the staggered quadrature grid}\}, \quad (1)$$

For a triangle, we can follow the same idea of combining the quadrature points and the centroids to construct the lattice. An example lattice on a mesh with quadrilaterals and triangles is shown in Figure 1.

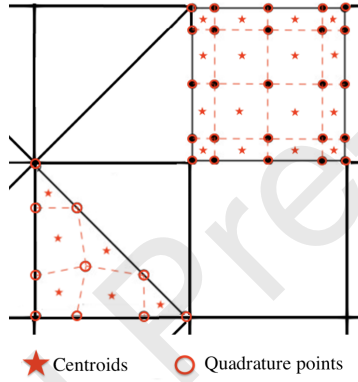


Figure 1: Example lattice used for detecting structural nonconformity in a mesh.

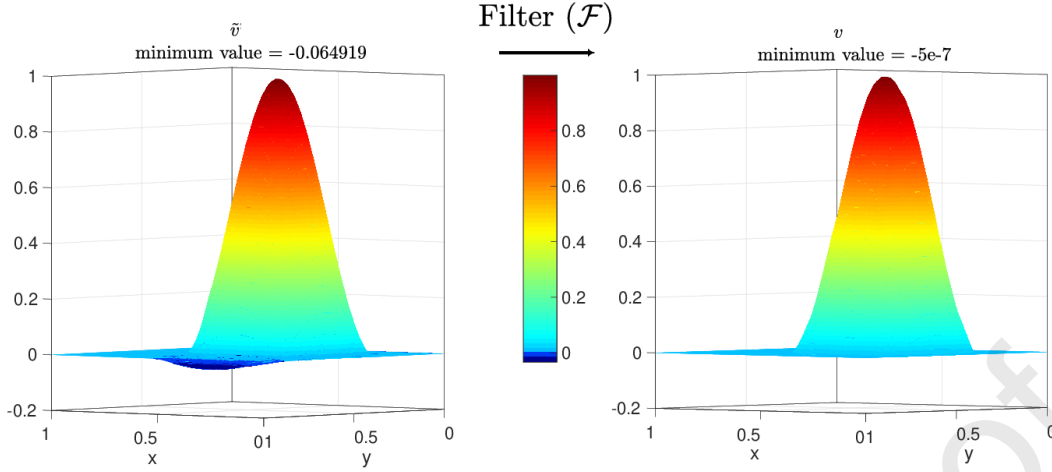
Consider a function in 2D defined by (2).

$$f(x, y) = v \sin(2\pi x) \sin(2\pi y - 0.85\pi), \quad (2)$$

where

$$v(x, y) = \begin{cases} 1, & \text{if } x \geq 0 \text{ and } x \leq 0.5 \text{ and } y \geq 0.4 \text{ and } y \leq 0.85 \\ 0, & \text{otherwise.} \end{cases}$$

Projecting (2) using polynomial order  $N = 5$  on a quadrilateral in the domain  $\Omega = [0, 1] \times [0, 1]$ , we get the coefficients  $\tilde{v} = \{\tilde{v}_j\}_{j=0}^{P-1}$ . The projection is shown in the left subfigure of Figure 2. In this case,  $P = (N + 1)^d$ , where  $d$  is dimension. Since  $d = 2$ ,  $P = 36$ . The original function is non-negative, so the projection should preserve positivity. To this end, we employ the optimization outlined in Section 4 as a postprocessing filter. The process behaves as a spectral filter based on the empirical evidence presented in [20, Section 5.2] and further theoretical explanation provided in [20, Proof 5.1]. We will hereafter refer to this optimization as a nonlinear filter ( $\mathcal{F}$ ). Let  $v$  be the filtered version of  $\tilde{v}$ . The filtered projection is shown in the right subfigure of Figure 2.



**Figure 2:** Left: Projection of (2) using polynomial order = 5. Right: Filtered version after application of the structure-preserving filter. The filter converges to a solution that preserves positivity within a tolerance (set to  $10^{-6}$  here).

## 2.2. Solution to advection problem in dG

The advection equation for a quantity described by a scalar field  $u$  is expressed mathematically by a continuity equation (3) in the domain  $\Omega \subset \mathbb{R}^d$ .

$$u_t + \mathbf{a} \cdot \nabla u = 0, \quad (3)$$

where  $\mathbf{a}$  represents the velocity of advection. Assume proper initial and boundary conditions are specified.

Galerkin-type methods assume an ansatz for  $u$  as a time-varying element of a fixed  $P$ -dimensional linear subspace  $V$  that contains the polynomial functions for a fixed degree  $P$ , where frequently  $V \subset L^2(\Omega)$ .

$$u(x, t) \approx u_P(x, t) := \sum_{i=0}^{P-1} \hat{v}_i(t) \phi_i(x), \quad V = \text{span}\{\phi_0 \dots, \phi_{P-1}\}, \quad (4)$$

where  $x \in \Omega$ , and  $\{\phi_j\}_{j=0}^{P-1}$  represents the traditional FEM basis, which is not necessarily orthogonal. Let  $\{\psi_j\}_{j=0}^{P-1}$  represent a collection of orthonormal basis functions. We can transform the vector of coefficients  $\hat{\mathbf{v}}$  into its orthonormal form  $\tilde{\mathbf{v}}$ .

Consider a partition of  $\Omega$  consisting of  $E$  subdomains defined by  $\tau(\Omega) = \{e_0, e_1, \dots, e_{E-1}\}$ . The discontinuous Galerkin formulation assumes that  $V$  is comprised of functions that are polynomials of a fixed degree  $P$  on each element on  $\Omega$ ; where discontinuities in derivatives are allowed only at partition boundaries. The semidiscrete form for (3) is derived in the standard Galerkin way by using the ansatz (4) 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, numerical flux and/or stabilization terms are included in the weak formulation.

The result is a system of ordinary differential equations prescribing time-evolution of the discrete degrees of freedom represented by vector  $\tilde{\mathbf{v}} = \{\tilde{v}_0, \dots, \tilde{v}_{(P-1)}\}$ .

$$\mathbf{M} \frac{\partial}{\partial t} \tilde{\mathbf{v}} + \mathbf{A} \tilde{\mathbf{v}} = \mathbf{F}(\tilde{\mathbf{v}}) \quad (5)$$

where  $\mathbf{M}$  and  $\mathbf{A}$  are the  $P \times P$  mass and advection matrices, respectively, defined as

$$(M_{i,j}) = \langle \phi_i, \phi_j \rangle, \quad (A_{i,j}) = \left\langle \phi_i, a \frac{\partial}{\partial x} \phi_j(x) \right\rangle,$$

and  $\mathbf{F}$  is a general term to compensate for any numerical fluxes or stabilization terms. Finally, (5) is transformed into a fully discrete system that can be solved using an appropriate numerical integration scheme.

Consider the case when the solution defined by  $\hat{v}^n$  on a particular partition  $e$  at a timestep  $n$  does not satisfy the desired structural properties. To preserve the structure, we employ the optimization defined in (8) as a postprocessing filter. Let  $v$  represent the filtered solution that satisfies the structure. The process can be illustrated in Equation (6). The proposed filter works by simply augmenting this scheme nonintrusively.

$$v^n \xrightarrow{\text{Timestepper for (3)}} \hat{v} \xrightarrow{(F)} v, \quad (6)$$

Since DG discretizations have degrees of freedom that are decoupled across the subdomains, the total degrees of freedom in the optimization step need to be the only ones over the subdomain. In addition, the optimization needs to be applied only in the subdomains that have structural violations discovered at the lattice points, which leads to a parallelizable set of independent optimizations each in  $P$  dimensions. We can filter the selected subdomains simultaneously, which further adds to the procedure's efficiency.

For a mesh with the number of elements  $E > 1$ , the process of enforcing positivity per element leads to changes in solution properties; in particular, elementwise boundary values and the mean of the discrete solution are not preserved. Assume that numerical fluxes are computed as explicit functions of element boundary values. A shift in element boundary values by the filter causes changes in the corresponding fluxes, thus adding errors to the simulation. We can resolve this issue by imposing additional equality constraints (i.e., function values at element boundaries) in the filter. However, for the flux conservation, we cannot make an assertion that the resulting filtered solutions should satisfy the original flux/jump conditions or some updated conditions, based on the new filtered basis coefficients. In the case of mass (integral) conservation, additional equality constraints can be imposed. The values of fluxes are often used to ensure the properties of numerical schemes therefore by preserving fluxes one can attest that the properties of the original scheme have been retained. Preserving fluxes in this context corresponds to corrections in one element need not have an impact on neighboring elements, and one way to promote isolation of the optimization effect to the element under consideration is to ensure that interelement communication through fluxes is undisturbed. The formulation for flux conservation in [21, Section 3.2 and 3.3] is unique to one-dimensional scenarios. Generalization to higher dimensions is not immediate. Additional analysis is needed to incorporate different elements and, therefore, beyond the scope of this paper. The robust design of the filter allows for the incorporation of an arbitrary number of structural constraints into one optimization problem. However, increasing the number of constraints reduces the degrees of freedom for the filter accordingly. In particular, if the filter has  $P$  degrees of freedom and we have  $q$  linear equality constraints, then the dimension of the optimization problem can be reduced to  $(P - q)$  using the procedure described in [21, Section 3.2].

### 3. Formulation of structure-preservation problem and solution design

This section establishes the notations and summarizes formulation, filter design, and implementation ideas from [20, 21].

#### 3.1. The problem

Let  $u(x) \in V$  be a function, where  $V$  is a finite-dimensional subspace of a Hilbert space of real-valued functions on  $\Omega \in \mathbb{R}^d$ . Let  $V$  contain a collection of orthonormal basis functions  $\{\psi_n\}_{n=1}^N$  for some  $N \in \mathbb{N}$ .

Therefore, we have

$$V = \text{span} \{ \psi_1, \dots, \psi_N \}, \quad \langle \psi_i, \psi_j \rangle = \delta_{ij}, \quad i, j = 1, \dots, N,$$

where  $\langle \cdot, \cdot \rangle$  is the inner product on  $V$ , and  $\delta_{ij}$ , the Kronecker delta function.

The value of  $d$  is set to 1 for simplicity. The formulation follows the same steps for higher dimensions. A comprehensive constraint-satisfaction problem can be constructed as detailed in this section.

Let  $K$  be the collection of  $u$  that satisfies the constraints. Therefore,  $K$  is the feasible subset of  $V$ . An example of such a family of linear constraints is {positivity, boundedness}. Consider a family of linear constraints such as the one in (7), where  $L$  is a linear operator bounded on  $V$ , and  $\ell$  is a function on the domain  $\Omega$ .

$$L(u) \leq \ell(x), \quad x \in \Omega. \quad (7)$$

Note that the framework in [20] allows a finite number of such constraints to be considered simultaneously.

According to the Riesz representation theorem, if  $V^*$  is a dual of  $V$ , then a functional  $L \in V^*$  can be associated with a unique  $V$ -representer  $\ell \in V$  satisfying

$$L(u) = \langle u, \ell \rangle, \quad \forall u \in V.$$

Furthermore, this  $L \leftrightarrow \ell$  identification is an isometry. We will use these facts in what follows. Given  $L$  that identifies  $\ell$ , we consider the coordinates  $\hat{\ell}_j$  of  $\ell$  in a  $V$ -orthonormal basis,

$$\ell(x) = \sum_{j=1}^N \hat{\ell}_j \psi_j(x), \quad \hat{\ell}_j = \langle \ell, \psi_j \rangle = L(\psi_j).$$

Then we have the following relations:

$$\|L\|_{V^*} = \|\ell\|_V = \|\hat{\ell}\|_2, \quad \hat{\ell} = \left( \hat{\ell}_1, \hat{\ell}_2, \dots, \hat{\ell}_N \right)^T,$$

where  $\|\cdot\|_2$  is the Euclidean norm on vectors in  $\mathbb{R}^N$ .

