SIAM J. NUMER. ANAL.
Vol. 49, No. 5, pp. 1899–1920

SMOOTHNESS-INCREASING ACCURACY-CONSERVING (SIAC) POSTPROCESSING FOR DISCONTINUOUS GALERKIN SOLUTIONS OVER STRUCTURED TRIANGULAR MESHES*

HANIEH MIRZAEE†, LIANGYUE JI‡, JENNIFER K. RYAN§, AND ROBERT M. KIRBY†

Abstract. Theoretically and computationally, it is possible to demonstrate that the order of accuracy of a discontinuous Galerkin (DG) solution for linear hyperbolic equations can be improved from order $k+1$ to $2k+1$ through the use of smoothness-increasing accuracy-conserving (SIAC) filtering. However, it is a computationally complex task to perform this in an efficient manner, which becomes an even greater issue considering nonquadrilateral mesh structures. In this paper, we present an extension of this SIAC filter to structured triangular meshes. The basic theoretical assumption in the previous implementations of the postprocessor limits the use to numerical solutions solved over a quadrilateral mesh. However, this assumption is restrictive, which in turn complicates the application of this postprocessing technique to general tessellations. Additionally, moving from quadrilateral meshes to triangulated ones introduces more complexity in the calculations as the number of integrations required increases. In this paper, we extend the current theoretical results to variable coefficient hyperbolic equations over structured triangular meshes and demonstrate the effectiveness of the application of this postprocessor to structured triangular meshes as well as exploring the effect of using inexact quadrature. We show that there is a direct theoretical extension to structured triangular meshes for hyperbolic equations with bounded variable coefficients. This is a challenging first step toward implementing SIAC filters for unstructured tessellations. We show that by using the usual B-spline implementation, we are able to improve on the order of accuracy as well as decrease the magnitude of the errors. These results are valid regardless of whether exact or inexact integration is used. The results here demonstrate that it is still possible, both theoretically and computationally, to improve to $2k+1$ over the DG solution itself for structured triangular meshes.

Key words. high-order methods, discontinuous Galerkin, smoothness-increasing accuracy-conserving filtering, accuracy enhancement

AMS subject classification. 65M60

DOI. 10.1137/110830678

1. Introduction and motivation.

In this paper we consider accuracy enhancement of numerical solutions to two-dimensional linear hyperbolic equations of the form

$$u_t + \sum_{i=1}^{2} \frac{\partial}{\partial x_i}(A_i(\mathbf{x})u) = 0, \qquad \mathbf{x} \in \Omega \times [0,T], \tag{1.1}$$

$$u(\mathbf{x},0) = u_o(\mathbf{x}), \qquad \mathbf{x} \in \Omega,$$

*Received by the editors April 13, 2011; accepted for publication (in revised form) June 22, 2011; published electronically September 22, 2011.
http://www.siam.org/journals/sinum/49-5/83067.html

†School of Computing, University of Utah, Salt Lake City, UT 84112 (mirzaee@cs.utah.edu, kirby@cs.utah.edu). The work of these authors was supported by the Air Force Office of Scientific Research (AFOSR), Computational Mathematics Program (Program Manager: Dr. Fariba Fahroo), under grant FA9550-08-1-0156.

‡Department of Mathematics, University of Science and Technology of China, Hefei, Anhui 230026, People's Republic of China (jlyue@mail.ustc.edu.cn). This work was performed while the author was visiting Delft Institute of Applied Mathematics, Delft University of Technology, 2628 CD Delft, The Netherlands.

§Corresponding author. Delft Institute of Applied Mathematics, Delft University of Technology, 2628 CD Delft, The Netherlands (J.K.Ryan@tudelft.nl). This author's work was supported by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant FA8655-09-1-3055.

where $\Omega \in \mathbb{R}^2$, and $A_i(\mathbf{x})$, $i = 1, 2$, is bounded in the L^∞-norm. We assume that the numerical solution is obtained using the discontinuous Galerkin (DG) method where the numerical domain is subdivided into a structured triangular mesh. DG makes use of a variational formulation; however, instead of using a piecewise polynomial basis that requires continuity across element interfaces as typical in finite element methods (FEM), DG instead implements piecewise polynomials on each element with an imposed weak-continuity condition through the fluxes. This gives the DG method greater implementation flexibility across a variety of platforms. This lack of continuity, however, can be a hindrance when visualizing discontinuous data. For quadrilateral mesh structures, this problematic issue has been addressed through the use of smoothness-increasing accuracy-conserving (SIAC) filters [11]. These SIAC filters introduce continuity of $\mathcal{C}^{k-1}$ back into the solution at element interfaces, where k is the highest-degree polynomial used in the DG solution. For linear hyperbolic equations solved over uniform quadrilateral meshes, this also improves the order of accuracy from $\mathcal{O}(h^{k+1})$ to $\mathcal{O}(h^{2k+1})$. This idea was originally proposed as an accuracy enhancement technique for FEM for elliptic problems by Bramble and Schatz [1] and extended to discontinuous Galerkin methods for linear hyperbolic equations by Cockburn et al. [3, 4].

One drawback of the previous implementations of the postprocessor is that there is a basic assumption that the data is obtained over a uniform quadrilateral mesh. However, this assumption is restrictive, which makes the application of this postprocessing technique to general tessellations a challenging task. In this paper, we demonstrate that the theoretical extension to variable coefficient equations over structured triangular meshes is straightforward and show the behavior and complexity of the computational extension of this SIAC filter to structured triangular meshes. Moving from quadrilateral meshes to triangulated ones introduces more complexity in the calculations as the number of required integrations increases. Therefore, we not only introduce the results of the application of this postprocessor to structured triangular meshes, but we also explore the effect of using inexact quadrature as done in [8]. This is a challenging first step toward implementing SIAC filters for unstructured tessellations. By using the usual B-spline implementation, we are able to improve on the order of accuracy as well as decrease the magnitude of the errors. We are essentially able to increase the order of accuracy from $\mathcal{O}(h^{k+1})$ to approximately $\mathcal{O}(h^{2k+1})$ for structured triangular meshes and show accuracy enhancement for smoothly varying meshes and criss-cross meshes. These results are valid regardless of whether we employ exact or inexact integration.

We begin by reviewing the basics of DG methods over triangulations and SIAC filtering for quadrilateral meshes. In section 2, we show how there is a natural theoretical extension of the accuracy enhancing capabilities of the SIAC filter for structured triangular meshes. In section 3, we continue by showing the computational extension and demonstrating the complexity of the integrations. We then propose to implement inexact integration. In section 4, we give numerical results confirming the usefulness of our SIAC filter for triangulated meshes.

1.1. The DG formulation for triangular mesh structures. In this section, we provide an overview of the DG method over a domain that is subdivided into triangular elements. A more thorough study of the DG formulation can be found in the series of papers [2, 7].

We consider the numerical solution of a two-dimensional linear hyperbolic equa-

tion of the form

$$u_t + \sum_{i=1}^{2} \frac{\partial}{\partial x_i}(A_i(\mathbf{x})u) = 0, \qquad \mathbf{x} \in \Omega \times [0,T], \tag{1.2}$$
$$u(\mathbf{x},0) = u_o(\mathbf{x}), \qquad \mathbf{x} \in \Omega,$$

where $\mathbf{x} \in \Omega$, $t \in \mathbb{R}$, and $A_i(\mathbf{x})$, $i = 1,2$, is bounded in the L^∞-norm. We also assume smooth initial conditions are given along with periodic boundary conditions.

We begin by defining our mesh such that the computational domain, Ω, consists of N nonoverlapping triangular elements. We designate these by τ_e and assume that their characteristic length is h. The tessellated computational domain then is given by $\widetilde{\Omega} = \bigcup_{e=1}^{N} \tau_e$. We also define V_h to be the approximation space consisting of piecewise polynomials of degree at most k on element τ_e,

$$V_h = \Big\{ \varphi \in L^2(\widetilde{\Omega}) : \quad \varphi|_{\tau_e} \in \mathbb{P}^k(\tau_e) \quad \forall \tau_e \in \widetilde{\Omega} \Big\}. \tag{1.3}$$

We will approximate the exact solution $u(\mathbf{x})$ with $u_h(\mathbf{x}) \in V_h$.

To begin defining our numerical method, we consider the weak form of (1.3) in order to derive our DG approximation. That is, we multiply (1.3) by a smooth function $v(x)$ and integrate over τ_e. We then make use of Green's theorem in the second term to obtain a formulation that includes boundary terms. Next, the test function, v, is replaced with a piecewise polynomial function, $v_h \in \mathbf{V}_h$, in the approximation space. The flux function is defined to be $f_i(\mathbf{x},t) = A_i(\mathbf{x})u_h(\mathbf{x},t)$, $i = 1,2$. The DG formulation is then given by

$$\int_{\tau_e} \frac{\partial u_h}{\partial t} v_h \, d\mathbf{x} - \sum_{i=1}^{2} \int_{\tau_e} f_i(\mathbf{x},t) \frac{\partial v_h}{\partial x_i} \, d\mathbf{x} + \sum_{i=1}^{2} \int_{\partial \tau_e} \hat{f}_i \hat{n}_i v_h \, ds = 0, \tag{1.4}$$

where $\partial\tau_e$ is the boundary of the element τ_e, and $\hat{n}_i$ denotes the unit outward normal to the element boundary in the ith direction. Notice that the flux is multiply defined at element interfaces and, therefore, we impose the definition that $\hat{f}_i \hat{n}_i = h(u_h(\mathbf{x}^{\text{exterior}},t), u_h(\mathbf{x}^{\text{interior}},t))$ is a consistent two-point monotone Lipschitz flux as in [2]. We note that if we sum over all the elements, τ_e, we can rewrite (1.4) as

$$\left(\frac{\partial u_h}{\partial t}, v_h \right)_{\widetilde{\Omega}} + B(\mathsf{A}, u_h; v_h)_{\widetilde{\Omega}} = 0, \tag{1.5}$$

where

$$B(\mathsf{A}, u_h; v_h) = -\sum_{i=1}^{2} \left(A_i(\mathbf{x}) u_h(\mathbf{x},t), \frac{\partial v_h}{\partial x_i} \right)_{\widetilde{\Omega}} + \sum_{\tau_e} \sum_{i=1}^{2} \int_{\partial \tau_e} A_i(\mathbf{x}) \hat{u_h} v_h \hat{n}_i \, ds, \tag{1.6}$$

and we have denoted $A_i(\mathbf{x})$, $i = 1,2$, by A for simplicity.

