METHODS AND APPLICATIONS OF ANALYSIS. © 2003 International Press
Vol. 10, No. 2, pp. 295–308, June 2003 008

ON A ONE-SIDED POST-PROCESSING TECHNIQUE FOR THE DISCONTINUOUS GALERKIN METHODS [*]

JENNIFER RYAN[†] AND CHI-WANG SHU[‡]

Dedicated to Professor Stanley Osher on the occasion of his 60th birthday

Abstract. In this paper we study a class of one-sided post-processing techniques to enhance the accuracy of the discontinuous Galerkin methods. The applications considered in this paper are linear hyperbolic equations, however the technique can be used for the solution to a discontinuous Galerkin method solving other types of partial differential equations, or more general approximations, as long as there is a higher order negative norm error estimate for the numerical solution. The advantage of the one-sided post-processing is that it uses information only from one side, hence it can be applied up to domain boundaries, a discontinuity in the solution, or an interface of different mesh sizes. This technique allows us to obtain an improvement in the order of accuracy from $k+1$ of the discontinuous Galerkin method to $2k+1$ of the post-processed solution, using piecewise polynomials of degree k, throughout the entire domain and not just away from the boundaries, discontinuities, or interfaces of different mesh sizes.

1. Introduction. In this paper we study a class of one-sided post-processing techniques to enhance the accuracy of the discontinuous Galerkin methods. This is a modification of the local post-processing technique originally developed by Bramble and Schatz [1] in the context of continuous finite element methods for elliptic problems, and later by Cockburn, Luskin, Shu and Süli [6, 7] and by Ryan, Shu and Atkins [15] in the context of the discontinuous Galerkin methods for linear hyperbolic equations. Two key ingredients of this post-processing technique are a negative norm estimate for the numerical solution, which should be of higher order than the L^2 error estimate, and a local translation invariance of the mesh within the support of the local post-processor. The technique then allows the recovery of the L^2 error of the post-processed solution to the order of accuracy in the negative norm estimate. The main advantages of this technique, compared to other post-processing techniques, include its local feature, hence its efficiency and its easiness in the parallel implementation framework, and its effectiveness in almost doubling the order of accuracy rather than increasing the order of accuracy by one or two.

The original local post-processor in [1, 6, 7, 15] is based on a symmetric local stencil, using the information in about $2k$ neighboring elements to either side of the element being post-processed, for $\mathbb{P}^k$ (piecewise polynomials of degree up to k) elements. The mesh must be uniform within this local stencil, and the post-processed solution is $(2k+1)$-th order accurate in the L^2 norm instead of the usual $(k+1)$-th order accuracy before post-processing, for the discontinuous Galerkin method applied to linear hyperbolic equations, which maintains a $(2k+1)$-th order accuracy in the negative k norm of the numerical solution.

The symmetric nature of the local stencil prevents the application of the post-processor to the following situations: near a boundary of a computational domain; near a discontinuity in the solution; and near an interface of meshes with different mesh sizes. We can see clearly in the numerical examples in [15], that the post-processor fails in all

[*]Received March 31, 2003; accepted for publication August 21, 2003.

[†]Current address: Oak Ridge National Lab, P.O. Box 2008, MS 6367, Oak Ridge, TN 37831-6367, USA (ryanjk@ornl.gov). Research supported by NASA Langley grant NGT-1-01037.

[‡]Division of Applied Mathematics, Brown University, Providence, Rhode Island 02912, USA (shu@dam.brown.edu). Research supported by ARO grant DAAD19-00-1-0405, NSF grant DMS-0207451 and NASA Langley grant NCC1-01035.

such cases, rendering the enhancement of the order of accuracy not uniform across the computational domain when any one of these situations exist.

In this paper we develop a one-sided local post-processor based on a one-sided biased or completely one-sided stencil. That is, the stencil uses more elements to the left than to the right of the element being post-processed, or in the extreme case it uses only elements to the left, for the left-sided post-processor. The right-sided post-processor is a mirror image of the left-sided post-processor. Under the same assumptions as before, namely the numerical solution has a negative norm estimate which is higher order than the L^2 error estimate, and the mesh is translation invariant (uniform) within the stencil of the post-processor, the same accuracy enhancement can be expected and is verified by numerical experiments. This technique thus allows us to obtain an improvement in the order of accuracy from $k+1$ of the discontinuous Galerkin method to $2k+1$ of the post-processed solution throughout the entire domain and not just away from the boundaries, discontinuities, or interfaces of different mesh sizes, for problems with negative error estimates to the order of $2k+1$, such as the discontinuous Galerkin method using $\mathbb{P}^k$ elements for linear hyperbolic equations with smooth solutions. The applications considered in this paper are mainly linear hyperbolic equations, however the technique can be used for the solution to a discontinuous Galerkin method solving other types of partial differential equations as long as there is a higher order negative norm error estimate for the numerical solution, for example the local discontinuous Galerkin methods for convection diffusion equations [11, 7] and for partial differential equations with even higher spatial derivatives [17].