The problem is essentially that of developing a map  $\mathcal{F}$  from a function  $u \in V$ , to a unique function  $\mathcal{F}(u) \in K$ . Initially,  $u$  may not satisfy the structural constraints but its mapped version  $\mathcal{F}(u)$  does. In [20] we introduce a strategy to solve the following well-posed convex feasibility problem.

$$\mathcal{F}(u) := \operatorname{argmin}_{f \in K} \|u - f\|_V, \quad (8)$$

Under certain conditions, such as if  $0 \in K$ ,  $\mathcal{F}$  is norm-contractive; therefore, the operation can be called a filter. For a brief discussion of the norm-contractive nature of (8), see [20, Proposition 5.1].

### 3.2. Toward construction of a convex optimization-based solution to (8)

We first transform the continuous problem (8) to a discrete version to construct a feasible solution. Let  $\{\psi_j\}_{j=0}^{P-1}$  be a collection of orthonormal basis functions in  $V$ , different from the standard FEM basis ( $\phi$ ), which are not orthonormal.

Consider  $C$  as the affine conic region in  $\mathbb{R}^P$  corresponding to  $K \subset V$ . From previously established notations,  $\tilde{v}$  is a vector of expansion coefficients of  $u$  that does not satisfy that desired constraints, and  $v$  is a vector of filtered coefficients of  $u$  that satisfies the desired constraints. Representing the Euclidean 2-norm on vectors as  $\|\cdot\|_2$ , and the fact that  $C \subset V$ , we get the equivalent of the optimization problem (8) as follows:

$$\operatorname{argmin}_{v \in C} \|\tilde{v} - v\|_2. \quad (9)$$

For a fixed  $x \in \Omega$ , let  $H_x$  be a  $(P-1)$ -dimensional planar surface defined by an equality constraint corresponding to (7) for a fixed  $x$ . Therefore,  $L(u) = \ell(x)$ . Writing  $u$  as an expansion in degrees of freedom  $\tilde{v}$  that corresponds to a single linear equality constraint for the  $\tilde{v}$ , i.e., a  $(P-1)$ -dimensional plane. In a geometric sense,  $H_x$  is a surface

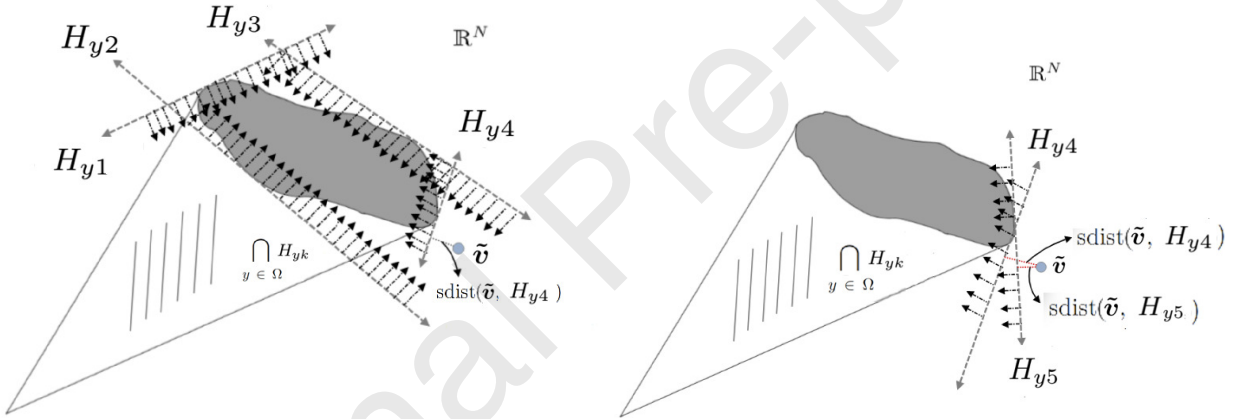
representing the constraint boundary in the space  $\mathbb{R}^P$ , dividing it into two hyperspaces as shown in Figure 3. One hyperspace represents the region that satisfies a linear inequality constraint and the other that does not. For a single linear constraint,  $C$  can be defined in terms of hyperspaces as (7).

$$C = \bigcap_{x \in \Omega} \{\text{feasible halfspace defined by plane } H_x\},$$

To obtain the full feasible set  $C$ , the filter iteratively projects  $\tilde{\mathbf{v}}$  onto the intersection of feasible halfspaces defined by individual hyperplanes  $\{H_x\}$ . To describe the feasible regions further, let us consider a geometric interpretation. Let the current state vector of the projection defined by  $\tilde{\mathbf{v}}$  be a point shown by the blue dot in Figure 3. The filter works by applying correction (13) to  $\tilde{\mathbf{v}}$  by computationally inspecting the signed distance function (10).

$$s(x) := \text{sdist}(\tilde{\mathbf{v}}, C_x) = \begin{cases} -\text{dist}(\tilde{\mathbf{v}}, H_x), & x \notin C, \\ +\text{dist}(\tilde{\mathbf{v}}, H_x), & x \in C. \end{cases} \quad (10)$$

For the positivity-preservation example, the process of “inspection” means the ability to compute the global minimum of  $s(x)$  to determine a region where  $s$  is negative. Based on this inspection, the algorithm projects the state vector of the current iterate  $\tilde{\mathbf{v}}$  onto  $H_y$  for some  $y \in \Omega$ . This projection can cause a particular  $H_{y1}$  with a positive  $s(y1)$  to go negative. Therefore, we need to repeat the projection step until the solution  $\tilde{\mathbf{v}}$  lies completely in (or on the boundary of)  $C$ . [20, Algorithm 1] summarizes all the steps of the filter.



**Figure 3:** Division of the space into half-spaces by the hyperplanes representing constraint boundaries defined by the set of points (say  $y_1, y_2 \dots y_n$ ) for the initial projection with coefficients  $\tilde{\mathbf{v}}$ . The algorithm greedily calculates the correction  $s(x)$ . Left: A geometrical visual of the distance calculation from  $\tilde{\mathbf{v}}$  to the hyperplanes defining the boundaries of the constraints. The hatched area inside the cone represents the feasible region. Right: Projection of  $\tilde{\mathbf{v}}$  on to  $H_{y4}$ .  $H_{y4}$  is selected over  $H_{y5}$  since it defines the violating hyperplane that is farthest away.

At each iteration, the spatial point  $x$  is found that minimizes the signed distance function given by (11). Note that for more than one constraint we need to keep track of  $x$  and the constraint specific to the minimum signed distance function.

$$(x^*) := \underset{x \in \Omega}{\operatorname{argmin}} s(x) \quad (11)$$

This greedy procedure is an infinite-constraint generalization of Motzkin’s relaxation method [26]. The major computational work in the filtering procedure is to minimize the objective defined by (12) at each iteration.

$$s(x) = \lambda(x) (L_x(u) - \ell(x)), \quad \lambda^2(x) := \frac{1}{\sum_{j=0}^{P-1} (L_x(\psi_j))^2}. \quad (12)$$

Next, update  $\tilde{\mathbf{v}}$  by projecting it onto the hyperplane corresponding to  $\mathbf{x}^*$ :

$$\tilde{\mathbf{v}} \leftarrow \tilde{\mathbf{v}} + \mathbf{h}(\mathbf{x}^*) \min \{0, s(\mathbf{x}^*)\}. \quad (13)$$

where  $\mathbf{h}(\mathbf{x},)$  is the normal vector corresponding to hyperplane  $H_x$  of a single constraint family that points toward  $C$ . This vector is readily computable from the orthonormal basis, see [20] for details. This procedure is repeated until  $s(\mathbf{x}^*)$  vanishes to within a numerical tolerance.

Finding the violating hyperplane that is farthest away corresponds to finding the global minimum of  $s(\mathbf{x})$  on  $\Omega$ . Unlike the 1D procedure described in [20], multidimensional global minimization requires a more expensive approach such as gradient descent (GD). In this paper, we implement 2D and 3D global minimization using line-search-based GD with backtracking. We further provide a strategy to investigate the times taken for convergence and the accuracy of this GD approach by varying the values of the parameters. Then, we choose the best values for the parameters for the accuracy of the method. Section 4.1 provides details of the GD variant chosen for filter implementation and corresponding parameter selection to efficiently solve the minimization problem.

## 4. Algorithm

The structure of the solution is preserved by applying the filter as a postprocessing step that corrects  $\hat{\mathbf{v}}$  obtained from the timestepping algorithm using iterative optimization. Each iteration of the filter involves the following crucial computational processes:

1. A GD-based minimization to solve (11), which is a part of the global search for structural violations.
2. A calculation of the correction to  $\hat{\mathbf{v}}$  using (13)

We briefly discuss the algorithmic details and the choices made to streamline the processes and their applications to the PDE timestepping.

### 4.1. Finding the global minimum

An important and the most expensive part of the optimization process is the global minimization (11). For efficiency and usability of the filter, we need to fine-tune the most expensive step, i.e., minimizing the scaled distance function (12), which quantifies the difference between the current state of the coefficients and the coefficients that satisfy the constraints at a critical point of the current iteration. The success of the proposed filter depends on accurately finding the points in the domain that violate the structure. Since the problem formulation uses dG, it can be solved element by element on a domain with  $E$  elements. Applying the filter at each timestep is essentially solving  $E$  optimization problems at every timestep. Each optimization problem is over a single element. We can reduce the number of optimization problems by testing the solution for structural inconsistencies at a lattice of points defined by (1). For example, in the case of positivity preservation, if the minimum value of an element is nonpositive, the filter is applied to that element.

In the 1D case, the filter finds the minimum value by evaluating the solution at the critical points on the element. If the subspace  $V$  restricted to an element is a space of polynomials, then the critical points of (12) are defined by the roots of the derivative of (12). For the root-finding problem, we employ the confederate matrix-generation approach. Relative to 1D, minimization in higher dimensions is more complex and expensive. Many variants of the gradient descent-based (GD-based) methods have been proposed to perform this operation efficiently and accurately. These methods operate only up to a certain tolerance and, like all the GD-based methods, have the potential for stagnation.

Since there is no multidimensional version of the confederate linearization approach that exactly solves the first-order optimality conditions, the problem of finding the critical points on the multidimensional domain is nonconvex. Therefore, we cannot ensure that all such points will be flagged for correction by any particular minimization algorithm. From our analysis, one suitable GD approach for this job is to use adaptive descent size and the ability to retrace the steps. For this reason, we choose the method of steepest descent with a backtracking line search strategy. The backtracking line search starts with a relatively large step size and repeatedly shrinks it by a factor  $\gamma$  until the Armijo–Goldstein condition (14) is fulfilled. This widely used algorithm makes an informed choice about step size and direction



at each iteration to efficiently arrive at the optimum value. It avoids stagnation by tracing back its steps if the difference between descent values falls below a fixed tolerance. An essential part of this algorithm is choosing the parameters denoted by  $c \in (0, 1)$  and  $\gamma \in (0, 1)$  that determine the step size and backtracking performed by the algorithm. For an objective function  $f$ , with a starting position  $x$ , given parameters  $c$  and  $\gamma$  the Armijo–Goldstein condition is defined as (14).

$$f(x + \gamma_j p) \leq f(x) + \gamma_j c \nabla f(x)^T p \quad \forall j \text{ until convergence,} \quad (14)$$

where  $\nabla f(x)^T$  is the slope in the direction of descent  $p$  and  $\gamma$  is calculated as

$$\gamma_j = \begin{cases} \gamma, & \text{if } j = 0 \\ c\gamma_{j-1} & \text{otherwise.} \end{cases}$$

$c$  and  $\gamma$  play a vital role in testing the condition [27], which determines whether a step-wise movement from a current position to a modified position achieves an adequately corresponding decrease in the objective function. Although many recommendations exist [27, 28, 29, 30, 31] for an optimal way to choose  $c$  and  $\gamma$ , these parameters must be tuned for maximum efficiency.