The approximate solution, u_h, within an element is given by

$$u_h(\mathbf{x},t) = \sum_{p=0}^{k} \sum_{q=0}^{k-p} u_{\tau_e(t)}^{(p,q)} \phi^{(p,q)}(\mathbf{x}), \qquad \mathbf{x} \in \tau_e, \tag{1.7}$$

where $u_{\tau_e}^{(p,q)}(t)$ represents the expansion coefficients and $\phi^{(p,q)}(\mathbf{x})$ are the given basis functions. We note here that for our DG solvers, we are using the NEKTAR++ implementation given in [7] and available online at http://www.nektar.info.

1.2. A brief review on SIAC filters. We begin by reviewing the implementation of the postprocessor over *quadrilateral meshes*. This is well studied and details can be found in [9, 5]. For postprocessing over triangular mesh structures, we simply modify the existing quadrilateral mesh implementation and apply the same ideas for triangulated tessellations. For the purposes of this paper, we denote the two-dimensional coordinate system as (x_1, x_2). However, when we discuss the mesh structure, we denote the cell centers in the x_1-direction by x_i and the cell centers in the x_2-direction by y_j.

Assume that we are given an evaluation point $(\bar{x}, \bar{y}) \in I_{i,j} = [x_{i-1/2}, x_{i+1/2}] \times [y_{j-1/2}, y_{j+1/2}]$. The postprocessed solution at time $t = T$ is given by

$$u^\star(\bar{x}, \bar{y}) = \frac{1}{\triangle x \triangle y} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} K^{\nu,\ell}\left(\frac{\bar{x} - x_1}{\triangle x}\right) K^{\nu,\ell}\left(\frac{\bar{y} - x_2}{\triangle y}\right) u_h(x_1, x_2)\, dx_1 dx_2, \tag{1.8}$$

where $K^{\nu,\ell}$, $\nu = 2(k+1)$, $\ell = k+1$, is the postprocessing convolution kernel and $u_h(x_1, x_2)$ is the DG solution given over a quadrilateral mesh.

The convolution kernel in multiple dimensions is just a tensor product of the one-dimensional kernels. The one-dimensional kernel is a linear combination of B-splines,

$$K^{\nu,\ell}(x) = \sum_{\gamma=0}^{r} c_\gamma \psi^{(\ell)}(x - x_\gamma), \tag{1.9}$$

where $\psi^{(\ell)}(x)$ is generated by convolving $\chi_{[-1/2,1/2]}$ with itself k times and c_γ are chosen such that the kernel reproduces polynomials of degree r. In general, r represents the number of B-splines used in the postprocessor. We use this general designation so that a more local kernel can be used in the interior of the domain, consisting of $r_1 + 1 = 2k + 1$ B-splines, and at the boundary a less localized kernel consisting of $r_2 + 1 = 4k + 1$ B-splines is used. In the above definition, $x_\gamma = -\frac{r}{2} + \gamma + \lambda(\bar{x})$ are the kernel nodes. These nodes have a shift function $\lambda(\bar{x})$ which depends upon the evaluation point, $\bar{x}$, and is given by

$$\lambda(\bar{x}) = \begin{cases} \min\{0, -\frac{r+\ell}{2} + \frac{\bar{x} - x_L}{h}\}, & \bar{x} \in [x_L, \frac{x_L + x_R}{2}), \\ \max\{0, \frac{r+\ell}{2} + \frac{\bar{x} - x_R}{h}\}, & \bar{x} \in [\frac{x_L + x_R}{2}, x_R], \end{cases} \tag{1.10}$$

where x_L and x_R denote the left and right boundaries. Furthermore, we combine the kernels in the interior and at the boundaries through a convex combination:

$$u_h^\star(\bar{x}) = \theta(\bar{x})(u_h^\star(\bar{x}))_{r_1 = 2k} + (1 - \theta(\bar{x}))(u_h^\star(\bar{x}))_{r_2 = 4k}, \tag{1.11}$$

where $\theta \in \mathcal{C}^{k-1}$. For further details, see [10].

In the simplest case, we implement the symmetric filter and assume $\triangle x = \triangle y = h$ so that the mesh-spacing is uniform in the x_1- and x_2-directions. We can then use the compact support property of the kernel and the quadrilateral elements of this implementation to write this as a finite local sum,

$$u^\star(\bar{x}, \bar{y}) = \frac{1}{h^2} \sum_{r=-k'}^{k'} \sum_{s=-k'}^{k'} \int\int_{I_{i+r,j+s}} K^{\nu,\ell}\left(\frac{\bar{x} - x_1}{h}\right) K^{\nu,\ell}\left(\frac{\bar{y} - x_2}{h}\right) u_h(x_1, x_2)\, dx_1 dx_2, \tag{1.12}$$

where $k' = \lceil (3k+1)/2 \rceil$ and k is the highest polynomial degree used in the approximation. We note that near a boundary a skewed kernel is implemented and the support

would vary, but still remain compact. Furthermore, notice that in this representation we have not yet taken into account the kernel breaks in this equation. The integration over one element $I_{i+r,j+s}$ would be further divided into four integration regions that respect both the DG-element breaks and the continuity breaks in the kernel.

For the purposes of this paper, we call the quadrilateral mesh element, $I_{i,j}$, a superelement that is diagonally intersected to form two triangular elements in our structured mesh. If we assume that $(\bar{x}, \bar{y}) \in I_{i,j}$, where I represents the quadrilateral superelement in which the triangular element is contained, then $I_{i+r,j+s}$ indicates the superelement that is (partially) covered by the kernel support.

2. Higher-order accuracy in DG solutions. In this section, we discuss the theoretical extension of the SIAC filter to multiple dimensions. We build on previous proofs in [4, 6] and account for a variable coefficient as well as the triangular mesh structure. This allows us to arrive at an estimate that demonstrates that the order accuracy of the DG solution can be improved from $\mathcal{O}(h^{k+1})$ to $\mathcal{O}(h^{2k+1})$ for structured triangular meshes such as that in Figure 1.

We begin by stating our main theorem.

THEOREM 2.1. *Let u_h be the DG solution for the variable coefficient problem* (1.3)*, where $A_i(\mathbf{x})$, $i = 1, 2$, is bounded in the L^∞-norm and $u_h^\star = K_H^{\nu,\ell} \star u_h$. $K_H^{\nu,\ell} = \frac{1}{H} K^{\nu,\ell}(\frac{\mathbf{x}}{H})$ is the position-dependent SIAC kernel given in* (1.9)*. Given sufficient smoothness in the initial data, we have that*

$$\|u - K_H^{\nu,\ell} \star u_h\|_\Omega \leq \mathsf{C} h^{2k+1}, \tag{2.1}$$

where C depends upon the smoothness of the solution as well as the boundedness of $A_i(\mathbf{x})$, $i = 1, 2$.

In this statement, we see that it is possible to improve the order of accuracy of the approximation through the use of a B-spline kernel. This improvement is possible *provided the negative-order norm of the solution and the divided differences are of higher-order*. Since we are considering the simplest case of structured triangular mesh, we have translation invariance of the kernel. It only remains to prove that we have higher-order accuracy in the negative-order norm for the solution and the divided differences. To show this, we follow the proofs in [4, 6]. We note that in those proofs, the coefficients multiplying the convection term are constant. Here, we consider the variable-coefficient case.

Proof. We note that we can rewrite the estimate in (2.1) as

$$\|u - K_H^{\nu,\ell \mathbf{1}} \star u_h\|_\Omega \leq \|u - K_H^{\nu,\ell \mathbf{1}} \star u\|_\Omega + \|K_H^{\nu,\ell \mathbf{1}} \star (u - u_h)\|_\Omega. \tag{2.2}$$

The estimate for the first term on the right-hand side was shown in [4]. For the second term, we need to demonstrate that the divided differences of the error in a negative-order norm are of higher-order for the variable coefficient equation solved over a structured triangular mesh. We show this by considering the inner product of the errors from the DG solution with a smooth function, $\Phi(x)$, that is the final time condition to the dual problem. We can define this dual problem by

$$\varphi_t + \sum_{i=1}^{2} A_i(\mathbf{x}) \varphi_{x_i} = 0, \qquad (\mathbf{x}, t) \in \Omega \times (0, T], \tag{2.3}$$

$$\varphi(x, T) = \Phi(x), \tag{2.4}$$

with φ having periodic boundary conditions. Multiplying the equation for the exact solution, (1.3), by φ and the dual equation, (2.3), by the exact solution, we see that

we have $\frac{d}{dt}(u,\varphi)=0$. We can use this to begin estimating $((u-u_h)(T),\Phi)_\Omega$, which would become

$$\begin{aligned}((u-u_h)(T),\Phi)_\Omega &= (u-u_h,\varphi)_\Omega(0) - \int_0^T \{((u_h)_t,\varphi)_\Omega + (u_h,\varphi_t)_\Omega\}\,dt \\ &= (u-u_h,\varphi)_\Omega(0) - \int_0^T \{((u_h)_t,\varphi-\chi)_\Omega + B(\mathsf{A},u_h;\varphi-\chi)_\Omega\}\,dt \\ &= \Theta_1+\Theta_2, \end{aligned} \tag{2.5}$$

where $B(\mathsf{A},u_h;\varphi-\chi)$ is the bilinear form given in (1.6) and φ is the solution to the dual equation (2.3).