We remark that the idea of one-sided post-processors or filters has been developed before in [2, 13] for spectral methods, and mentioned in [14] for finite difference methods, for similar purpose of enhancing accuracy up to a boundary or a discontinuity.

The details of the discontinuous Galerkin method that we will be using can be found in [9, 8, 5, 4, 10, 3, 12]. We use the third-order TVD Runge-Kutta method in time [16]. The outline of the paper is as follows. In section 2, we discuss the form of the one-sided post-processor. The numerical examples, mostly those given in [15], are then presented using this one-sided form of the post-processor in section 3 to demonstrate the accuracy enhancement capability of this technique. Although only one dimensional scalar examples are given in this paper, the one-sided post-processor is expected to work equally well for multi-dimensional linear hyperbolic systems; see [7, 15] for such examples using the symmetric post-processors.

2. The one-sided post-processor.

The structure of the post-processor we will be using was initially designed by Mock and Lax [14] and Bramble and Schatz [1]. A discussion of this technique for the discontinuous Galerkin method can be found in [6, 7, 15]. For the purposes of this paper, we introduce a one-sided technique to handle post-processing near a boundary, a discontinuity, or an interface of meshes with different mesh sizes.

The post-processed solution for the numerical solution $u_h(x)$ is of the form

$$u^*(x) = K_h * u_h(x)$$

where the symmetric post-processing kernel is given by

$$K^{2(k+1),k+1}(x) = \sum_{\gamma=-k}^{k} c_\gamma^{2(k+1),k+1} \psi^{(k+1)}(x-\gamma), \tag{2.1}$$

and $K_h = \frac{1}{h}K(\frac{x}{h})$ for a locally uniform mesh ($h = \triangle x_i,\ i = 1, \cdots, N$). Here $c_\gamma^{2(k+1),k+1}$

are constants and $\psi^{(k+1)}(x)$ are $(k+1)$-th order B-splines. Recall that the post-processor can be implemented by doing small matrix-vector multiplications, [15]. The matrix-vector format is found from evaluating

$$\begin{aligned} u^*(x) &= \frac{1}{h}\sum_{j=-2k}^{2k}\int_{I_{i+j}} K^{2(k+1),k+1}\left(\frac{y-x}{h}\right)\sum_{l=0}^{k} u_{i+j}^{(l)}\left(\frac{y-x_{i+j}}{h}\right)^l dy \\ &= \sum_{j=-2k}^{2k}\sum_{l=0}^{k} u_{i+j}^{(l)}\, C(j,l,k,x) \end{aligned} \tag{2.2}$$

where

$$C(j,l,k,x) = \frac{1}{h}\sum_{\gamma=-k}^{k} c_\gamma^{2(k+1),k+1}\int_{I_{i+j}} \psi^{(k+1)}\left(\frac{y-x}{h}-\gamma\right)\left(\frac{y-x_{i+j}}{h}\right)^l dy$$

is a polynomial of degree $2k+1$ on the cell $I_i = (x_i - \frac{h}{2}, x_i + \frac{h}{2})$. Notice that in this symmetric version the post-processed solution has a support of $2k$ cells on either side for a total support of $4k+1$ cells. In order to perform a one-sided version of this, we simply extend this support biased to one-side. Thus, to perform a purely left-sided post-processing, the post-processed solution would be of the form

$$\begin{aligned} &u^*(x) \\ &= \sum_{j=-4k}^{0}\sum_{l=0}^{k} u_{i+j}^{(l)}\frac{1}{h}\sum_{\gamma=-2k-1}^{-1} c_\gamma^{2(k+1),k+1}\int_{I_{i+j}} \psi^{(k+1)}\left(\frac{y-x}{h}-\gamma\right)\left(\frac{y-x_{i+j}}{h}\right)^l dy. \end{aligned} \tag{2.3}$$

Although we do not provide the details here, we note that it is only necessary to find the new coefficients, $c_\gamma^{2(k+1),k+1}$, in the post-processing kernel - the evaluation of the integral does not change.

As shown in [7, 15], the coefficients used in the kernel can be found easily by implementing the property that $K * p = p$. In the case of the symmetric post-processing this property holds for polynomials p up to degree $2k+1$. For the one sided post-processor, we can only require this property to hold for polynomials up to degree $2k$. As an example, we can look at the left-sided post-processor used for a linear polynomial approximation (which gives second order accuracy). The left-sided post-processed solution is given by

$$u^*(x) = \sum_{j=-4}^{0}\sum_{l=0}^{1} u_{i+j}^{(l)}\frac{1}{h}\sum_{\gamma=-3}^{-1} c_\gamma^{4,2}\int_{I_{i+j}} \psi^{(2)}\left(\frac{y-x}{h}-\gamma\right)\left(\frac{y-x_{i+j}}{h}\right)^l dy, \tag{2.4}$$

where the $c_\gamma^{4,2}$ are found from $K * p = p$ for $p = 1, x, x^2$. This gives the coefficients

$$\begin{bmatrix} c_{-3} \\ c_{-2} \\ c_{-1} \end{bmatrix} = \begin{bmatrix} \frac{11}{12} \\ -\frac{17}{6} \\ \frac{35}{12} \end{bmatrix}.$$

We proceed similarly for partially left sided post-processing. The right-sided post-processing is a mirror image of the left-sided post-processing.