#### 4.2. Applying the structure-preserving filter to $\tilde{v}$ obtained from the timestepper

From the discussion in Section 2, the optimization that enforces a constrained solution is applied as a postprocessing part inside the timestepper, independent of the choice of the timestepping routine. As predicted by the ‘curse of dimensionality’, structure preservation and optimization for higher dimensional problems are more complex than their 1D counterparts. Since the central idea of the filter is agnostic to the dimensions of the problem, the Algorithm 1 remains the same as described in [20]. Similarly, the applications to PDE solutions follow the procedures established in [21] for 1D problems.

---

##### Algorithm 1 Constrained PDE timestepping

---

```

1: Input: Terminal time  $T$ , timestep size  $\Delta t$ , PDE solver spatial basis  $\phi$ 
2: Define nsteps =  $\frac{T}{\Delta t}$ 
3: for  $i = 0, \dots, \text{nsteps}$  do
4:   Solve PDE and obtain the orthogonalized coefficients  $\tilde{v}_e^i \in \mathbb{R}^P$  for all elements in  $e$  in  $\Omega$ 
5:   for Each element  $e \in \Omega$  do
6:     while True do
7:       if  $\exists (x \in \Omega)$  such that  $s(x) < 0$  then
8:         Compute  $(x^*)$  as defined in (11) via GD line search from Section 4.1
9:       else
10:        break
11:       end if
12:       Update  $\tilde{v}_e^{i+1}(t)$  via (13).
13:     end while
14:     Append to global  $\mathbf{v}^{i+1} = [\mathbf{v}^{i+1}, \tilde{v}_e^{i+1}]$ 
15:   end for
16: end for

```

---

Conversion from an input coefficient vector  $\hat{\mathbf{v}}$  to corresponding coefficients  $\tilde{\mathbf{v}}$  in an orthonormal basis is the first stage of the filtering procedure. Algorithm 1 presents a summary of steps taken by the PDE solver for the filter application.

A significant chunk of work in the iterative correction stage is done by evaluating the solution at the lattice points using basis interpolation, an expensive operation that involves processing and storage of large interpolation matrices. To

make the processing computationally efficient, we use the Barycentric interpolation method from [24], expanded upon by [25]. This method reduces the number of calculations and storage required to evaluate the basis functions at arbitrary points in the domain, thereby reducing the cost of the iterative correction stage. Although the implementation details of Barycentric interpolation method are beyond the scope of this paper, for the minimization part of the numerical experiments in Section 5, we observe a significant speed-up using the Barycentric approach.

## 5. Main results

This section presents four-part numerical results of the filter's application to different types of problems in 2D and 3D. Since applying multiple constraints together can be reduced to one optimization problem on a smaller solution space, the numerical results presented here focus on positivity preservation. For 1D examples that preserve multiple structural constraints simultaneously, refer to the results presented in [21].

Section 5.1 details the process to choose the parameter values for the GD line search algorithm described in Section 4.1. Section 5.2 shows the results of applying the filter to a single-element function projection in multiple dimensions. Section 5.3 presents the results and analysis of the application of the filter using the dG formulation of the advection problems defined on composite and homogeneous domains, respectively. Finally, in Section 5.6 we present the results of the FEM solution to the mathematical-biology problem of platelet aggregation and blood coagulation (PAC) with and without the application of the filter. The PAC model is posed as an advection-diffusion-reaction (ADR) system of PDEs.

### 5.1. Parameter selection for backtracking line search used in GD-based minimization

To fix the values for  $c$  and  $\gamma$  (14) for the numerical results in Section 5, we first choose a set of functions. The choice of the functions depends on factors such as the constraint type and the domain geometry. For example, for an application where positivity is the primary structural concern, a list of functions with values close to zero is a good choice. For applications involving discontinuities in the function or the domain, discontinuous functions would be a good choice. The feasible space for  $c$  and  $\gamma$  is  $(0, 1)$ . A non-comprehensive sampling on  $(0, 1)$  is performed for both parameters, and the GD is called with the parameters set to permutations of the sampled values. `GD_linesearch` denotes the call to the GD with line search. As its output, the `GD_linesearch` routine provides the number of iterations (denoted by *niter*) taken by the algorithm. The error refers to the difference between the minimum value found and the known minimum value.

As a large number of GD iterations per filter iteration makes the filtering process more expensive, choosing the parameter values for which GD converges in the least number of iterations is desirable. Another important quantity to consider here is the error. It is desirable to have GD return with a minimum value as close to the known (or golden) minimum value as possible. With these quantities (*niter* and *err*) as the metrics, we prescribe a procedure to select the ideal value of  $c$  and  $\gamma$  in Algorithm 2 [Appendix A]. The selection criteria narrow the samples to a range of  $c$  and  $\gamma$  with the minimal number of iterations and then pick from that subset the ones with the smallest error. In the case of a tie, all the  $c$  and  $\gamma$  values are included in the return variables. The procedure can be summarized as taking a list of functions that fairly represent the complexity of space and returning two ranges of ideal values: one for each of the parameters  $c$  and  $\gamma$ , respectively. Since the quadrilateral and hexahedron represent richer space than their other canonical 2D and 3D counterparts, Algorithm 2 [Appendix A] is applied and analyzed only on a quadrilateral for 2D and a hexahedron for 3D.

Future developments in the approaches for finding optima in multiple dimensions may improve the efficacy and efficiency of the filter. To this end and to provide robustness, we employ the object-oriented principle by modularizing the minimization part of the filter.

We consider different example functions (15) on the 2D domain  $[-1, 1] \times [-1, 1]$  and (16) on the 3D domain  $[-1, 1] \times [-1, 1] \times [-1, 1]$ , which have varying complexities and projection orders to set the GD parameters comprehensively and fairly.

$$f_0(x, y) = (x + 0.6)^2 + (y - 0.2)^2.$$

$$f_1(x, y) = -\sin((x - 0.1) + 0.5\pi) \cos(y - 0.2).$$

$$f_2(x, y) = \begin{cases} 1, & \text{if } x \leq 0 \text{ and } y \leq 0, \\ 0, & \text{otherwise.} \end{cases} \quad (15)$$

$$f_3(x, y, z) = (x + 0.6)^2 + (y - 0.2)^2 + (z + 0.1)^2.$$

$$f_4(x, y, z) = -\sin((x - 0.1) + 0.5\pi) \cos(y - 0.2) \cos(z - 0.2).$$

$$f_5(x, y, z) = \begin{cases} 1, & \text{if } x \leq 0 \text{ and } y \leq 0 \text{ and } z \leq 0, \\ 0, & \text{otherwise.} \end{cases} \quad (16)$$

Using the orthogonal basis for (13),  $\tilde{\mathbf{v}}$  is obtained, which is then used by Algorithm 2 to find the values of parameters  $c$  and  $\gamma$ . For functions that have an exact analytical minimum, calculating the error between the exact minimum and the optimum returned by the GD is straightforward. For the discontinuous functions such as  $f_2$  and  $f_5$ , an approximate *golden* minimum is calculated by projecting the function using polynomial order  $N = 8$  on a dense grid of points in  $\Omega$  and finding the least value of the projected polynomial. In all our test cases, GD converges to the exact (or golden) minimum within an acceptable tolerance (numerical 0 set to  $10^{-7}$ ). Therefore, the selection of the parameters is made based on the fewest iterations taken by GD (*niter*) to converge.

**Table 1**

Results for the 2D experiment on a quadrilateral domain to find the optimal ranges of the gradient descent line search parameters:  $c$  and  $\gamma$ .  $M$  denotes the polynomial order for projected functions (15). The number of iterations taken by GD is denoted by *niter*. The number of quadrature points for projection is constant ( $Q = 11$ )

M	$f_0(x, y)$			$f_1(x, y)$			$f_2(x, y)$		
	Range $c$	Range $\gamma$	niter	Range $c$	Range $\gamma$	niter	Range $c$	Range $\gamma$	niter
3	[0.4,1)	(0,1)	6	[0.7,1)	(0,1)	3	(0,1)	(0,1)	3
4	[0.5,1)	(0,1)	7	(0,1)	[0.3,1)	4	[0.2,1)	(0,1)	7
5	[0.5,1)	[0.3,1)	7	[0.6,1)	[0.2,1)	4	(0,1)	(0,1)	7
6	(0,1)	[0.5,1)	7	[0.3,1)	[0.5,0.8]	4	[0.4,0.9]	[0.7,1)	6
7	[0.2,1)	(0,1)	7	[0.3,1)	(0,1)	5	[0.5,1)	(0,1)	10
8	[0.7,1)	[0.5,1)	6	(0,1)	[0.7,1)	4	[0.3,1)	(0,1)	10
9	[0.7,1)	[0.7,1)	6	[0.5,1)	(0,0.2]	6	[0.2,0.6]	(0,1)	7
10	[0.5,0.7]	[0.5,1)	7	[0.5,1)	[0.3,1)	6	[0.5,1)	[0.8,1)	9

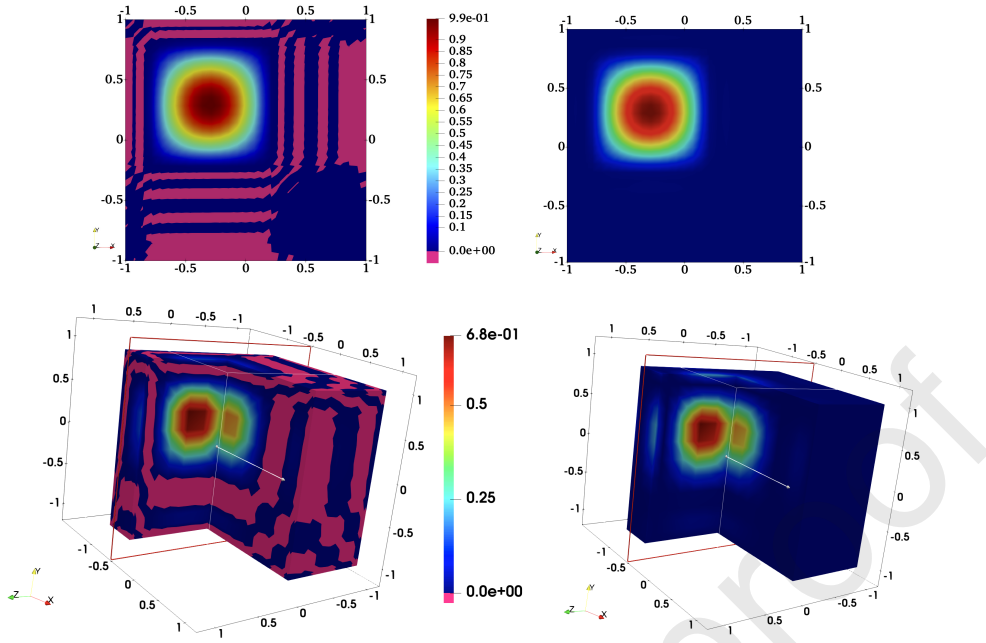
Consider  $k$  discrete equispaced samples of  $\in (0, 1)$ . For  $k = 9$ , by applying Algorithm 2, we get the results shown in Table 1.

Based on the results of these tests, we infer that for the numerical experiments, the appropriate choices for  $c$  and  $\gamma$  in 2D are 0.7 and 0.7, respectively. Following a similar procedure, Algorithm 2 on a hexahedron for functions defined in (16), the choices of  $c$  and  $\gamma$  for 3D are 0.2 and 0.7, respectively. These values are used for all the numerical experiments in the numerical results.