We note that the proof of Θ_1 being of higher-order was given in [4] and is $|\Theta_1|\le C_1h^{2k+2}\|u_o\|_{k+1}\|\varphi(0)\|_{k+1}$. The proof for Θ_2 uses $\chi=P\varphi$, the projection of φ and Lipschitz continuity, to obtain

$$\begin{aligned}|\Theta_2| &\le \left|-\int_0^T \{((u_h)_t,\varphi-\chi)_\Omega + B(\mathsf{A},u_h;\varphi-\chi)_\Omega\}\,dt\right| \le \left|-\int_0^T \{B(\mathsf{A},u_h;\varphi-\chi)_\Omega\}\,dt\right| \\ &\le \left|\sum_{i=1}^2\int_0^T \left\{(A_i(\mathbf{x})u_h(\mathbf{x},t),(\varphi-P\varphi)_{x_i}) - \sum_{\tau_e}\int_{\partial\tau_e} A_i(\mathbf{x})\hat{u}_h\hat{n}_i(\varphi-P\varphi)\,ds\right\}dt\right|. \end{aligned} \tag{2.6}$$

Adding and subtracting the term $((A_i(x)u)_{x_i},\varphi-P\varphi)_\Omega$ and using simple inequalities gives the bound on Θ_2 as

$$\begin{aligned}|\Theta_2| \le Ch^{2k+1}&\left(\int_0^T \|u_h-u\|_\Omega^2\,dt\right)^{1/2}\left(\int_0^T \|\varphi\|_{k+1}^2\,dt\right)^{1/2} \\ &+ Ch^{2k+2}\left(\int_0^T \|\varphi\|_{k+1}^2\,dt\right)^{1/2}. \end{aligned} \tag{2.7}$$

For the negative-order norm of the solution, we then have the estimate

$$\|u-u_h\|_{-(k+1),\Omega} = \mathsf{C}\,h^{2k+1}, \tag{2.8}$$

where C depends on the smoothness of the solution and the bound on $\|\mathsf{A}\|_{L^\infty(\Omega)}$. We have obtained this by using the above bounds on $|\Theta_1|$, $|\Theta_2|$, as well as the definition of the negative-order norm together with the regularity for the variable coefficient equation, $\|\varphi(\mathbf{x},t)\|_{\ell,\Omega}\le C\|\varphi(\mathbf{x},0)\|_{\ell,\Omega}$, where C is a constant that depends on $\|\mathsf{A}\|_{L^\infty(\Omega)}$.

We emphasize that the proof of the negative-order norm estimate of the DG solution did not rely on the mesh from the DG solution. The mesh type only plays a role in the estimation of the divided differences that help to bound the error. However, because we are assuming a structured triangular mesh, we maintain the translation invariance property. The constant now relies on the bound on the variable coefficient A_i, $i=1,2$. To show that this is indeed the case, we first define the DG method for the divided differences of the variable coefficient equation,

$$\partial_H^\alpha u_t + \sum_{i=1}^2 \partial_H^\alpha (A_i(\mathbf{x})u)_{x_i} = 0, \qquad \mathbf{x}\in\mathbb{R}^2\times(0,T], \tag{2.9}$$

$$\partial_H^\alpha u(\mathbf{x},0) = \partial_H^\alpha u_o(\mathbf{x}), \tag{2.10}$$

where α is the order of the divided difference, $\partial_H^\alpha = \partial_{H_1}^{\alpha_1}\partial_{H_2}^{\alpha_2}$, and

$$\partial_{H_j}^m v(x) = \frac{1}{H^m}\sum_{i=0}^{m}(-1)^i\binom{m}{i}\, v\left(x + \left(\frac{m}{2} - i\right)He_j\right).$$

The DG scheme is given by

$$(\partial_H^\alpha u_h, v_h)_\Omega + B_\alpha(\mathsf{A}, u_h; v_h)_\Omega = 0 \tag{2.11}$$

for all $v_h \in V_h$, where

$$B_\alpha(\mathsf{A}, u_h; v_h)_\Omega = -\sum_{i=1}^{2}\left(\partial_H^\alpha(A_i(\mathbf{x})u_h), \frac{\partial v_h}{\partial x_i}\right)_\Omega + \sum_{\tau_e}\sum_{i=1}^{2}\int_{\partial\tau_e}\partial_H^\alpha(A_i(\mathbf{x})\widehat{u_h})v_h\widehat{n}_i\, ds. \tag{2.12}$$

For the dual equation, we need to factor in the divided differences of the error:

$$\partial_H^\alpha(\varphi_\alpha)_t + \sum_{i=1}^{2} A_i(\mathbf{x})\partial_H^\alpha(\varphi_\alpha)x_i = 0, \qquad (\mathbf{x}, t) \in \Omega \times (0, T], \tag{2.13}$$

$$\varphi_\alpha(\mathbf{x}, T) = \widetilde{\Phi}_\alpha(\mathbf{x}). \tag{2.14}$$

Defining the dual equation in this way gives the relation $\frac{d}{dt}(\partial_H^\alpha u, \varphi_\alpha)_\Omega = 0$. Having this relation is an important step in the proof of the higher-order accuracy in the negative-order norms.

We can perform a similar rearrangement of terms as above to obtain

$$\begin{aligned}(\partial_H^\alpha(u - u_h)(T), \widetilde{\Phi}_\alpha)_\Omega &= (\partial_H^\alpha(u - u_h), \varphi_\alpha)_\Omega(0)\\ &\quad - \int_0^T \{(\partial_H^\alpha(u_h)_t, \varphi_\alpha - \chi)_\Omega + B_\alpha(\mathsf{A}, \partial_H^\alpha u_h; \varphi_\alpha - \chi)_\Omega\}\, dt\\ &\quad - \int_0^T \{(\partial_H^\alpha(u_h), (\varphi_\alpha)_t)_\Omega - B_\alpha(\mathsf{A}, \partial_H^\alpha u_h; \varphi_\alpha)_\Omega\}\, dt\\ &= d\Theta_1 + d\Theta_2 + d\Theta_3.\end{aligned} \tag{2.15}$$

Here, $d\Theta_1$, $d\Theta_2$, and $d\Theta_3$ are

$$d\Theta_1 = (\partial_H^\alpha(u - u_h), \varphi_\alpha)_\Omega(0), \tag{2.16}$$

$$d\Theta_2 = -\int_0^T \{(\partial_H^\alpha(u_h)_t, \varphi_\alpha - \chi)_\Omega + B_\alpha(\mathsf{A}, \partial_H^\alpha u_h; \varphi_\alpha - \chi)_\Omega\}\, dt, \tag{2.17}$$

$$d\Theta_3 = -\int_0^T \{(\partial_H^\alpha(u_h), (\varphi_\alpha)_t)_\Omega - B_\alpha(\mathsf{A}, \partial_H^\alpha u_h; \varphi_\alpha)_\Omega\}\, dt. \tag{2.18}$$

The bounds on $d\Theta_1$ and $d\Theta_2$ provide a similar estimate to that appearing above and, therefore, we concentrate only on bounding $d\Theta_3$. For the interior part of the integral

of $d\Theta_3$ we have

$$\begin{aligned} INT(d\Theta_3) &= (\partial_H^\alpha(u_h), (\varphi_\alpha)_t)_\Omega - B_\alpha(\mathsf{A}, \partial_H^\alpha u_h; \varphi_\alpha)_\Omega \\ &= (-1)^\alpha (u_h, \partial_H^\alpha(\varphi_\alpha)_t)_\Omega + \sum_{i=1}^{2} (\partial_H^\alpha(A_i(\mathbf{x})u_h), (\varphi_\alpha)_{x_i})_\Omega \\ &\quad - \sum_{\tau_e} \sum_{i=1}^{2} \int_{\partial\tau_e} \partial_H^\alpha(A_i(\mathbf{x})\widehat{u_h})\varphi_\alpha \widehat{n}_i \, ds \\ &= (-1)^\alpha (u_h, \partial_H^\alpha(\varphi_\alpha)_t)_\Omega + \sum_{i=1}^{2} (-1)^\alpha \, (u_h, A_i(\mathbf{x})\partial_H^\alpha(\varphi_\alpha)_{x_i})_\Omega = 0. \end{aligned} \tag{2.19}$$

Combining the estimates and using the regularity of φ, we then have

$$(\partial_H^\alpha(u - u_h)(T), \widetilde{\Phi})_\Omega \leq Ch^{2k+1}, \tag{2.20}$$

proving our theorem. □

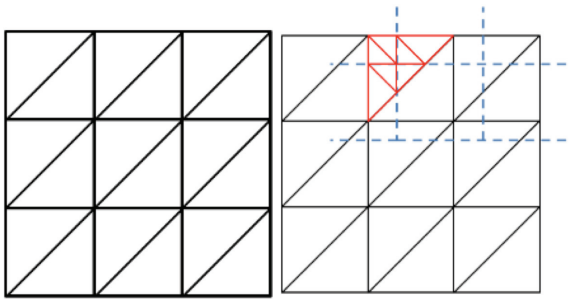
We point out that the negative-order norm estimates for the solution did not rely on the mesh type, and that in fact we do see higher-order convergence; however, the divided differences do rely on the mesh type. In order to extend this to unstructured meshes, we will need to focus our attention on improving the estimates for the divided differences of the error. The proofs for the accuracy extracting capabilities of the SIAC filter do not change.

3. SIAC filters for triangular meshes. In this section, we provide the implementation details of the postprocessor over triangular mesh structures. The implementation discussed in this section is used to produce the results given in section 4.

3.1. Postprocessing on the consistent integration mesh. The current version of the SIAC filter simply takes the existing quadrilateral mesh implementation and applies the same ideas for triangular mesh structures. Here we discuss the details of this implementation, including the modification of these ideas to structured triangular meshes. For simplicity, we discuss only the implementation of the symmetric kernel over the structured triangular meshes as shown in Figure 1. This discussion can easily be adapted to handle the position-dependent kernel near the boundaries. Second, we discuss the formidable challenge it poses with respect to numerical integration, as the number of integrals increases with this implementation.

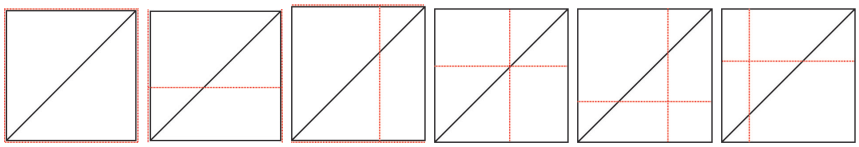
The postprocessing routine takes as input an array of the polynomial modes used in the DG method and produces the values of the postprocessed solution at a set of specified evaluation points. For simplicity, we assume these evaluation points correspond with specific quadrature points. Let us examine how to calculate the postprocessed value at a single evaluation point. Postprocessing of the entire domain is obtained by repeating the same procedure for all of the evaluation points. For the sake of simplicity, we consider the case of a DG solution produced over a structured triangular mesh shown in Figure 1(a).

To calculate the integral involved in the postprocessed solution (see (1.12)) exactly, we need to decompose the quadrilateral element $I_{i+r,j+s}$ into subelements that respect the DG-element breaks as well as the kernel knots (which we also refer to as breaks); the resulting integral is calculated as the summation of the integrals over each subinterval. The term *consistent integration* is used to denote the integration that respects both the DG and kernel breaks by finding the necessary subelements



(a) (b)

FIG. 1. (a) *Structured triangular mesh.* (b) *A sample triangulation of an element to produce the subelements (light). Dash lines represent the kernel breaks.*



(a) (b) (c) (d) (e) (f)

FIG. 2. *One superelement of the DG mesh (dark) with possible kernel breaks (light). The number of integration regions increases with triangular meshes, with possibly two to seven different regions.*

such that the integral on each subelement can be done exactly to machine precision. Figure 2 depicts the possible number of integration regions per superelement given, and Figure 1(b) depicts a typical triangulation of an element in a structured domain to produce the subelements.