The C++ implementation of the filter in Nektar++ [32] supports change in  $c$  and  $\gamma$  as parameters in the configuration file.

## 5.2. Projection examples on 2D and 3D elements and application of the structure-preserving filter

Consider a 2D function (17) and a 3D function (18) that are both discontinuous clamped versions of a smooth sinusoidal function. The initial projections of  $f(x, y)$  on a quadrilateral and  $f(x, y, z)$  on a hexahedron are shown in Figure 4. The discontinuity produces oscillations similar to Gibb's phenomenon upon projection, which is interesting for the application and analysis of the filter.



**Figure 4:** Galerkin projection of  $f(x, y)$  (17) and  $f(x, y, z)$  (18) on a quadrilateral and hexahedron element, respectively. The 2D projection uses polynomial order = 7, and the 3D projection uses polynomial order = 5. *Left:* Unfiltered version showing areas where the desired structure (positivity) is lost. *Right:* After the filter is applied, the solution is non-negative up to tolerance.

$$f(x, y) = v(x, y) \sin(\pi(0.2 - x)) \sin(\pi(y + 0.2)), \quad (17)$$

where

$$v(x, y) = \begin{cases} 1, & \text{if } x \in [-0.8, 0.2] \text{ and } y \in [-0.2, 0.8], \\ 0, & \text{otherwise.} \end{cases}$$

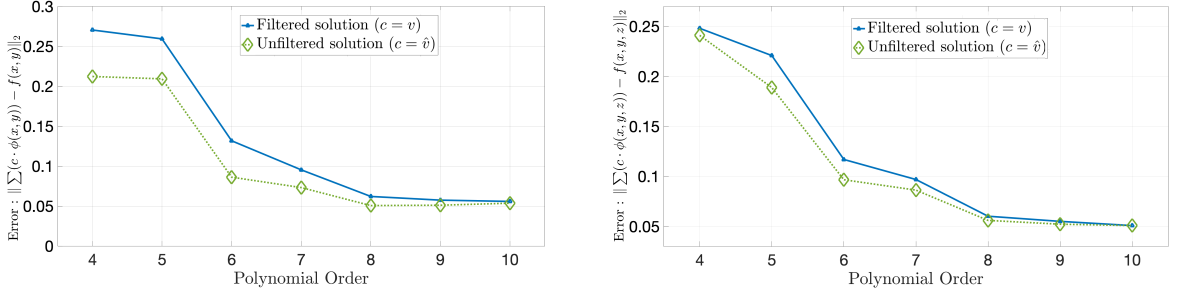
$$f(x, y, z) = v(x, y, z) \sin(\pi(0.2 - x)) \sin(\pi(y + 0.2)) \sin(\pi(z + 0.2)), \quad (18)$$

where

$$v(x, y, z) = \begin{cases} 1, & \text{if } x \in [-0.8, 0.2] \text{ and } y \in [-0.2, 0.8] \text{ and } z \in [-0.2, 0.8] \\ 0, & \text{otherwise.} \end{cases}$$

It is evident from the left subfigures in Figure 4 that the projection of discontinuous non-negative functions leads to the negative values as shown in the highlighted regions. As seen in the right subfigures of Figure 4, the application of the structure-preserving filter restores the non-negative structures of (17) and (18), respectively. To analyze the p-convergence, the experiment is repeated for different polynomial orders, as shown in Figure 5.

Figure 5 presents  $L_2$  error comparison between the filtered and unfiltered version of projections of nonsmooth functions in 2D and 3D. The choice of a nonsmooth function is to demonstrate the effect of filter application in worst-case scenarios.



**Figure 5:** A p-convergence study of the filtered and unfiltered Galerkin projections from Figure 4. *Left:*  $f(x, y)$  defined in (17) and *Right:*  $f(x, y, z)$  defined in (18)

### 5.3. Structure preservation in 2D and 3D advection problems

Consider a dG solution to the advection problem (19) on a 2D domain as shown in the first row of Figure 6.

$$u_t + \mathbf{a} \cdot \nabla u = 0, \quad (19)$$

Consider a function  $f$  defined on  $[-1, 1] \times [-1, 1]$  as (20).

$$f(x, y) = 1 - \cos\left(\frac{\pi x}{2}\right) \cos\left(\frac{\pi y}{2}\right) \quad (20)$$

For a given initial condition to be  $f(x, y)$ ,  $\mathbf{a} = [1, 1]$ , and the periodic boundary conditions, we formulate the problem in the discontinuous Galerkin framework. The first step is projecting  $f$  on a set of  $E$  elements  $\{e_0, e_1 \dots e_{E-1}\} \in \Omega$  using the typical nonorthogonal (hats and bubbles) basis  $\phi$ . Locally, for an element  $e$ , we have

$$f_e(x, y) = \sum_{i=0}^{P-1} \hat{u}_i \phi_i(x, y),$$

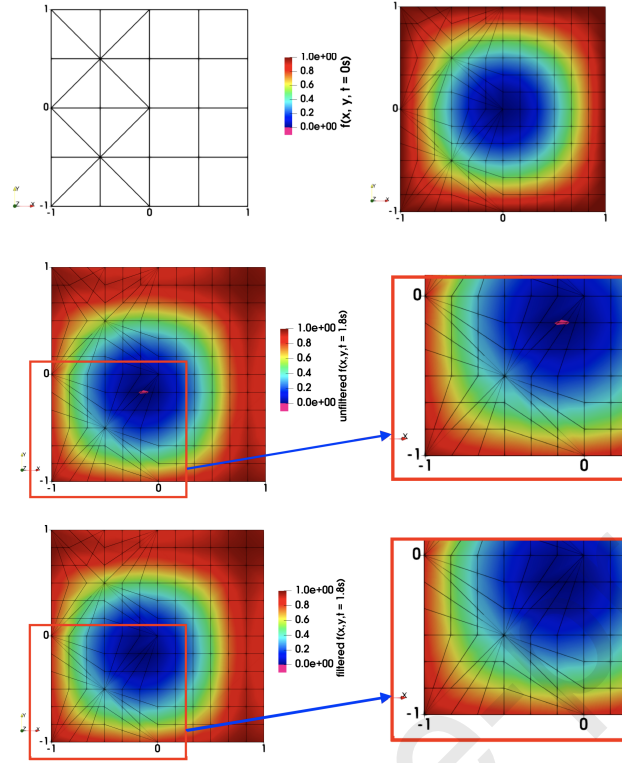
where  $\Omega$  is the mesh shown in Figure 6 consisting of a mix of quadrilaterals and triangles. The filter steps do not change for different element types as emphasized by the choice of composite mesh.

The solution for the advection problem using the timestep  $\Delta t = 1e - 3$ , polynomial order 4, RK-4 integration scheme, and upwind flux calculations is shown in Figure 6. Row 2 of the figure shows the simulation state at a particular timestep and highlights the negative values in the domain. In row 3 the non-negative structure is restored after the application of the filter. The filter is applied at each timestep to preserve the structure (positivity) of the solution on a lattice of points of interest. The lattice is a set of points defined in (1). If a structure violation is found at any point in the lattice, the parent element is flagged for filtering. Since the boundaries are periodic, the final state of the simulation looks similar to the initial state.

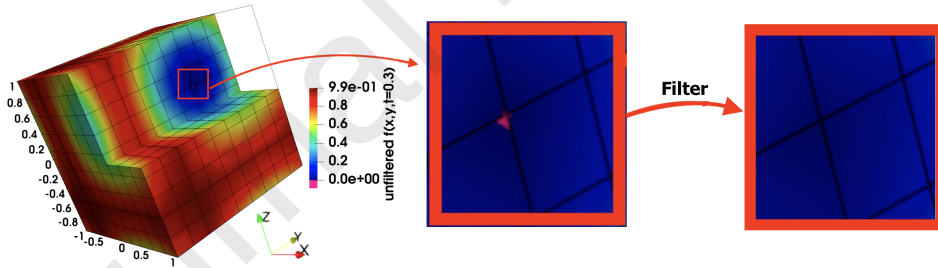
For a similar analysis of the 3D advection problem, consider the initial state as a smooth continuous function (21) on a cube mesh of 64 hexahedron elements defined in the domain  $[-1, -1, -1] \times [1, 1, 1]$ . The advection velocity is defined by  $\mathbf{a} = [1, 1, 1]$ , and all the boundary conditions are periodic. The integration method used is RK-4 with a timestep of  $1e-3$ . The timestepping is performed for a total of 2000 timesteps and the flux calculation is upwind. Following a similar selection procedure as in 2D, we discover the elements that violate the structure at each timestep and apply the structure-preserving filter to those elements.

$$f(x, y, z) = 1 - \cos\left(\frac{\pi x}{2}\right) \cos\left(\frac{\pi y}{2}\right) \cos\left(\frac{\pi z}{2}\right) \quad (21)$$

Figure 7 shows an instance during the advection process where a loss of structure is encountered, i.e., negative intermediate values are found, as shown by the highlighted region in the left and middle subfigures of Figure 7. The



**Figure 6:** Row 1, Left: A 2D composite mesh used for solving (19). The mesh contains 16 triangles and 8 quadrilaterals. Row 1, Right: Initial state of the system, which is a projection of (20) on the mesh. Row 2: Unfiltered solution at timestep 1800. Row 3: Filtered solution at timestep 1800.

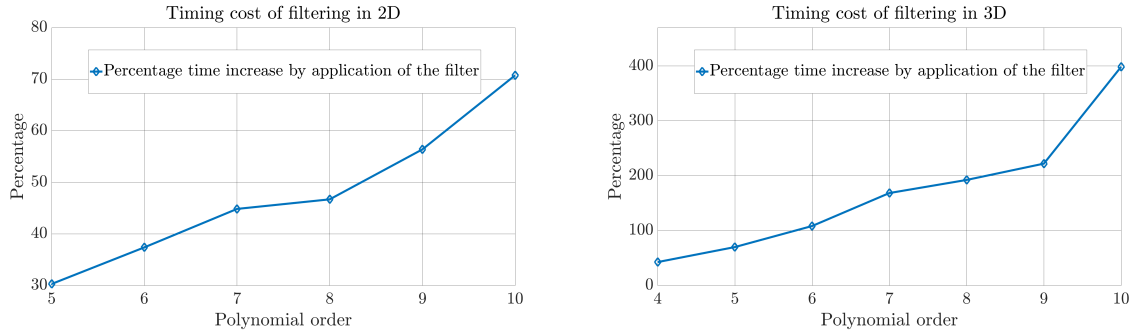


**Figure 7:** The state of the simulation at  $t = 0.3$  seconds (timestep = 300) using initial condition Equation (21). Left and middle: The highlighted region and its zoomed counterpart show the points in the elements that lose the positivity structure. Right: The filtered values at the same elements that preserve the positivity structure.

process of discovering the structurally nonconformal elements and application of the filter to those elements adds to the total time cost of the advection solution, which is an important aspect to consider when deciding on the criteria for applying the filter.