Let us outline exactly how the integrals are further split into subregions that respect both the DG-mesh breaks and the kernel knots. We begin with the simplest integral division, that of the DG break (Figure 2(a)). Assuming the structured triangular mesh, we use one quadrilateral element, which we refer to as a superelement. We frame our discussion in terms of this superelement in order to demonstrate the complexity of extending these ideas to triangular meshes. This element is given by

$$\begin{aligned} I_{i+r,j+s} &= \left[x_{i+r-1/2}, x_{i+r+1/2}\right] \times \left[y_{j+s-1/2}, y_{j+s+1/2}\right] \\ &= \left[x_{i+r} - \frac{h}{2}, x_{i+r} + \frac{h}{2}\right] \times \left[y_{j+s} - \frac{h}{2}, y_{j+s} + \frac{h}{2}\right]. \end{aligned}$$

To create the structured triangular mesh, we subdivide this superelement into two triangular elements, so that we have

$$\begin{aligned} I_{i+r,j+s} = &\left[x_{i+r} - \frac{h}{2}, x_{i+r} + \frac{h}{2}\right] \times \left[y_{j+s} - \frac{h}{2}, (x_1 - x_{i+r}) + y_{j+s}\right] \\ &\cup \left[x_{i+r} - \frac{h}{2}, x_{i+r} + \frac{h}{2}\right] \times \left[(x_1 - x_{i+r}) + y_{j+s}, y_{j+s} + \frac{h}{2}\right], \end{aligned} \tag{3.1}$$

where (x_{i+r}, y_{j+s}) is the center of the quadrilateral element $I_{i+r,j+s}$. The form of the integral in (1.12) is then

$$\begin{aligned}(3.2)\ \frac{1}{h^2}\int_{x_{i+r-1/2}}^{x_{i+r+1/2}} & K^{2(k+1),k+1}\left(\frac{\bar{x}-x_1}{h}\right) \\ & \cdot\left\{\int_{y_{j+s-1/2}}^{(x_1-x_{i+r})+y_{j+s}} K^{2(k+1),k+1}\left(\frac{\bar{y}-x_2}{h}\right) u_h(x_1,x_2)\,dx_2\right. \\ & \left.+\int_{(x_1-x_{i+r})+y_{j+s}}^{y_{j+s+1/2}} K^{2(k+1),k+1}\left(\frac{\bar{y}-x_2}{h}\right) u_h(x_1,x_2)\,dx_2\right\} dx_1.\end{aligned}$$

Note that respecting the DG-element breaks alone increases the number of integrals from two to three for one superelement. Having properly separated the integrals for the DG-element breaks, we can now concentrate on the kernel B-spline breaks. Let us consider only the simplified case of the symmetric kernel. In this case, the full kernel support is given by $\left[\bar{x}-\frac{3k+1}{2}h, \bar{x}+\frac{3k+1}{2}h\right]\times\left[\bar{y}-\frac{3k+1}{2}h, \bar{y}+\frac{3k+1}{2}h\right]$, with the B-spline breaks in the x_1-direction occurring at $\bar{x}+m_1h$, and at $\bar{y}+m_2h$ for the x_2-direction, where $m_1, m_2 = -\frac{3k+1}{2}, \ldots, \frac{3k+1}{2}$. We remind the reader that $(\bar{x}, \bar{y})$ is the evaluation point at which the filtered solution will be calculated. In Figure 2, one superelement of the DG mesh with possible kernel breaks (light) is shown. Notice that the possibilities for the number of integrals increases when we allow for triangular mesh structures. If we were to maintain this mesh structure without further subdivisions, we would have the following: For one superelement consisting of two triangular elements, there are only two integration regions per superelement in case (b). These regions coincide with the DG-element breaks. However, if the kernel breaks do not occur at superelement boundaries, the number of integration regions increases to four in case (c) and (d) and six integration regions in case (e), and cases (f) and (g) show seven integration regions per superelement.

We note here that although we concentrate on a structured triangular mesh, we also have to consider the possibility of an unstructured mesh. For our Gaussian quadrature rules to be applicable, we need to divide the integration region to either triangles or quadrilaterals or both when possible. However, in three dimensions and in the presence of totally unstructured grids, it is not always possible to tessellate the integration region, which is the result of the DG mesh intersecting with the kernel mesh, to a combination of hexahedral and tetrahedral, whereas for triangulation it is always possible. Therefore, to account for a more complex situation as well as ease of implementation, we always triangulate the integration region in two dimensions. This results in a mesh as shown in Figure 1(b).

Having developed the consistent integration mesh on a single element that respects both the DG element breaks and the B-spline kernel (knot) breaks, our postprocessed solution is

$$\begin{aligned}(3.3)\ & u_h^\star(\bar{x},\bar{y}) \\ &= \sum_{r,s\in supp\{K\}}\sum_{n=1}^{N}\frac{1}{h^2}\int_{\tau_n} K^{2(k+1),k+1}\left(\frac{\bar{x}-x_1}{h}\right) K^{2(k+1),k+1}\left(\frac{\bar{y}-x_2}{h}\right) u_h(x_1,x_2)\,dx_1dx_2,\end{aligned}$$

where $\tau_n \in I_{i+r,j+s}$ represents a subelement of $I_{i+r,j+s}$ and N is the total number of subelements per that superelement. The integral in (3.3) is a two-dimensional

integration over a triangular region. For simplicity, it is necessary to map τ_n to a standard triangular region in Cartesian coordinates ξ_1 and ξ_2,

$$\Omega_{st} = \{-1 \leq \xi_1 \leq 1, \xi_2; \xi_1 + \xi_2 \leq 0\}. \tag{3.4}$$

To do this, denote $\tau_n = \Omega_e$ as an arbitrary triangular region in the global Cartesian coordinate system (x_1, x_2). Then we can relate the integral over *one element*, τ_n, to an integral over the standard region, Ω_{st}, through the usual coordinate transformation, $\xi = A_{2x2}\mathbf{x} + \mathbf{b}$, so that we have

$$\begin{aligned} &\frac{1}{h^2}\int_{\Omega_e} K^{2(k+1),k+1}\left(\frac{\bar{x}-x_1}{h}\right) K^{2(k+1),k+1}\left(\frac{\bar{y}-x_2}{h}\right) u_h(x_1,x_2)dx_1dx_2 \\ &= \frac{1}{h^2}\int_{\Omega_{st}} \widetilde{K}(\xi_1,\xi_2,\bar{x})\widetilde{K}(\xi_1,\xi_2,\bar{y})\,\tilde{u}_h(\xi_1,\xi_2)\left|\frac{\partial(x_1,x_2)}{\partial(\xi_1,\xi_2)}\right| d\xi_1 d\xi_2. \end{aligned} \tag{3.5}$$

In the notation above, $\widetilde{K}(\xi_1, \xi_2, \bar{x})$ represents the kernel $K^{2(k+1),k+1}\left(\frac{\bar{x}-x_1}{h}\right)$ after a coordinate transform to the standard triangular region:

$$\begin{aligned} K^{2(k+1),k+1}(x_i) &= K^{2(k+1),k+1}\left(\frac{1}{\det(A_{2x2})}(A_{i,1}(\xi_1-\beta_1)+A_{i,2}(\xi_2-\beta_2))\right) \\ &= \widetilde{K}(\xi_1,\xi_2,\cdot), \end{aligned} \tag{3.6}$$

where $A_{i,j}$ are the elements of matrix A_{2x2}, β_i are the elements of vector $\mathbf{b}$ describing the transformation, and $i = 1, 2$. We have made a similar transformation for the numerical approximation. We note that in (3.5), $|\frac{\partial(x_1,x_2)}{\partial(\xi_1,\xi_2)}| = \det(A_{2x2})$ is the Jacobian of the coordinate transformation. We also want to remind the reader that for triangles the transform is affine, but in the general case it is not.

Furthermore, we note that the standard triangular region in (3.4) is not conveniently represented for one-dimensional function evaluations because the upper boundary of the region is described in terms of both coordinates. Therefore, to develop a suitable tensorial type basis within a triangular region, we need to use a coordinate system that has independent bounds. We do this by mapping into a region bounded by constants using collapsed coordinates as in [7]. This coordinate system also defines an appropriate system upon which we can perform important numerical operations such as integration. This is equivalent to transforming the triangular region into a quadrilateral region as shown in Figure 3. In other words, for one triangular element, we obtain

$$\begin{aligned} &\frac{1}{h^2}\int_{\Omega_{st}} \widetilde{K}(\xi_1,\xi_2,\bar{x})\widetilde{K}(\xi_1,\xi_2,\bar{y})\,\tilde{u}_h(\xi_1,\xi_2)\left|\frac{\partial(x_1,x_2)}{\partial(\xi_1,\xi_2)}\right| d\xi_1 d\xi_2 \\ &= \frac{1}{h^2}\int_{-1}^{1}\int_{-1}^{-\xi_2} \widetilde{K}(\xi_1,\xi_2,\bar{x})\widetilde{K}(\xi_1,\xi_2,\bar{y})\,\tilde{u}_h(\xi_1,\xi_2)\left|\frac{\partial(x_1,x_2)}{\partial(\xi_1,\xi_2)}\right| d\xi_1 d\xi_2 \\ &= \frac{1}{h^2}\int_{-1}^{1}\int_{-1}^{1} \widehat{K}\,(\eta_1,\eta_2,\bar{x})\,\widehat{K}\,(\eta_1,\eta_2,\bar{y})\,\,\widehat{u}_h(\eta_1,\eta_2)\left|\frac{\partial(\xi_1,\xi_2)}{\partial(\eta_1,\eta_2)}\right|\left|\frac{\partial(x_1,x_2)}{\partial(\xi_1,\xi_2)}\right| d\eta_1 d\eta_2 \\ &= S_{\tau_n}, \end{aligned} \tag{3.7}$$

where $\eta_1 = 2\frac{(1+\xi_1)}{(1-\xi_2)} - 1$, $\eta_2 = \xi_2$, and $\partial(\xi_1, \xi_2)/\partial(\eta_1, \eta_2)$ is the Jacobian of the Cartesian to local-coordinate transformation. $\widehat{K}\,(\eta_1, \eta_2, \cdot)$ is $\widetilde{K}(\xi_1, \xi_2, \cdot)$ after transformation

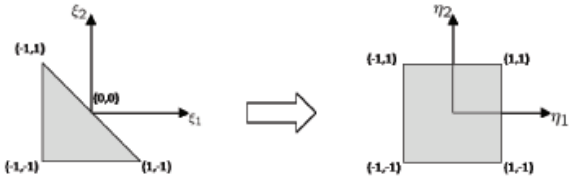


FIG. 3. *Triangle to rectangle transformation.*

to the collapsed coordinate system. For a more thorough discussion on numerical integration over triangular regions, we refer the reader to [7].

Furthermore, we replace $\widehat{u}_h(\eta_1, \eta_2)$ by its local (elemental) polynomial expansion, and we arrive at

(3.8)
$$S_{\tau_n} = \int_{-1}^{1}\int_{-1}^{1} \widehat{K}\left(\eta_1, \eta_2, \bar{x}\right) \widehat{K}\left(\eta_1, \eta_2, \bar{y}\right) \left(\sum_{p,q}^{k,k-p} \widehat{u}_{\tau_n}^{(p,q)} \phi_{\tau_n}^{(p,q)}(\eta_1, \eta_2) \right) \left|J_{2D}^1\right| \left|J_{2D}^2\right| d\eta_1 d\eta_2,$$

where $\phi_{\tau_n}^{(p,q)}$ are the two-dimensional basis functions that are formed from the tensor product of the one-dimensional basis functions, $\widehat{u}_{\tau_n}^{(p,q)}$ are the local polynomial modes on the mesh element that contains τ_n, and $J_{2D}^1 = \frac{\partial(x_1,x_2)}{\partial(\xi_1,\xi_2)} = \det(A_{2x2})$, $J_{2D}^2 = \frac{\partial(\xi_1,\xi_2)}{\partial(\eta_1,\eta_2)} = \frac{1}{2}(1-\eta_2)$ represent the Jacobians of the transformations.

After these many transformations, we have the following formula for performing consistent integration over a structured triangular element in the DG mesh:

$$u^\star(\bar{x}, \bar{y}) = \sum_{r,s \in supp\{K\}} \sum_{n=1}^{N} S_{\tau_n}. \tag{3.9}$$

We know that these integrals can be computed exactly to machine precision using Gaussian quadrature techniques with sufficient quadrature points, and therefore, the postprocessed solution in (1.8) will also be evaluated exactly to machine precision.

3.2. Postprocessing on the DG mesh. In the above calculations, we note that we have respected both kernel breaks and DG-element breaks. This has the potential for increasing the number of integral regions to be evaluated in the postprocessed solution from two to seven. Because numerical integration is quite costly, in this section we suggest an alternative to overcoming this issue by ignoring the kernel breaks, as at these breaks the kernel function maintains $\mathcal{C}^{k-1}$-continuity. The idea is that at the DG breaks there is a lack of continuity, whereas at the kernel breaks there exists $\mathcal{C}^{k-1}$-continuity. Gaussian quadrature can overcome the weak continuity of the kernel breaks by using more quadrature points, but cannot overcome that of the DG breaks. These ideas for a quadrilateral mesh were explored in [8].

If we choose to implement the SIAC kernel such that we respect only the DG-element breaks, then the postprocessed solution that we need to evaluate is now given

by

$$
\begin{aligned}
&u^\star(\bar{x},\bar{y})\\
&= \sum_{r,s\in supp\{K\}} \frac{1}{h^2}\int_{I_{i+r,j+s}} K^{2(k+1),k+1}\left(\frac{\bar{x}-x_1}{h}\right) K^{2(k+1),k+1}\left(\frac{\bar{y}-x_2}{h}\right) u_h(x_1,x_2)dx_1dx_2\\
&= \sum_{r,s\in supp\{K\}} \frac{1}{h^2}\int_{\Omega_{st}} \widetilde{K}(\xi_1,\xi_2,\bar{x})\widetilde{K}(\xi_1,\xi_2,\bar{y})\,\widetilde{u}_h(\xi_1,\xi_2)\left|J^1_{2D}\right| d\xi_1 d\xi_2\\
&= \sum_{r,s\in supp\{K\}} \frac{1}{h^2}\int_{-1}^{1}\int_{-1}^{-\xi_2} \widetilde{K}(\xi_1,\xi_2,\bar{x})\widetilde{K}(\xi_1,\xi_2,\bar{y})\,)\widetilde{u}_h(\xi_1,\xi_2)\left|J^1_{2D}\right| d\xi_1 d\xi_2\\
&= \sum_{r,s\in supp\{K\}} \frac{1}{h^2}\int_{-1}^{1}\int_{-1}^{1} \widehat{K}\left(\eta_1,\eta_2,\bar{x}\right)\widehat{K}\left(\eta_1,\eta_2,\bar{y}\right)\,\widehat{u}_h(\eta_1,\eta_2)\left|J^1_{2D}\right|\left|J^2_{2D}\right| d\eta_1 d\eta_2\\
&= \sum_{r,s\in supp\{K\}} S_{I_{i+r,j+s}}.
\end{aligned}
\tag{3.10}
$$

Notice that there are fewer integration regions—so we expect the computational cost to be reduced. We perform the integrations using Gaussian quadrature, where $J^1_{2D} = \frac{\partial(x_1,x_2)}{\partial(\xi_1,\xi_2)}$ is the two-dimensional Jacobian due to the mapping of the triangular element to the standard region and $J^2_{2D} = \frac{\partial(\xi_1,\xi_2)}{\partial(\eta_1,\eta_2)}$.

4. Numerical results. The main contribution of this section is the demonstration of the effectiveness of the quadrilateral B-spline postprocessor applied to DG solutions calculated over a structured triangular mesh and its accuracy enhancing capabilities. Furthermore, we present the results for the various choices of quadrature to see the computational effect of the numerical crimes committed in our integration. We examine both the L^2- and L^∞-errors, but note that the theory presented in this paper only applies to the L^2-errors. We note that all examples have been implemented using NEKTAR++ for the DG solution to the partial differential equation under consideration. The classical Runge–Kutta 4 time-stepping scheme was used for the time discretization and the CFL was taken so that spatial errors dominate. We note that this restriction is not necessary in practical applications—the errors will still be improved over the DG errors. However, in order to see the higher rate of convergence it is necessary to have such a restriction. For the consistent integration approach, we always use exact integration to calculate the L_2 projections, i.e., the number of quadrature points is what is required for exact integration. However, when performing integration considering only the DG mesh, we increase the number of quadrature points. In order to isolate the applicability of this filter, we assume that we have periodic boundary conditions for our test cases to simplify the application of the postprocessor so that it is only necessary to implement the symmetric filter with $r+1 = 2k+1$ B-splines. In addition, Gauss–Lobatto–Legendre (GLL) points are used in one direction, and Gauss–Radau–Legendre (GRL) points are used on the other. In all examples, we calculate the error for six quadrature points on each element, i.e., three GLL points on one direction and two GRL points on the other.

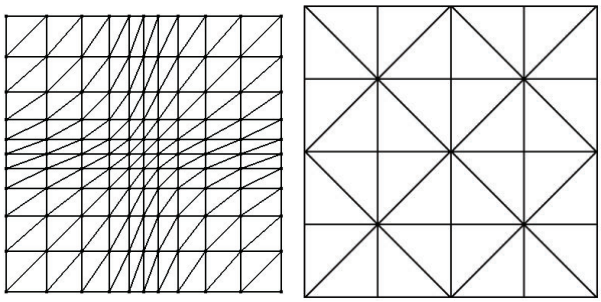
4.1. Constant coefficient linear advection equation. For this example we consider solutions of the equation

$$
u_t + u_x + u_y = 0, \qquad (x,y)\in(0,1)\times(0,1), \qquad T = 12.5 \tag{4.1}
$$

TABLE 1

Errors for the linear constant coefficient advection equation using $\mathbb{P}^2$, $\mathbb{P}^3$, and $\mathbb{P}^4$ polynomials for the structured (uniform) triangular mesh. Before postprocessing and after postprocessing on the DG mesh, where Q_0 represents GLL points on one direction and Q_1 represents GRL points on the other direction, CI indicates consistent integration.

$\mathbb{P}^2$								
Mesh	L^2 error	Order	L^∞ error	Order	L^2 error	Order	L^∞ error	Order
	Before postprocessing				Inexact postprocessing, $Q0 = 5, Q1 = 4$			
$10^2 \times 2$	6.86E-03	–	3.27E-02	–	2.32E-03	–	3.28E-03	–
$20^2 \times 2$	9.86E-04	2.78	4.35E-03	2.91	7.15E-05	5.02	1.02E-04	5.01
$40^2 \times 2$	1.36E-04	2.86	5.45E-04	3.00	2.22E-06	5.01	3.21E-06	4.99
	Inexact postprocessing, $Q0 = 34, Q1 = 33$				Consistent postprocessing, CI			
$10^2 \times 2$	2.32E-03	–	3.28E-03	–	2.32E-03	–	3.28E-03	–
$20^2 \times 2$	7.14E-05	5.02	1.01E-04	5.02	7.14E-05	5.02	1.01E-04	5.02
$40^2 \times 2$	2.18E-06	5.03	3.09E-06	5.03	2.18E-06	5.03	3.09E-06	5.03
$\mathbb{P}^3$								
	Before postprocessing				Inexact postprocessing, $Q0 = 6, Q1 = 6$			
$10^2 \times 2$	4.95E-04	–	2.60E-03	–	4.58E-05	–	6.52E-05	–
$20^2 \times 2$	2.44E-05	4.34	1.74E-04	3.90	2.68E-07	7.42	4.21E-07	7.27
$40^2 \times 2$	2.01E-06	3.60	1.11E-05	3.97	3.17E-09	6.40	6.70E-09	5.97
	Inexact postprocessing, $Q0 = 45, Q1 = 44$				Consistent postprocessing, CI			
$10^2 \times 2$	4.54E-05	–	6.42E-05	–	4.54E-05	–	6.42E-05	–
$20^2 \times 2$	2.44E-07	7.54	3.51E-07	7.51	2.44E-07	7.54	3.51E-07	7.51
$40^2 \times 2$	1.41E-09	7.44	2.65E-09	7.05	1.41E-09	7.44	2.65E-09	7.05
$\mathbb{P}^4$								
	Before postprocessing				Inexact postprocessing, $Q0 = 8, Q1 = 7$			
$10^2 \times 2$	3.62E-05	–	2.05E-04	–	4.02E-06	–	5.69E-06	–
$20^2 \times 2$	1.14E-06	4.99	6.47E-06	4.99	5.81E-09	9.44	8.27E-09	9.43
$40^2 \times 2$	3.56E-08	5.00	2.00E-07	5.02	2.30E-10	4.66	3.27E-10	4.66
	Inexact postprocessing, $Q0 = 56, Q1 = 55$				Consistent postprocessing, CI			
$10^2 \times 2$	4.02E-06	–	5.68E-06	–	4.02E-06	–	5.68E-06	–
$20^2 \times 2$	5.84E-09	9.43	9.35E-09	9.25	5.84E-09	9.43	9.35E-09	9.25
$40^2 \times 2$	2.31E-10	4.66	4.74E-10	4.30	2.31E-10	4.66	4.74E-10	4.30



(a) Smoothly varying mesh (b) Criss-cross mesh

FIG. 4.

with initial condition $u(0, x, y) = \sin(2\pi(x + y))$.