The percentage increase in total time taken by the experiment by the application of structure-preserving filter is shown in Figure 8. Note that for 2D, at low orders ( $N=2,3,4$ ), the cost of filtering is between 50-70% of the total time. This is due, in part, to the efficiency at these orders of the unfiltered solver such as caching effects, as well as up-front costs associated with the optimization. We find that for 2D the ratio of filtered to unfiltered is lowest at  $N=5$ , and then



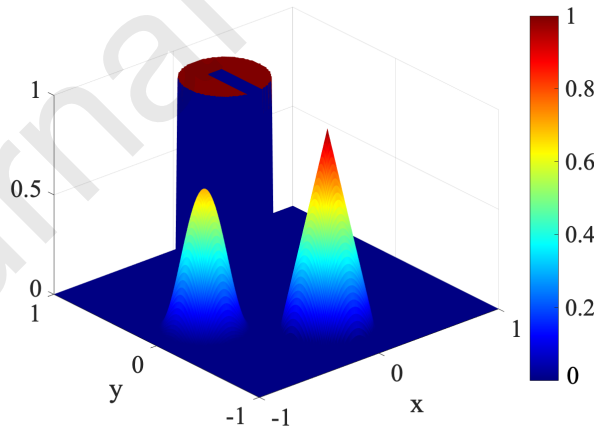


**Figure 8:** Percentage increase in time to complete the experiment by application of the filter. *Left: 2D Right: 3D*

proceeds to climb linearly as shown in the left subfigure of Figure 8. The time taken by the filter in 3D is notably higher than the time taken by the filter in 2D because of the higher complexity of finding the global minimum in 3D. The cost of overall filtering per timestep depends on the number of elements that the filter operates upon. Therefore, the procedure of selective application of the filter becomes increasingly important.

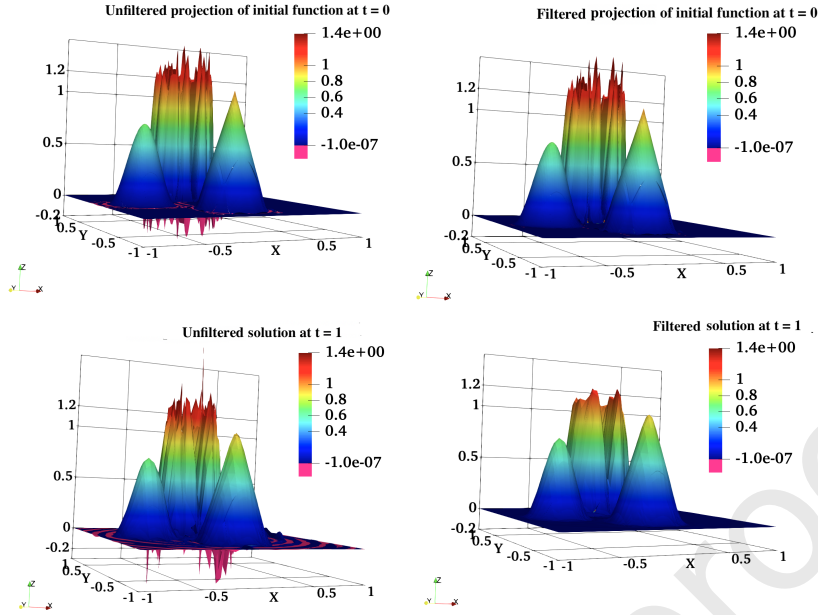
#### 5.4. Structure preservation of the canonical rotating solid body test

We now investigate the application of the filter on the solid body rotation experiment using a discontinuous initial condition defined as a combination of a notched cylinder, a cone, and a smooth hump. Consider the initial data shown in Figure 9. This example is extensively used in advection and structure preservation literature [33]. The parameters to reproduce the initial state: the cylinder, cone, and hump have a radius of 0.3 each. The centers of the cylinder, cone, and hump are  $(0, 0.5)$ ,  $(0, -0.5)$ , and  $(-0.6, 0)$  respectively. Without changing the domain and boundary conditions used for the 2D advection experiment shown in Figure 6, the advection velocity is changed to circular such that in one time period, the solution returns to its original state. For all tests in this section, the domain has 144 quadrilateral elements and the timestep  $(\Delta t)$  is  $5e - 4$ .

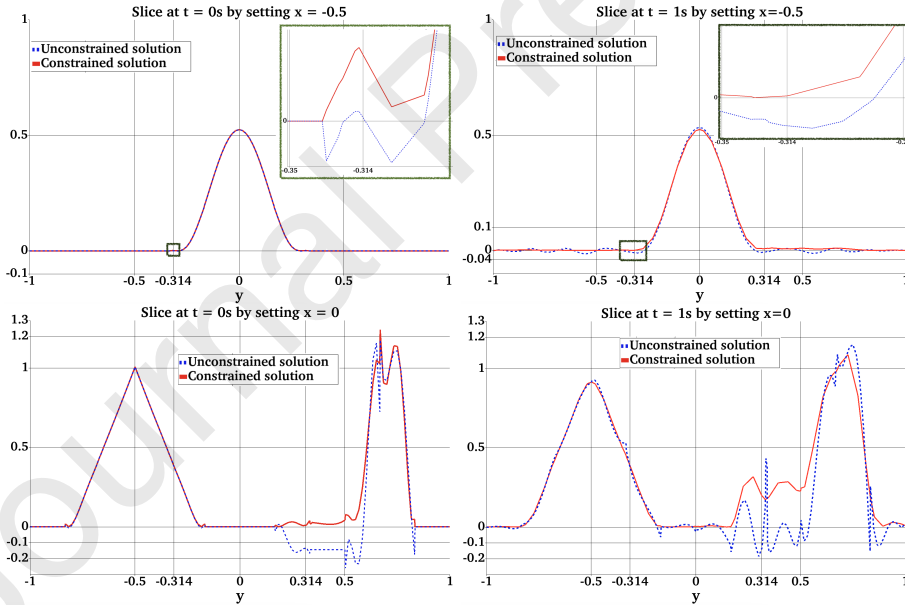


**Figure 9:** Initial state for solid body rotation tests.

Since the initial condition for the experiment is defined by a complicated discontinuous function, the solution suffers from a larger loss of structure as compared to the case with a smooth initial condition. Figure 10 shows the initial and final states during rotation of the body shown in Figure 9, with and without the filter application.

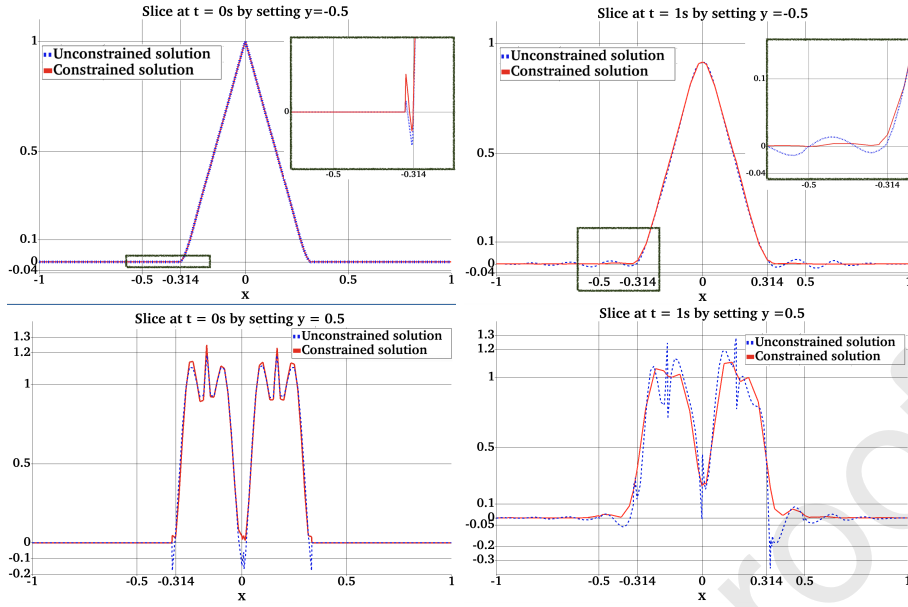


**Figure 10:** Snapshots at the beginning and end of advection. The highlighted region shows the values below tolerance for negativity ( $10^{-7}$ ). At  $t = 1$  s, the solid body finishes one rotation and returns to the original position. Parameters for the test: Timestep =  $5e - 4$ , polynomial order = 5. Row 1: Initial function projection at time = 0s. Row 2: Solution at time = 1s. Column 1: Unconstrained. Column 2: Constrained.

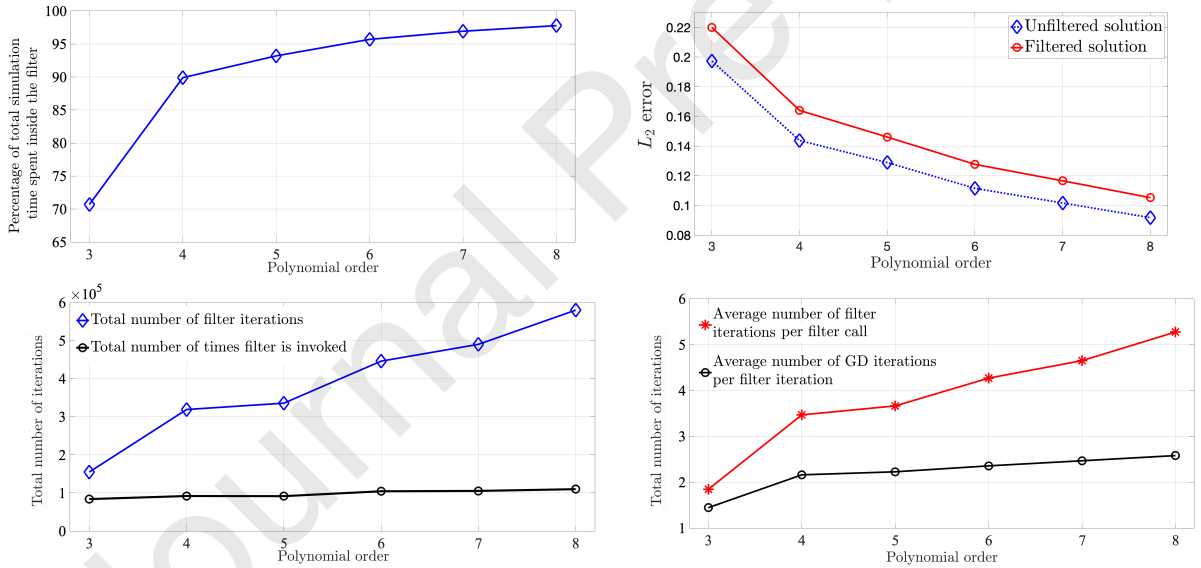


**Figure 11.a:** Snapshots at various points in time during advection.  $\Delta t = 5e - 4$ , time period = 1 and polynomial order = 5. Some subfigures show insets with the zoomed-in area of interest. Column 1: State of the system at  $t = 0$ . Column 2: State of the system at  $t = 1$  s. Row 1: Slice at  $x = -0.5$  and Row 2: Slice at  $x = 0$ .

Figures 11.a and 11.b show different cross sections of the solution with and without filtering at the initial and final time. The total time taken by the filter as a function of the entire simulation time is shown in Figure 12.