Table 1 shows the errors when postprocessing using inexact quadrature that respects only the DG-element breaks and postprocessing on a mesh that respects both

TABLE 2

Errors for the linear constant coefficient advection equation using $\mathbb{P}^2$, $\mathbb{P}^3$, and $\mathbb{P}^4$ polynomials for the smoothly varying triangular mesh. Before postprocessing and after postprocessing on the DG mesh, where Q_0 represents GLL points on one direction and Q_1 represents GRL points on the other direction, CI indicates consistent integration.

$\mathbb{P}^2$								
Mesh	L^2 error	Order	L^∞ error	Order	L^2 error	Order	L^∞ error	Order
	Before postprocessing				Inexact postprocessing, $Q0 = 5, Q1 = 4$			
$10^2 \times 2$	1.11E-02	–	9.15E-02	–	6.32E-03	–	1.21E-02	–
$20^2 \times 2$	1.37E-03	3.01	1.31E-02	2.80	5.97E-04	3.40	3.75E-03	1.69
$40^2 \times 2$	1.71E-04	3.00	1.66E-03	2.98	2.53E-04	1.24	1.60E-03	1.23
	Inexact postprocessing, $Q0 = 34, Q1 = 33$				Consistent postprocessing, CI			
$10^2 \times 2$	6.17E-03	–	1.14E-02	–	6.17E-03	–	1.14E-02	–
$20^2 \times 2$	1.95E-04	4.98	5.53E-04	4.37	1.96E-04	4.98	5.49E-04	4.38
$40^2 \times 2$	7.02-E06	4.80	6.04E-05	3.19	7.08E-06	4.78	6.26E-05	3.14
$\mathbb{P}^3$								
	Before postprocessing				Inexact postprocessing, $Q0 = 6, Q1 = 6$			
$10^2 \times 2$	1.22E-03	–	1.05E-02	–	3.81E-04	–	9.88E-04	–
$20^2 \times 2$	7.82E-05	3.96	7.49E-04	3.81	5.00-E05	2.93	2.64E-04	1.90
$40^2 \times 2$	4.97E-06	3.98	4.75E-05	3.98	2.37e-05	1.08	1.07E-04	1.30
	Inexact postprocessing, $Q0 = 45, Q1 = 44$				Consistent postprocessing, CI			
$10^2 \times 2$	3.45E-04	–	9.37E-04	–	3.45E-04	–	9.37E-04	–
$20^2 \times 2$	1.86E-06	7.54	4.83E-06	7.60	1.86E-06	7.54	4.83E-06	7.60
$40^2 \times 2$	1.24E-08	7.23	1.02E-07	5.57	1.24E-08	7.23	1.02E-07	5.57
$\mathbb{P}^4$								
	Before postprocessing				Inexact postprocessing, $Q0 = 8, Q1 = 7$			
$10^2 \times 2$	1.11E-04	–	1.34E-03	–	7.31E-05	–	2.13E-04	–
$20^2 \times 2$	3.66E-06	4.92	4.73E-05	4.82	2.83E-06	4.69	1.63E-05	3.71
$40^2 \times 2$	1.17E-07	4.97	1.52E-06	4.96	1.87E-06	0.60	1.38E-05	0.24
	Inexact postprocessing, $Q0 = 56, Q1 = 55$				Consistent postprocessing, CI			
$10^2 \times 2$	7.25E-05	–	2.13E-04	–	7.25E-05	–	2.13E-04	–
$20^2 \times 2$	9.55E-08	9.57	2.76E-07	9.59	9.55E-08	9.57	2.76E-07	9.59
$40^2 \times 2$	3.70E-09	4.69	6.28E-09	5.46	3.70E-09	4.69	6.28E-09	5.46

kernel breaks and DG-element breaks for a uniformly structured triangular mesh. For this case, we clearly see improvements from $k+1$ to $2k+1$ in the order of the errors. Furthermore, we see a decrease in the magnitude of the errors. We note that the errors for inexact postprocessing on the DG mesh are represented for two different sets of quadrature points. The first two sets of errors are for inexact integration, where only the DG-element breaks are respected. The second set of errors for inexact quadrature demonstrates our attempt to overcome the numerical crimes with increased quadrature points. These sets of quadrature points are designated "inexact postprocessing," and the number of quadrature points are given in all of the tables. We see that even in the first set of quadrature points, the results of the DG mesh are similar to the results of the consistent-integration (CI) approach. In addition, increasing the number of quadrature points results in decreased error and improved order of convergence. The final set of errors is what is required to calculate the integrals exactly when applying the postprocessor on the CI mesh. In all of the tables, we designate this "consistent postprocessing." We see that for this set of errors, we obtain the desired order accuracy as given by the theory in (2.1), except for $\mathbb{P}^4$ and the finest mesh, where we

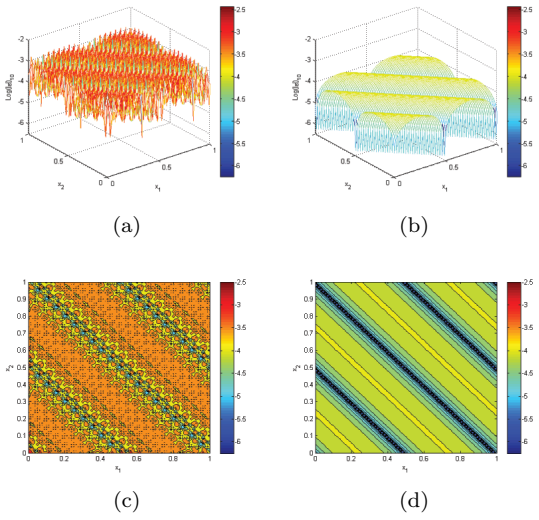


FIG. 5. *Three- and two-dimensional view of pointwise errors in logarithmic scale for constant coefficient advection equation when a* $\mathbb{P}^2$ *DG method is used over a structured (uniform) triangular mesh.* (a) *and* (c) *demonstrate the initial DG approximation errors;* (b) *and* (d) *represent the errors after the application of the postprocessor on the CI mesh. Filled contour plots have been used to visualize the data for* (c) *and* (d). *The SIAC filter works to reduce the oscillations in the error.*

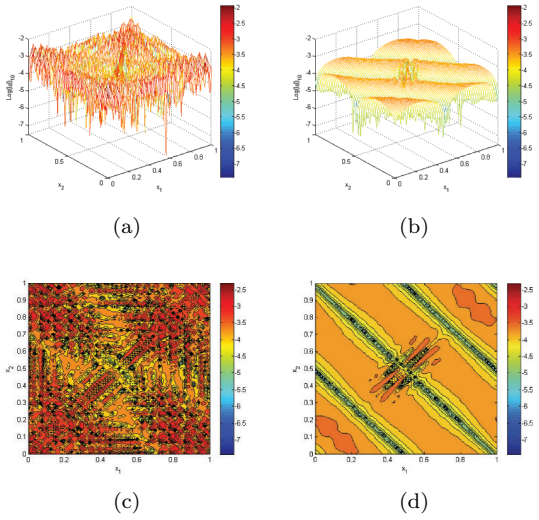


FIG. 6. *Three- and two-dimensional view of pointwise errors in logarithmic scale for constant coefficient advection equation when a* $\mathbb{P}^2$ *DG method is used over a smoothly varying triangular mesh.* (a) *and* (c) *demonstrate the initial DG approximation errors and* (b) *and* (d) *represent the errors after the application of the postprocessor on the CI mesh. We can clearly see how the SIAC filter reduces the oscillations in the error.*

expect that roundoff errors have begun to dominate.

The errors for the smoothly varying triangular mesh shown in Figure 4 are provided in Table 2. We note that technically the theory does not cover this case. We see that errors have decreased and the order of convergence in the L^2-norm are generally

TABLE 3

Errors for constant coefficient advection equation over a uniformly structured criss-cross mesh using $\mathbb{P}^2$, $\mathbb{P}^3$, and $\mathbb{P}^4$ polynomials. Before postprocessing and after postprocessing on the CI and DG meshes.