**Figure 11.b:** Snapshots at various points in time during advection.  $\Delta t = 5e - 4$ , time period = 1 and polynomial order = 5. Some subfigures show insets with the zoomed-in area of interest. *Column 1:* State of the system at  $t = 0$ . *Column 2:* State of the system at  $t = 1$ . *Row 1:* Slice at  $y = -0.5$  and *Row 2:* Slice at  $y = 0.5$



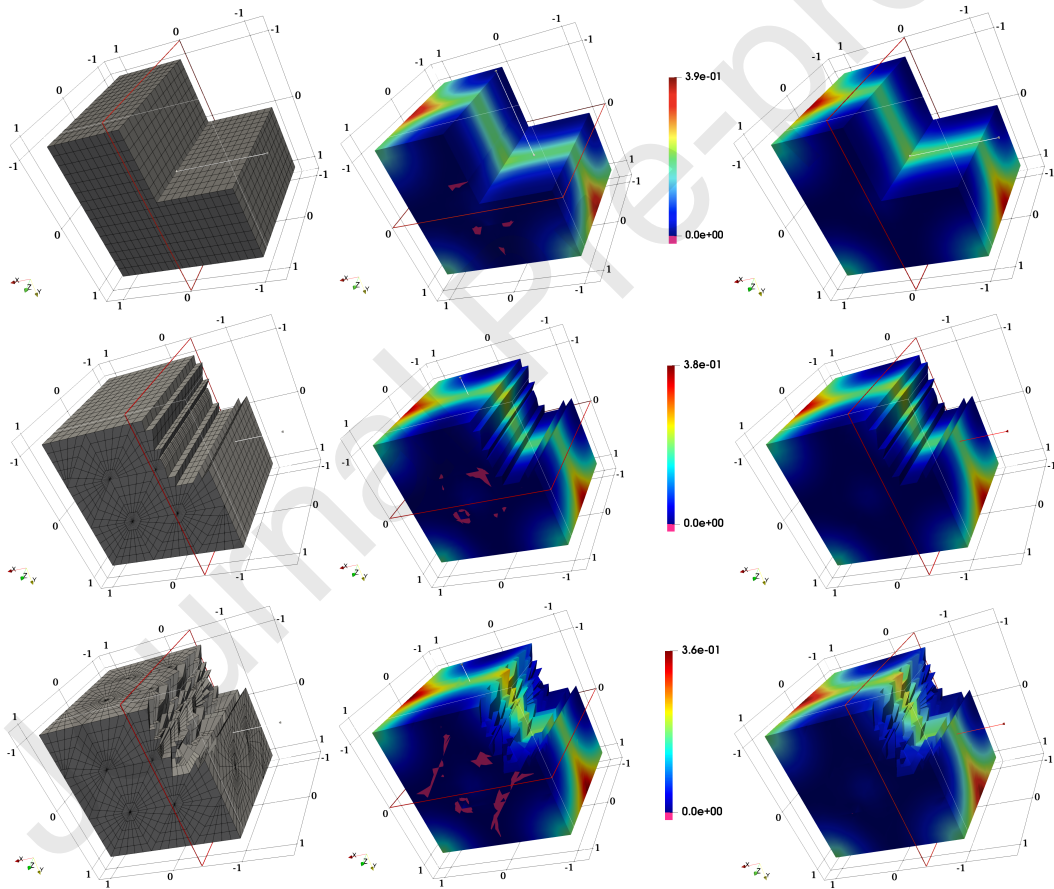
**Figure 12:** Analysis of filter application to the experiment shown in Figure 9. *Top left:* Percentage of total simulation time spent inside the filter, *Top right:*  $p$ -convergence of  $L_2$  errors for filtered and unfiltered solution at  $t=1s$ . *Bottom left:* Total number of filter iterations vs the total number of times the filter is called for various polynomial orders. *Bottom right:* Average number of filter iterations per call vs the average number of GD line-search algorithm per filter iteration for various polynomial orders.

As evident from the top left panel in Figure 12, the filter takes proportionally longer to restore the structure as the loss of structure increases. In many practical applications, such as the one presented in Section 5.6, we observe that paying the extra filtering cost results in the successful termination of the simulation, which otherwise fails due to the presence of structural inconsistencies. In such cases, the tradeoff between the time cost vs guarantee of structure

preservation leans towards the latter. The filtered and unfiltered  $L_2$  errors after one rotation of the solid body are shown in the top right panel of Figure 12. Whereas filtering changes the final state of the solution, thus contributing to the errors, the difference between filtered and unfiltered errors is still comparable. The filtered version of the tests using different polynomial order follows the same p-convergence as the unfiltered counterpart. A comparison between the number of times the filter is invoked and the number of iterations taken per invocation is shown in the bottom left panel of Figure 12. We notice that the number of iterations taken per call to the filter increases with the increase in polynomial order because of the larger magnitude of structural inconsistencies observed as a result of the oscillations in the solution. The largest contributor to the cost of applying the filter occurs from the GD line-search to find the global minimum. At each iteration of the filter, we find the minimum. Each call to find the minimum, in turn, takes a few iterations of GD. On the bottom right panel of Figure 12 is a comparison between the average number of GD iterations per iteration of the filter. Also shown is the average number of filter iterations per call to the filter.

### 5.5. Structure preservation in advection tests on various 3D domains

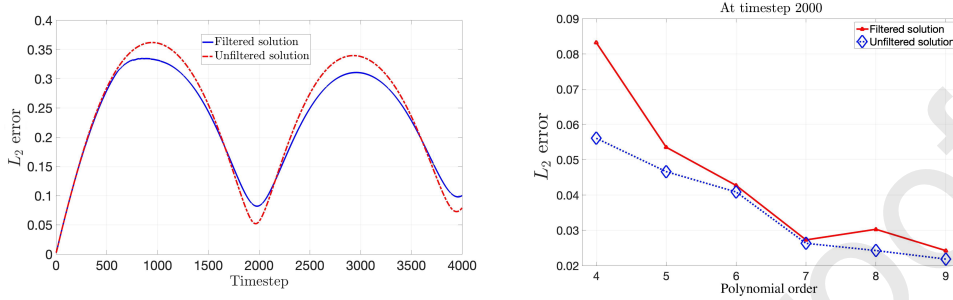
The procedure to filter the elements that exhibit loss of structure remains agnostic to the type of element. To demonstrate this, consider different tessellations of the domain  $[-1, -1, -1] \times [1, 1, 1]$  and solve an advection problem with the nontrivial initial state (22). The setup remains the same as the previous 3D example, i.e.,  $\mathbf{a} = [1, 1, 1]$  in all directions, periodic boundaries, RK-4 integration scheme using timestep  $1e-3$ , and for a total of 2000 timesteps.



**Figure 13:** A summary of the advection experiment using polynomial order = 4 and  $\Delta t = 1e-3$  on the 3D cube domains meshed using different canonical elements individually. *Left to right:* The mesh profile, the unfiltered solution, and the filtered solution at timestep = 1000, respectively. *Top row:* Cube mesh with 27 hexahedra, *Middle row:* Cube mesh with 54 prisms, *Bottom row:* Cube mesh with 162 tetrahedra.

$$f(x, y, z, t = 0) = 0.2 \left( [1 - \sqrt{(x^2 + y^2)}]^2 + z^2 \right) \quad (22)$$

The initial state looks like a torus inside a cube. The advection experiment is repeated on different homogeneous meshes with element types hexahedra, tetrahedra, and prisms individually. Figure 13 shows the results before and after the filter application at a particular timestep for all these meshes.



**Figure 14:** Left: Per timestep  $L_2$  error for advection problem for polynomial order= 4 on a cube mesh of 24 tetrahedral elements. The timestep =  $1e-3$ , and the total steps = 4000. At each timestep, for the  $L_2$  error computation, we consider 10 quadrature points in each direction. Right: p-convergence for filtered and unfiltered  $L_2$  error at timestep 2000 for the advection problem on a cube mesh of 24 tetrahedral elements.

The analysis of the filtering effects on the accuracy and convergence of the advection process, especially on the meshes with hexahedron and prism elements, reveals that the difference between the  $L_2$  errors is too small to warrant a convergence comparison. However, for the mesh with tetrahedral elements, the  $L_2$  errors perceptibly vary at each timestep, as shown by a longer run of the same experiment (total 4000 timesteps) in Figure 14.

## 5.6. Filter application to dG FEM implementation of the PAC model

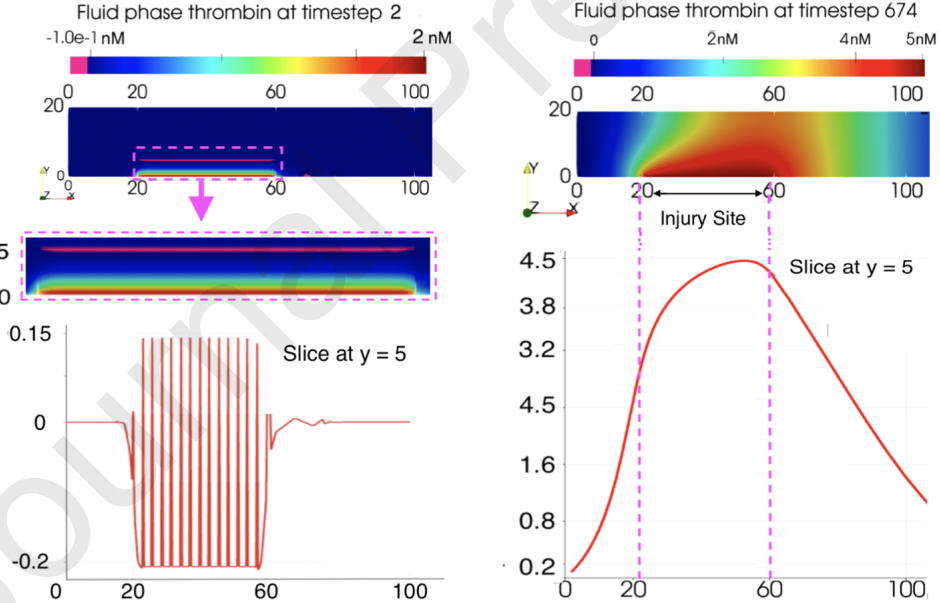
A prominent scenario in which the problem of structure-preservation is encountered is in the mathematical-biology domain. A detailed mathematical model for platelet aggregation and blood coagulation (PAC) by [34] is considered for this experiment. The model describes the process of evolution, interaction, and decay of the chemical species involved in the process of thrombosis.

Although the details of the individual species in the PAC process are beyond the scope of this paper, [34] has a detailed explanation of the nature and the evolution of the chemical species. The model can be summarized as a comprehensive collection of ODEs and PDEs that track all the chemical species in various phases during the process. The results in [34] use a finite-difference method with a specified limiter to truncate nonpositive concentration values to zero. In order to implement this model using the FEM without loss of structure, an alternative approach to truncation, such as the one presented in this paper, is needed. For our experiments, we use the structure-preserving filter instead of a truncation limiter. The focus is limited to the system of equations to track the chemical species that advect, diffuse, and react. One such species is fluid-phase thrombin (FP  $e_2$ ), an essential component of the thrombosis process. For this section, we track the evolution of thrombin by advection, diffusion, and decay in the fluid-phase. For the detailed results of the evolution of all the chemical species, including thrombin, under various circumstances in the original model, refer to [34].

We set up a version of the PAC model that solves the species evolution problem by a combination of advection, diffusion, and reaction (ADR) PDEs. Unlike the original model, the velocity  $\mathbf{u}$  of the fluid medium is not reported by a modified Navier-Stokes solver. Instead, the velocity is constant  $\mathbf{u} = [u_x, u_y] = [5, 0]$ . To solve this system of PDEs on a sample blood vessel represented by a rectangular domain  $\Omega = [0, 300] \times [0, 20]$ , we employ the dG FEM. The domain is tessellated using 2048 quadrilaterals. The bottom wall has an injury site spanning from  $x = 20$  to  $x = 60$ . We use polynomial order = 4 and timestep of  $\Delta t = 1e-2$  to solve the problem.

The total number of species tracked in our implementation of the model is 56. Since the focus is on the behavior of thrombin, we present the PDE describing the evolution of fluid-phase thrombin (FP  $e_2$ ) (23). For details on the other chemical species involved in (23), refer to [34]. The boundary conditions vary depending on the chemical being tracked. For FP  $e_2$ , we have a Dirichlet zero boundary on the left, and a no-flux boundary on the right and top. The bottom boundary has two kinds of conditions: a robin boundary condition on the injury site and Dirichlet zero everywhere else.

$$\begin{aligned}
 \frac{\partial e_2}{\partial t} = & \underbrace{-\mathbf{u} \cdot \nabla e_2 + \nabla \cdot (D \nabla e_2)}_{\text{Transport by Advection Diffusion}} \\
 & + \underbrace{k_{e_2}^{on} e_2 (N_2 P^{b,a} + N_2 P^{se,a} - z_2^{mtot} - e_2^{mtot}) + k_{e_2}^{off} e_2^m}_{\text{Binding to platelet receptor}} \\
 & + \underbrace{(k_{z_5:e_2}^{cat} + k_{z_5:e_2}^-)[Z_5 : E_2] - k_{z_5:e_2}^+ z_5 e_2}_{\text{Activation of V}} \\
 & + \underbrace{(k_{z_7:e_2}^{cat} + k_{z_7:e_2}^-)[Z_7 : E_2] - k_{z_7:e_2}^+ z_7 e_2}_{\text{Activation of VII}} \\
 & + \underbrace{(k_{z_8:e_2}^{cat} + k_{z_8:e_2}^-)[Z_8 : E_2] - k_{z_8:e_2}^+ z_8 e_2}_{\text{Activation of VIII}}
 \end{aligned} \tag{23}$$



**Figure 15:** *Top left:* The unconstrained simulation solution becomes invalid, as shown in the inset at **timestep 2**, which causes a blowup at timestep 87, leading to the failure of the simulation. *Top right:* The constrained simulation runs for **674 timesteps** until the depletion of the chemicals causes the model to terminate naturally. *Bottom left:* Profile of fluid phase thrombin in unconstrained simulation at timestep 2 on a domain slice at  $y = 5$ . *Bottom right:* Profile of fluid phase thrombin in constrained simulation at timestep 674 on a domain slice  $y = 5$

Figure 15 shows filtered and unfiltered runs of our implementation of this model and the resulting concentration profile of FP  $e_2$ . At timestep = 2, we get the invalid (negative) values around the injury site because of the sharp



concentration changes (refer to the slice at  $y = 5$  in the bottom left part of Figure 15). Therefore, the fluid phase thrombin concentration does not adhere to the desired structure (positivity), and the simulation does not succeed. If this experiment is allowed to continue, the effect of invalid values will compound over time, rendering the simulation results nonphysical and useless. At timestep 38, the simulation starts becoming unstable because of structural discrepancies. If it is allowed to run further, the accumulation and propagation of negative values results in the simulation blow-up at timestep 87, at which point the code reports invalid values such as *NaN* (datatype: Not a Number).

When the filter is applied, it runs for significantly more timesteps (total 674) and terminates naturally. The right side of Figure 15 shows the state of FP  $e_2$  at the depletion point of the chemicals involved.

## 6. Conclusions

We present a formalism that solves the problem of structure-preservation for PDE solutions in 2D and 3D. The construction and design of a postprocessing structure-preserving filter are detailed and applied to multidimensional dG FEM solutions to different PDEs. A geometric interpretation of the mathematical foundation behind the filter is presented, followed by an algorithm to apply the proposed filter in a timestepping PDE framework. At the core of the filter lies the expensive requirement for global minimization on a weighted distance function that corresponds to the objective we optimize. We employ gradient descent with backtracking line search for the minimization to reduce the cost of the filtering routine. To this end, we detail an investigative procedure to reduce the minimization cost by precomputing specific parameter values for the gradient descent approach. Using numerical examples, we compare the convergence rates with and without the application of the filter for different problem sizes and domain structures. The percentage increase in time taken by the simulation is computed to understand the cost of the filter and it is observed that the cost scales with the order and size of the problem. In the end, using a filtered solution to a mathematical-biology problem of platelet aggregation and blood coagulation, we provide evidence of the proposed method's efficacy and utility. One future direction for investigation could attempt to understand when inter-element flux preservation is beneficial in these types of numerical simulations. For example, comparing two filtered solutions, one with and one without flux preservation, could form the basis for more experiments.

## Acknowledgments

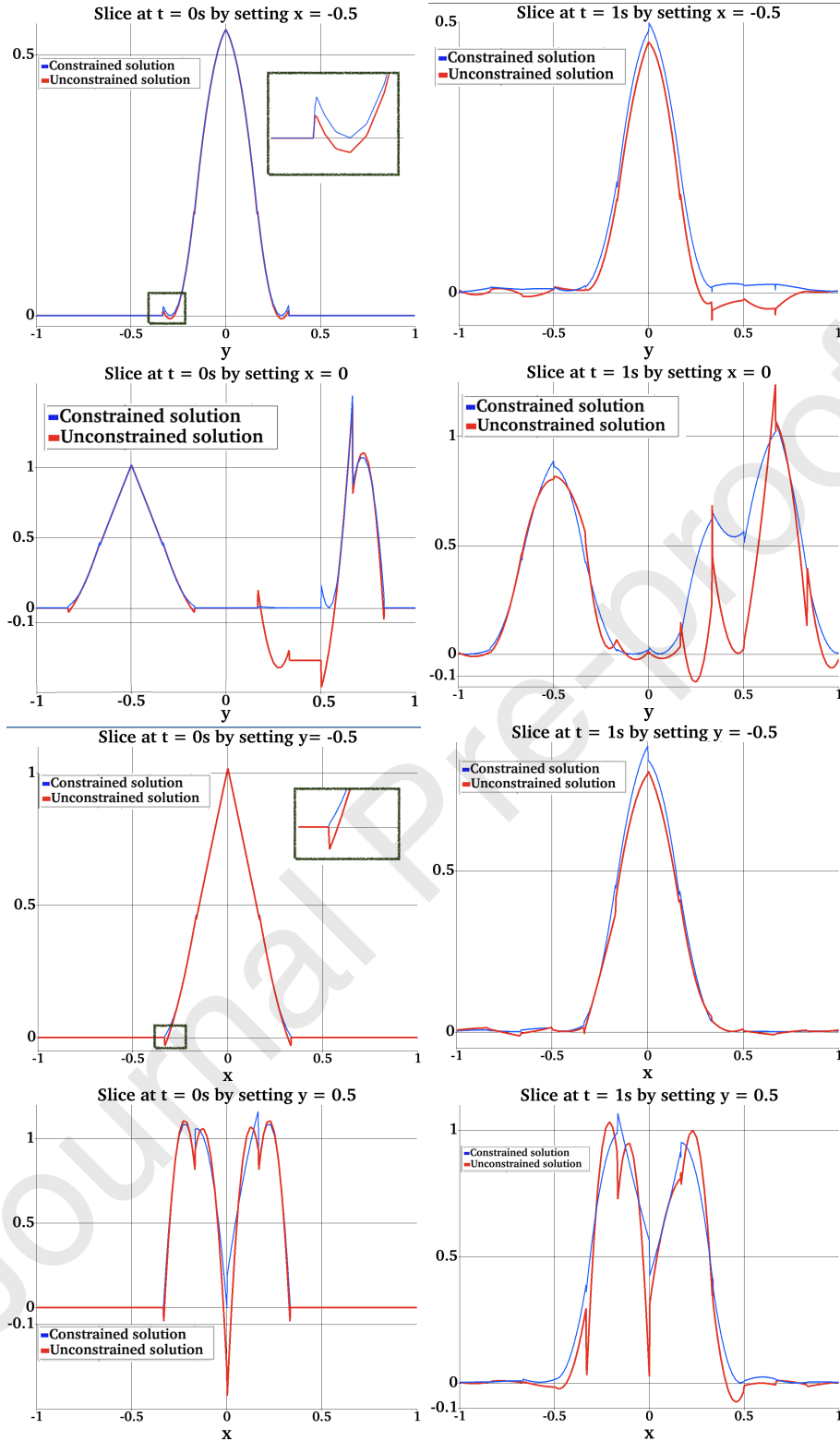
V. Zala and R.M. Kirby acknowledge that their part of this research was sponsored by ARL under cooperative agreement number W911NF-12-2-0023. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of ARL or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. A. Narayan was partially supported by NSF DMS-1848508. This material is based upon work supported by both the National Science Foundation under Grant No. DMS-1439786 and the Simons Foundation Institute Grant Award ID 507536 while A. Narayan was in residence at the Institute for Computational and Experimental Research in Mathematics in Providence, RI, during the Spring 2020 semester.

## References

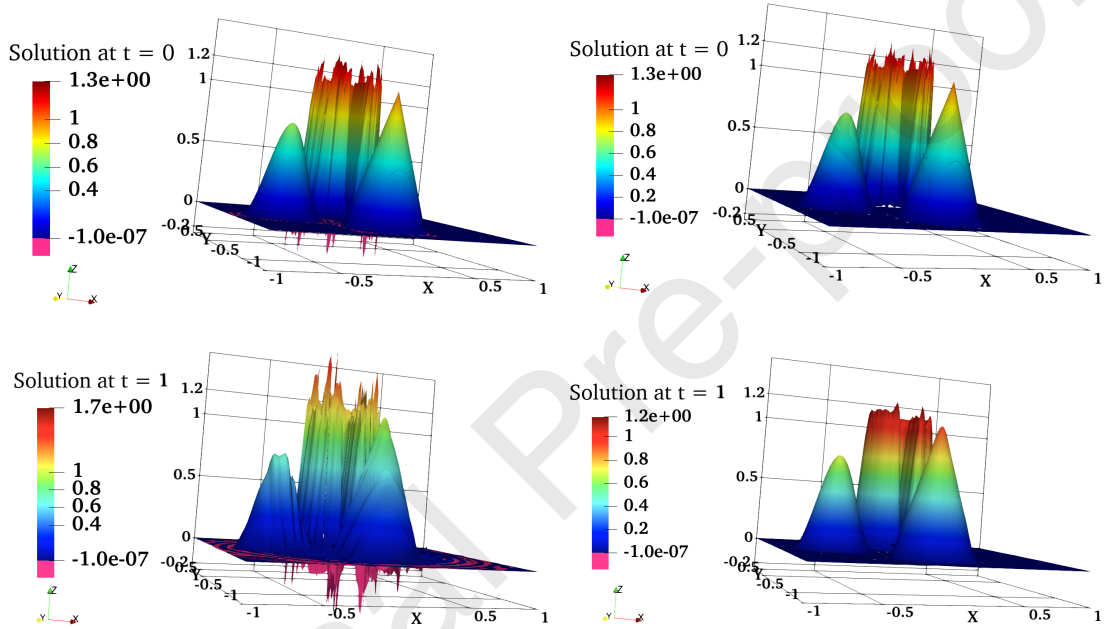
1. Zahr, MJ, Persson, PO. An optimization based discontinuous galerkin approach for high-order accurate shock tracking. In: *2018 AIAA Aerospace Sciences Meeting*. 2018:0063.
2. van der Vegt, J, Xia, Y, Xu, Y. Positivity preserving limiters for time-implicit higher order accurate discontinuous galerkin discretizations. *arXiv preprint arXiv:181108620* 2018.
3. Liu, C, Wang, C, Wang, Y. A structure-preserving, operator splitting scheme for reaction-diffusion equations with detailed balance. *Journal of Computational Physics* 2021 436:110253. URL: <https://www.sciencedirect.com/science/article/pii/S0021999121001480>. doi:<https://doi.org/10.1016/j.jcp.2021.110253>.
4. Anderson, RW, Dobrev, VA, Kolev, TV, Rieben, RN. Monotonicity in high-order curvilinear finite element arbitrary lagrangian-eulerian remap. *International Journal for Numerical Methods in Fluids* 2015 77(5):249–273.
5. Anderson, R, Dobrev, V, Kolev, T, Kuzmin, D, de Luna, MQ, Rieben, R, Tomov, V. High-order local maximum principle preserving (mpp) discontinuous galerkin finite element method for the transport equation. *Journal of Computational Physics* 2017 334:102–124.
6. Allen, L, Kirby, RC. Bounds-constrained polynomial approximation using the Bernstein basis. *arXiv:210411819 [cs, math]* 2021 URL: <http://arxiv.org/abs/2104.11819> arXiv: 2104.11819.
7. Lohmann, C, Kuzmin, D, Shadid, JN, Mabuza, S. Flux-corrected transport algorithms for continuous galerkin methods based on high order bernstein finite elements. *Journal of Computational Physics* 2017 344:151–186.