$\mathbb{P}^2$								
Mesh	L^2 error	Order	L^∞ error	Order	L^2 error	Order	L^∞ error	Order
	Before postprocessing				Inexact postprocessing, $Q0 = 5, Q1 = 4$			
$10^2 \times 2$	2.76E-03	–	1.78E-02	–	1.14E-03	–	3.61E-03	–
$20^2 \times 2$	3.47E-04	2.99	2.31E-03	2.94	6.74E-04	0.75	2.72E-03	0.40
$40^2 \times 2$	4.35E-05	2.99	3.03E-04	2.93	6.76E-04	-0.004	2.83E-03	-0.05
	Inexact postprocessing, $Q0 = 34, Q1 = 33$				Consistent postprocessing, CI			
$10^2 \times 2$	8.36E-04	–	1.30E-03	–	8.36E-04	–	1.30E-03	–
$20^2 \times 2$	2.88E-05	4.86	5.55E-05	4.55	2.87E-05	4.86	5.55E-05	4.55
$40^2 \times 2$	2.43E-06	3.57	5.05E-06	3.46	2.36E-06	3.60	4.63E-06	3.58
$\mathbb{P}^3$								
	Before postprocessing				Inexact postprocessing, $Q0 = 6, Q1 = 6$			
$10^2 \times 2$	2.44E-04	–	1.59E-03	–	4.22E-05	–	1.12E-04	–
$20^2 \times 2$	1.54E-05	3.98	1.03E-04	3.94	2.61E-05	0.69	6.82E-05	0.72
$40^2 \times 2$	9.72E-07	3.98	6.44E-06	3.99	2.55E-05	0.03	7.12E-05	-0.06
	Inexact postprocessing, $Q0 = 65, Q1 = 64$				Consistent postprocessing, CI			
$10^2 \times 2$	3.57E-05	–	5.48E-05	–	3.57E-05	–	5.49E-05	–
$20^2 \times 2$	2.40E-07	7.22	4.99E-07	6.78	2.40E-07	7.22	4.99E-07	6.78
$40^2 \times 2$	9.30E-09	4.69	1.83E-08	4.78	9.97E-09	4.60	1.83E-08	4.78
$\mathbb{P}^4$								
	Before postprocessing				Inexact postprocessing, $Q0 = 8, Q1 = 7$			
$10^2 \times 2$	1.48E-05	–	1.12E-04	–	3.98E-06	–	6.36E-06	–
$20^2 \times 2$	4.71E-07	4.97	3.40E-06	5.04	3.18E-07	3.65	1.19E-06	2.42
$40^2 \times 2$	1.47E-08	5.00	1.07E-07	4.99	3.09E-07	0.04	1.23E-06	-0.04
	Inexact postprocessing, $Q0 = 56, Q1 = 55$				Consistent postprocessing, CI			
$10^2 \times 2$	3.98E-06	–	5.68E-06	–	3.98E-06	–	5.68E-06	–
$20^2 \times 2$	4.95E-09	9.64	8.08E-09	9.45	4.95E-09	9.64	8.08E-09	9.45
$40^2 \times 2$	1.84E-09	1.43	2.65E-09	1.61	1.84E-09	1.43	2.65E-09	1.61

better when postprocessing on the CI mesh and improve to approximately $2k+1$. For the smoothly varying structured triangular mesh, when using quartic polynomials and the finest mesh, the error values improved compared to the initial solution; however, the convergence rate decreased. Using the smoothly varying triangular mesh, more quadrature points are required to simulate the exact results. Furthermore, Figures 5 through 6 depict the pointwise errors in logarithmic scale when using quadratic polynomials over a $20^2 \times 2$ mesh. From these plots, we observe that the error has decreased for the postprocessed solution, and the postprocessor has filtered out the oscillations. We note that the error plots for the inexact integration give the same results as for the consistent integration; therefore, we neglect including these.

We conclude the examination of the linear advection equation (see (4.1)) by studying the application of the postprocessor to DG solutions on the uniformly structured criss-cross mesh shown in Figure 4(b). Tables 3 and 4 present the error values for the uniformly structured and smoothly varying criss-cross meshes after one period in time. The error plots for the uniform structured mesh are displayed in Figure 7. For quartic polynomials and the finest mesh, we see that the error values decreased as well as the convergence rate. In addition, in the smoothly varying mesh case, the L^2-error value

TABLE 4

Errors for constant coefficient advection equation over a criss-cross mesh using $\mathbb{P}^2$, $\mathbb{P}^3$, and $\mathbb{P}^4$ polynomials. Before postprocessing and after postprocessing on the CI and DG meshes. We have assumed a smoothly varying criss-cross mesh.

$\mathbb{P}^2$								
Mesh	L^2 error	Order	L^∞ error	Order	L^2 error	Order	L^∞ error	Order
	Before postprocessing				Inexact postprocessing, $Q0 = 5, Q1 = 4$			
$10^2 \times 2$	5.57E-03	–	5.12E-02	–	4.10E-03	–	1.57E-02	–
$20^2 \times 2$	7.25E-04	2.94	8.09E-03	2.66	2.91E-03	0.49	1.42E-02	0.14
$40^2 \times 2$	9.16E-05	2.98	1.17E-03	2.78	2.88E-03	0.01	1.46E-02	0.04
	Inexact postprocessing, $Q0 = 34, Q1 = 33$				Consistent postprocessing, CI			
$10^2 \times 2$	2.67E-03	–	6.52E-03	–	2.67E-03	–	6.52E-03	–
$20^2 \times 2$	1.32E-04	4.34	5.77E-04	3.50	1.31E-04	4.35	5.78E-04	3.50
$40^2 \times 2$	1.57E-05	3.07	8.99E-05	2.68	1.35E-05	3.28	9.00E-05	2.68
$\mathbb{P}^3$								
	Before postprocessing				Inexact postprocessing, $Q0 = 6, Q1 = 6$			
$10^2 \times 2$	6.51E-04	–	6.63E-03	–	4.01E-04	–	2.43E-03	–
$20^2 \times 2$	4.18E-05	3.96	5.34E-04	3.63	3.72E-04	0.10	2.96E-03	-0.28
$40^2 \times 2$	2.64E-06	3.98	3.15E-05	4.08	3.87E-04	-0.05	3.17E-03	-.09
	Inexact postprocessing, $Q0 = 45, Q1 = 44$				Consistent postprocessing, CI			
$10^2 \times 2$	3.12E-04	–	8.80E-04	–	3.11E-04	–	8.80E-04	–
$20^2 \times 2$	3.00E-06	6.70	2.14E-05	5.36	2.97E-06	6.71	2.14E-05	5.36
$40^2 \times 2$	2.38E-07	3.66	1.63E-06	3.71	2.20E-07	3.75	1.63E-06	3.71
$\mathbb{P}^4$								
	Before postprocessing				Inexact postprocessing, $Q0 = 8, Q1 = 7$			
$10^2 \times 2$	5.52E-05	–	7.03E-04	–	7.59E-05	–	2.29E-04	–
$20^2 \times 2$	1.77E-06	4.96	2.67E-05	4.71	3.59E-05	1.08	3.20E-04	-0.48
$40^2 \times 2$	5.62E-08	4.97	9.03E-07	4.88	3.69E-05	-0.03	3.60E-04	-0.16
	Inexact postprocessing, $Q0 = 56, Q1 = 55$				Consistent postprocessing, CI			
$10^2 \times 2$	7.23E-05	–	2.12E-04	–	7.23E-05	–	2.12E-04	–
$20^2 \times 2$	1.07E-07	9.40	3.24E-07	9.35	1.07E-07	9.40	3.24E-07	9.35
$40^2 \times 2$	2.58E-09	5.37	1.73E-08	4.23	2.35E-09	5.51	1.73E-08	4.23

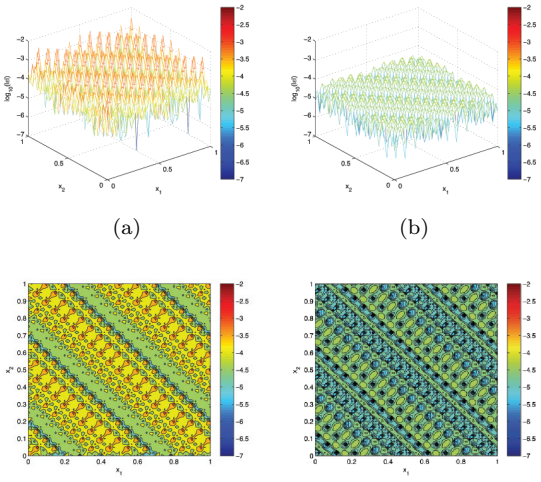
for the quartic polynomial and the coarsest mesh has increased, and this is because the kernel support covers the entire area of the mesh. In general, however, we see that the error values have decreased and the convergence rate has improved. Moreover, the inexact approach when using enough quadrature points yields results similar to the exact scheme. We emphasize that these results are not completely unexpected as the theory relies on a translation invariance property of the mesh.

4.2. Two-dimensional wave equation as a system. We conclude this section by demonstrating the application of the SIAC filter on the traditional second-order wave equation

$$\eta_{tt} - \eta_{xx} - \eta_{yy} = 0, \qquad (x, y) \in (0, 1) \times (0, 1), \qquad T = 6.28. \tag{4.2}$$

We rewrite (4.2) as a system of first-order linear equations shown below,

$$\begin{aligned} \eta_t + u_x + v_y &= 0, \\ u_t + \eta_x &= 0, \\ v_t + \eta_y &= 0, \end{aligned} \tag{4.3}$$



(a) (b)

(c) (d)

FIG. 7. *Three- and two-dimensional view of pointwise errors in logarithmic scale for constant coefficient advection equation when a* $\mathbb{P}^2$ *DG method is used over a uniformly structured criss-cross mesh.* (a) *and* (c) *demonstrate the initial DG approximation errors;* (b) *and* (d) *represents the errors after the application of the postprocessor on the CI mesh. We can clearly see how the SIAC filter reduces the oscillations in the error.*

with initial conditions

$$\begin{aligned} \eta(x,y,0) &= 0.01 \times (\sin(2\pi x) + \sin(2\pi y)), \\ u(x,y,0) &= 0.01 \times (\sin(2\pi x)), \\ v(x,y,0) &= 0.01 \times (\sin(2\pi y)), \end{aligned} \tag{4.4}$$

and 2π periodic boundary conditions in both directions. In Tables 5 and 6, we have provided the errors after one period in time for the η variable. As shown in Tables 5 and 6, the value of the error has decreased and the order of convergence has improved after the application of the postprocessor. For both the structured triangular mesh and the smoothly varying mesh, this is the expected improvement to $\mathcal{O}(h^{2k+1})$.

5. Conclusions. The most pressing issue in accuracy enhancement is to formulate a suitable technique for extracting extra accuracy from the DG solution solved over an unstructured triangular mesh. In this paper, we make a significant contribution to addressing this problem by demonstrating that there is a direct extension of the theory to structure triangular meshes as well as investigating the performance of the SIAC filter to two-dimensional hyperbolic equations solved over structured triangular meshes. The implementation of this method leads to formidable computational challenges because of the many (computationally intensive) integrations involved. For exact integration, the technique must respect both the DG mesh breaks as well as the B-spline kernel breaks. However, we have demonstrated that we can obtain optimal convergence even though we perform inconsistent integration and ignore the B-spline kernel breaks. We have given these elements both a uniform size structure and a

TABLE 5

Errors for the two-dimensional system using $\mathbb{P}^2$, $\mathbb{P}^3$, and $\mathbb{P}^4$ polynomials. Before postprocessing and after postprocessing on the CI and DG meshes. We have assumed a structured (uniform) triangular mesh.