8. Anderson, RW, Dobrev, VA, Kolev, TV, Rieben, RN, Tomov, VZ. High-order multi-material ale hydrodynamics. *SIAM Journal on Scientific Computing* 2018 40(1):B32–B58. URL: <https://doi.org/10.1137/17M1116453>. doi:10.1137/17M1116453. arXiv:<https://doi.org/10.1137/17M1116453>.
9. Light, D, Durran, D. Preserving nonnegativity in discontinuous galerkin approximations to scalar transport via truncation and mass aware rescaling (tmar). *Monthly Weather Review* 2016 144(12):4771–4786.
10. Zhang, X, Shu, CW. On maximum-principle-satisfying high order schemes for scalar conservation laws. *Journal of Computational Physics* 2010 229(9):3091 – 3120.
11. Sanders, R. A third-order accurate variation nonexpansive difference scheme for single nonlinear conservation laws. *Mathematics of Computation* 1988 51(184):535–558.
12. Liu, XD, Osher, S. Nonoscillatory high order accurate self-similar maximum principle satisfying shock capturing schemes i. *SIAM Journal on Numerical Analysis* 1996 33(2):760–779.
13. Shu, CW, Osher, S. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of computational physics* 1988 77(2):439–471.
14. Zhang, X. On positivity-preserving high order discontinuous galerkin schemes for compressible navier–stokes equations. *Journal of Computational Physics* 2017 328:301–343.
15. Laiu, MP, Hauck, CD, McClarren, RG, O’Leary, DP, Tits, AL. Positive Filtered P<sub>S</sub>–n<sub>S</sub> Moment Closures for Linear Kinetic Equations. *SIAM Journal on Numerical Analysis* 2016 54(6):3214–3238. doi:10.1137/15M1052871.
16. Dela Haije, T, Özarslan, E, Feragen, A. Enforcing necessary non-negativity constraints for common diffusion mri models using sum of squares programming. *NeuroImage* 2020 209:116405. URL: <https://www.sciencedirect.com/science/article/pii/S1053811919309966>. doi:<https://doi.org/10.1016/j.neuroimage.2019.116405>.
17. Van Benthem, MH, Keenan, MR. Fast algorithm for the solution of large-scale non-negativity-constrained least squares problems. *Journal of Chemometrics: A Journal of the Chemometrics Society* 2004 18(10):441–450.
18. Goh, A, Lenglet, C, Thompson, PM, Vidal, R. A nonparametric riemannian framework for processing high angular resolution diffusion images and its applications to odf-based morphometry. *NeuroImage* 2011 56(3):1181–1201.
19. Cheng, J, Deriche, R, Jiang, T, Shen, D, Yap, PT. Non-negative spherical deconvolution (nnsd) for estimation of fiber orientation distribution function in single-/multi-shell diffusion mri. *NeuroImage* 2014 101:750–764.
20. Zala, V, Kirby, RM, Narayan, A. Structure-preserving function approximation via convex optimization. *SIAM Journal on Scientific Computing* 2020 42(5):A3006–A3029.
21. Zala, V, Kirby, RM, Narayan, A. Structure-preserving nonlinear filtering for continuous and discontinuous galerkin spectral/hp element methods. *SIAM Journal on Scientific Computing (Under review)* 2020 .
22. Kucharik, M, Shashkov, M, Wendroff, B. An efficient linearity-and-bound-preserving remapping method. *Journal of computational physics* 2003 188(2):462–471.
23. Shashkov, M, Wendroff, B. The repair paradigm and application to conservation laws. *Journal of Computational Physics* 2004 198(1):265–277.
24. Berrut, JP, Trefethen, LN. Barycentric lagrange interpolation. *SIAM Review* 2004 46:501–517.
25. Loughton, E, Zala, V, Narayan, A, Kirby, RM, Moxey, D. Fast barycentric-based evaluation over spectral/hp elements. *Accepted by SIAM Journal on Scientific Computing* 2021 arXiv:2103.03594.
26. Motzkin, TS, Schoenberg, IJ. The Relaxation Method for Linear Inequalities. *Canadian Journal of Mathematics* 1954 6:393–404. URL: <https://www.cambridge.org/core/journals/canadian-journal-of-mathematics/article/relaxation-method-for-linear-inequalities/8EDDF8B5F08B146AC71A5CF4F304A560>. doi:10.4153/CJM-1954-038-x publisher: Cambridge University Press.
27. Armijo, L. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics* 1966 16(1):1 – 3. URL: <https://doi.org/10.2307/23095080>.
28. Crockett, JB, Chernoff, H, et al. Gradient methods of maximization. *Pacific Journal of Mathematics* 1955 5(1):33–50.
29. Curry, HB. The method of steepest descent for non-linear minimization problems. *Quarterly of Applied Mathematics* 1944 2(3):258–261.
30. Bertsekas, DP. Nonlinear programming. *Journal of the Operational Research Society* 1997 48(3):334–334.
31. Boyd, S, Boyd, SP, Vandenberghe, L. Convex optimization. Cambridge university press 2004.
32. Cantwell, CD, Moxey, D, Comerford, A, Bolis, A, Rocco, G, Mengaldo, G, De Grazia, D, Yakovlev, S, Lombard, JE, Ekelschot, D, et al. Nektar++: An open-source spectral/hp element framework. *Computer physics communications* 2015 192:205–219.
33. Leveque, RJ. High-resolution conservative algorithms for advection in incompressible flow. *SIAM Journal on Numerical Analysis* 1996 33(2):627–665.
34. Leiderman, K, Fogelson, AL. Grow with the flow: a spatial–temporal model of platelet deposition and blood coagulation under flow. *Mathematical Medicine and Biology: a journal of the IMA* 2011 28(1):47–84.

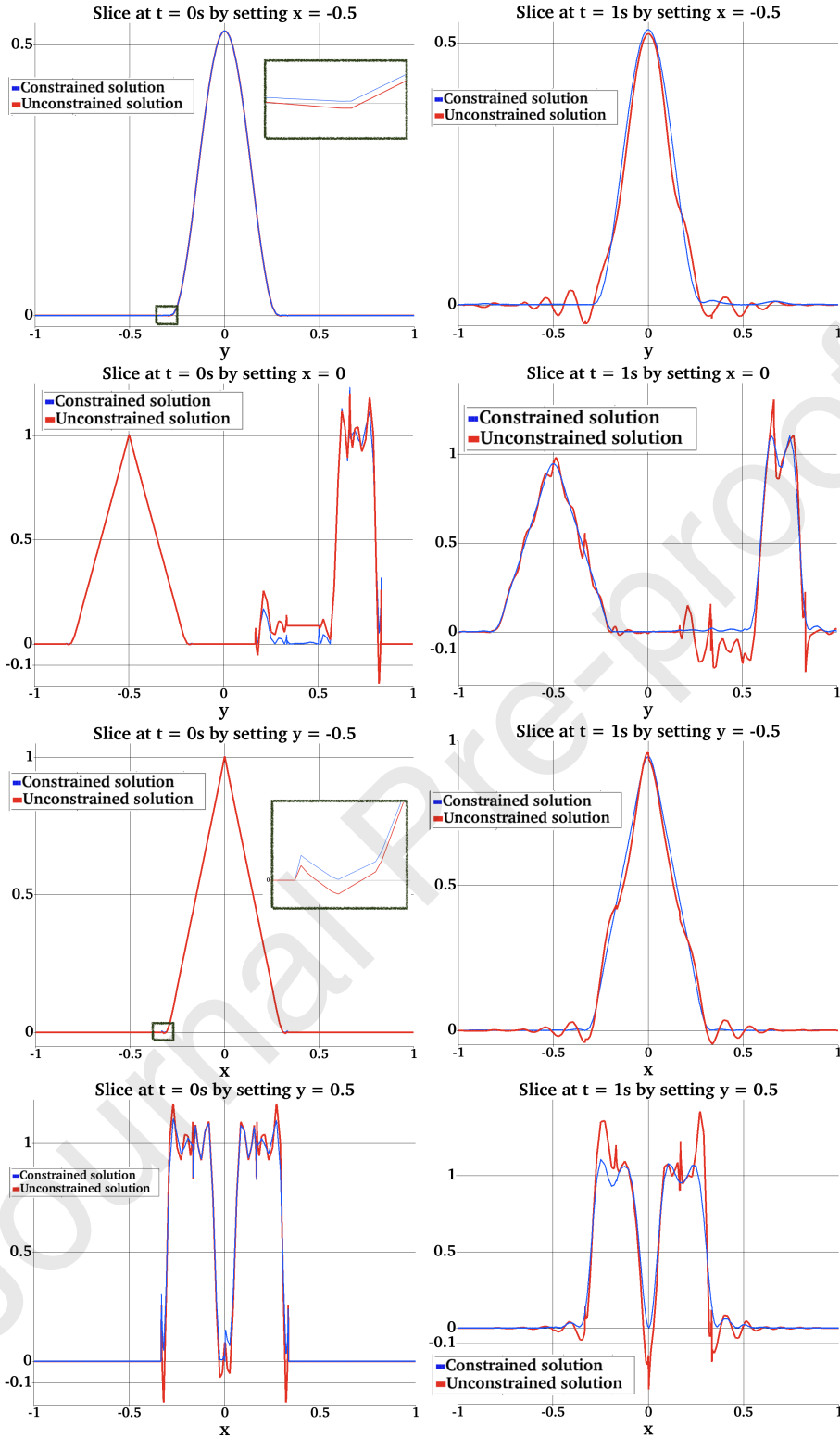




**Figure 11.a:** Snapshots at various points in time during advection.  $\Delta t = 5e - 4$ , time period = 1 and polynomial order = 3. Some subfigures show insets with the zoomed-in area of interest. *Column 1:* State of the system at  $t = 0$ . *Column 2:* State of the system at  $t = 1$  s. *Row 1:* Slice at  $x = -0.5$ . *Row 2:* Slice at  $x = 0$ . *Row 3:* Slice at  $y = -0.5$  and *Row 4:* Slice at  $y = 0.5$



**Figure 18:** Snapshots at the beginning and end of advection. The highlighted region shows the values below tolerance for negativity ( $10^{-7}$ ). At  $t = 1s$ , the solid body finishes one rotation and returns to the original position. Parameters for the test: Timestep =  $5e-4$ , polynomial order = 7. Row 1: Time = 0s. Row 2: Time = 1s. Column 1: Unconstrained solution. Column 2: Constrained solution.



**Figure 11.a:** Snapshots at various points in time during advection.  $\Delta t = 5e - 4$ , time period = 1 and polynomial order = 7. Some subfigures show insets with the zoomed-in area of interest. *Column 1:* State of the system at  $t = 0$ . *Column 2:* State of the system at  $t = 1$ . *Row 1:* Slice at  $x = -0.5$ . *Row 2:* Slice at  $x = 0$ . *Row 3:* Slice at  $y = -0.5$  and *Row 4:* Slice at  $y = 0.5$



**Declaration of interests**

✓ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

## CRediT author statement

**Vidhi Zala:** Formalism, Methodology, Algorithm and Software development, Data curation, Writing- Original draft preparation, Visualization, Editing.

**Akil Narayan:** Teaching, Supervision, Verification, Review, Funding

**Mike Kirby :** Teaching, Supervision, Review, Funding