$\mathbb{P}^2$								
Mesh	L^2 error	Order	L^∞ error	Order	L^2 error	Order	L^∞ error	Order
	Before postprocessing				Inexact postprocessing, $Q0 = 5, Q1 = 4$			
$10^2 \times 2$	1.24E-05	–	6.14E-05	–	2.35E-06	–	4.78E-06	–
$20^2 \times 2$	1.56E-06	2.99	7.97E-06	2.95	5.42E-08	5.44	1.22E-07	5.29
$40^2 \times 2$	1.95E-07	3.00	1.01E-06	2.98	3.85E-09	3.82	9.42E-09	3.70
	Inexact postprocessing, $Q0 = 34, Q1 = 33$				Consistent postprocessing, CI			
$10^2 \times 2$	2.33E-06	–	4.66E-06	–	2.33e-06	–	4.66e-06	–
$20^2 \times 2$	4.48E-08	5.70	9.03E-08	5.69	4.48e-08	5.70	9.03e-08	5.69
$40^2 \times 2$	9.17E-10	5.61	1.85E-09	5.61	9.17E-10	5.61	1.85E-09	5.61
$\mathbb{P}^3$								
	Before postprocessing				Inexact postprocessing, $Q0 = 6, Q1 = 6$			
$10^2 \times 2$	6.58E-07	–	3.36E-06	–	2.31E-07	–	4.70E-07	–
$20^2 \times 2$	4.12E-08	4.00	2.09E-07	4.01	1.17E-09	7.63	2.69E-09	7.45
$40^2 \times 2$	2.58E-09	4.00	1.32E-08	3.98	2.13E-11	5.78	5.51E-11	5.61
	Inexact postprocessing, $Q0 = 45, Q1 = 44$				Consistent postprocessing, CI			
$10^2 \times 2$	2.29E-07	–	4.58E-07	–	2.29E-07	–	3.36E-07	–
$20^2 \times 2$	9.80E-10	7.87	1.96E-09	7.87	9.80E-10	7.87	1.96E-09	7.42
$40^2 \times 2$	4.02E-12	7.93	8.05E-12	7.93	4.03E-12	7.93	1.37E-11	7.16
$\mathbb{P}^4$								
	Before postprocessing				Inexact postprocessing, $Q0 = 8, Q1 = 7$			
$10^2 \times 2$	2.44E-08	–	1.29E–07	–	2.80E-08	–	5.61E-08	–
$20^2 \times 2$	7.64E-10	5.00	4.16E-09	4.95	3.10E-11	9.82	6.22E-11	9.82
$40^2 \times 2$	2.40E-11	4.99	1.32E-10	4.98	8.07E-13	5.26	1.64E-12	5.25
	Inexact postprocessing, $Q0 = 56, Q1 = 55$				Consistent postprocessing, CI			
$10^2 \times 2$	2.80E-08	–	5.61E-08	–	2.80E-08	–	5.61E-08	–
$20^2 \times 2$	3.10E-11	9.82	6.19E-11	9.82	3.10E-11	9.82	6.19E-11	9.82
$40^2 \times 2$	8.17E-13	5.24	1.78E-12	5.12	8.17E-13	5.24	1.78E-12	5.24

smoothly varying structure. In both cases, we are able to improve the order of accuracy of the DG solution from $k+1$ to better then $2k$, and in some cases $2k+1$. The results obtained in this paper are exciting because previous accuracy enhancement on triangles has been restricted to $\mathcal{O}(h^{k+2})$. Furthermore, we have addressed one of the computational bottlenecks of the multidimensional implementation of this filter by reducing the number of integrations required.

Forming the filter for general unstructured triangular meshes is a challenging task and requires visiting the issue of the appropriate interpolation function to use for the convolution kernel. The proofs of the higher-order accuracy in the negative-order norm of the DG method do not rely on the mesh assumption; however, the accuracy-extracting capabilities of the kernel rely on the translation invariance of the mesh. Once this is accomplished, the postprocessor can then also be utilized for determining mesh adaptivity. These formidable issues constitute ongoing research.

TABLE 6
Errors for the two-dimensional system using $\mathbb{P}^2$, $\mathbb{P}^3$, and $\mathbb{P}^4$ polynomials for the smoothly varying triangular mesh. Before postprocessing and after postprocessing on the CI and DG meshes.

$\mathbb{P}^2$								
Mesh	L^2 error	Order	L^∞ error	Order	L^2 error	Order	L^∞ error	Order
	Before postprocessing				Inexact postprocessing, $Q0 = 5, Q1 = 4$			
$10^2 \times 2$	2.79E-05	–	1.81E-04	–	1.61E-05	–	9.08E-05	–
$20^2 \times 2$	3.57E-06	2.97	2.60E-05	2.80	8.14E-06	0.98	5.37E-05	0.76
$40^2 \times 2$	4.48E-07	2.99	3.37E-06	2.95	3.59E-06	1.18	2.59E-05	1.05
	Inexact postprocessing, $Q0 = 34, Q1 = 33$				Consistent postprocessing, CI			
$10^2 \times 2$	9.92E-06	–	3.14E-05	–	9.92E-06	–	3.14E-05	–
$20^2 \times 2$	1.90E-07	5.71	5.30E-07	5.89	1.92E-07	5.69	5.46E-07	5.85
$40^2 \times 2$	1.48E-08	3.68	9.27E-08	2.52	3.85E-09	5.64	2.72E-08	4.33
$\mathbb{P}^3$								
	Before postprocessing				Inexact postprocessing, $Q0 = 6, Q1 = 6$			
$10^2 \times 2$	1.70E-06	–	8.97E-06	–	2.67E-06	–	8.35E-06	–
$20^2 \times 2$	1.09E-07	3.96	6.38E-07	3.81	7.07E-07	1.92	4.20E-06	0.99
$40^2 \times 2$	6.85E-09	3.99	4.11E-08	3.96	3.55E-07	0.99	1.85E-06	1.18
	Inexact postprocessing, $Q0 = 45, Q1 = 44$				Consistent postprocessing, CI			
$10^2 \times 2$	1.95E-06	–	6.84E-06	–	1.95E-06	–	6.84E-06	–
$20^2 \times 2$	8.93E-09	7.77	2.84E-08	7.91	9.01E-09	7.76	2.84E-08	7.91
$40^2 \times 2$	1.11E-10	6.33	5.55E-10	5.68	3.69E-11	7.93	1.06E-10	8.07
$\mathbb{P}^4$								
	Before postprocessing				Inexact postprocessing, $Q0 = 8, Q1 = 7$			
$10^2 \times 2$	1.21E-07	–	8.29E-07	–	4.72E-07	–	1.70E-06	–
$20^2 \times 2$	3.87E-09	4.97	3.04E-08	4.77	4.34E-08	3.44	2.72E-07	2.64
$40^2 \times 2$	1.22E-10	4.99	9.88E-10	4.94	2.70E-08	1.61	2.25E-07	0.27
	Inexact postprocessing, $Q0 = 56, Q1 = 55$				Consistent postprocessing, CI			
$10^2 \times 2$	4.58E-07	–	1.72E-06	–	4.58E-07	–	1.72E-06	–
$20^2 \times 2$	5.58E-10	9.68	1.86E-09	9.85	5.58E-10	9.68	1.86E-09	9.85
$40^2 \times 2$	7.67E-13	9.51	2.98E-12	9.29	5.79E-13	9.91	1.77E-12	10.04

Acknowledgments. The authors thank Dr. Claes Eskilsson of Chalmers University of Technology in Gothenburg (Sweden), Dr. Spencer Sherwin of Imperial College London (United Kingdom), and all the Nektar++ developers (www.nektar.info) for helpful discussions, remarks, and support.

REFERENCES

[1] J. H. BRAMBLE AND A. H. SCHATZ, *Higher order local accuracy by averaging in the finite element method*, Math. Comp., 31 (1977), pp. 94–111.
[2] B. COCKBURN, *Discontinuous Galerkin methods for convection-dominated problems*, in High-Order Methods for Computational Physics, Lect. Notes Comput. Sci. Eng. 9, Springer, Berlin, 1999, pp. 69–224.
[3] B. COCKBURN, M. LUSKIN, C.-W. SHU, AND E. SÜLI, *Post-processing of Galerkin methods for hyperbolic problems*, in Proceedings of the International Symposium on Discontinuous Galerkin Methods, Lect. Notes Comput. Sci. Eng. 11, Springer, Berlin, 2000, pp. 291–300.
[4] B. COCKBURN, M. LUSKIN, C.-W. SHU, AND E. SÜLI, *Enhanced accuracy by post-processing for finite element methods for hyperbolic equations*, Math. Comp., 72 (2003), pp. 577–606.
[5] S. CURTIS, R. M. KIRBY, J. K. RYAN, AND C.-W. SHU, *Postprocessing for the discontinuous Galerkin method over nonuniform meshes*, SIAM J. Sci. Comput., 30 (2007), pp. 272–289.
[6] L. JI, Y. XU, AND J. K. RYAN, *Accuracy enhancement of the linear convection-diffusion equation in multiple dimensions*, Math. Comp., to appear.

[7] G. E. Karniadakis and S. J. Sherwin, *Spectral/hp Element Methods for Computational Fluid Dynamics*, 2nd ed., Oxford University Press, New York, 2005.

[8] H. Mirzaee, J. K. Ryan, and R. M. Kirby, *Quantification of errors introduced in the numerical approximation and implementation of smoothness-increasing accuracy conserving (SIAC) filtering of discontinuous Galerkin (DG) fields*, J. Sci. Comput., 45 (2010), pp. 447–470.

[9] J. K. Ryan, C.-W. Shu, and H. L. Atkins, *Extension of a postprocessing technique for the discontinuous Galerkin method for hyperbolic equations with application to an aeroacoustic problem*, SIAM J. Sci. Comput., 26 (2005), pp. 821–843.

[10] P. Van Slingerland, J. K. Ryan, and C. Vuik, *Position-depnedent smoothness-increasing accuracy-conserving (SIAC) filtering for improving discontinuous Galerkin solutions*, SIAM J. Sci. Comput., 33 (2011), pp. 802–825.

[11] M. Steffen, S. Curtis, R. M. Kirby, and J. K. Ryan, *Investigation of smoothness enhancing accuracy-conserving filters for improving streamline integration through discontinuous fields*, IEEE Trans. Vis. Comput. Graph., 14 (2008), pp. 680–692.