3. Numerical Examples.

3.1. On the approximation level. To confirm the properties of one-sided post-processing, we take as an example the L^2-projection of $u(x) = \sin(x)$ onto the piecewise $\mathbb{P}^1$- and $\mathbb{P}^2$-polynomials for $x \in (0, \frac{7\pi}{2})$ and calculate the errors throughout the domain both before and after post-processing. The one-sided biased or completely one-sided post-processor is applied near the domain boundaries when it is necessary, and the usual symmetric post-processing is applied in the interior of the domain whenever possible.

In Table 3.1 we can see that the post-processor does indeed give us the designed $(2k+1)$-th order accuracy. In fact, the L^2 errors are even half an order higher than the designed $(2k+1)$-th order, due to the fact that in the interior the symmetric post-processor is $(2k+2)$-th order accurate on the approximation level. However, looking at the L^∞ errors, we can see that unless the mesh is sufficiently refined it is not advantageous to apply the one-sided post-processor. This is a clear indication that the constant in front of the leading error term h^{2k+1} is much larger for the one-sided post-processor than for the symmetric post-processor. In Figure 3.1 it is illustrated that near the boundaries, where the one-sided post-processor is applied, the error is larger than in the interior, where the symmetric post-processor is applied. We also see that unless the mesh is sufficiently refined, it is not advantageous to apply the one-sided post-processor at the domain boundaries.

Next we look at the accuracy enhancement of the post-processor for the derivatives of the numerical solution. In [15], it is verified that the symmetric post-processor can improve the order of accuracy for the r-th derivative from $k+1$-r to $2k+2$-r. This is also the case for the one-sided post-processor, see Table 3.1 and Figure 3.1. We can see that, for the derivatives (especially for the second derivative), it is advantageous to do the one-sided post-processing even for coarse meshes.

3.2. Domains with different mesh sizes. In [15] we presented the example of a domain with two different mesh sizes. We explored the case of the linear hyperbolic equation $u_t + u_x = 0$ with $u(x, 0) = \sin(3x)$ for $x \in (0, 2\pi)$. The left half of the mesh contains two-thirds of the elements and the right-half of the domain contains one-third of the elements as shown in Figure 3.2. The problem is solved over the entire domain at a final time of $T = 12.5$. In the case of using the symmetric version of the post-processor, we found that when we evaluated the solution away from the mesh interface we did obtain the expected $(2k+1)$-th order accuracy in the post-processed solution. However, in the region near the mesh interface, there was contamination in the post-processed solution due to the post-processor taking information from both sides of the interface, violating the local translation invariance assumption in the post-processing stencil.

Here, we present the results of implementing the one-sided biased or completely one-sided form of the post-processor near the mesh interface and domain boundary when necessary. The symmetric version is used in the interior of the two meshes whenever possible. We evaluate the error throughout the entire domain, and not in the individual mesh regions as done in [15]. In Table 3.2, we see that for piecewise $\mathbb{P}^1$- and $\mathbb{P}^2$-polynomials we do indeed obtain the improvement in the order of accuracy to $2k+1$ throughout the entire domain, although the mesh has to be sufficiently refined in order for the one-sided post-processor to show an effect of reducing the errors near the interface. In Figure 3.3 we can compare the post-processed solution with the original solution. We can see an apparent improvement in accuracy after the post-processing. Comparing with the figures in [15] using the symmetric post-processor, we can see that the one-sided post-processor successfully eliminates a region of O(1) errors near the interfaces of two different mesh

TABLE 3.1
Errors for the L^2-projection of $u(x) = \sin(x)$ onto the piecewise $\mathbb{P}^1$- and $\mathbb{P}^2$-polynomials for $x \in (0, \frac{7\pi}{2})$.

	Before Post-Processing				After Post-Processing			
mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
Errors for $u(x)$								
$\mathbb{P}^1$								
20	7.93E-03	—	2.01E-02	—	2.95E-02	—	1.29E-01	—
40	1.99E-03	2.00	5.06E-03	1.99	2.71E-03	3.44	1.54E-02	3.07
80	4.98E-04	2.00	1.27E-03	2.00	2.42E-04	3.49	2.10E-03	2.87
160	1.24E-04	2.00	3.17E-04	2.00	2.15E-05	3.50	2.69E-04	2.97
320	3.11E-05	2.00	7.91E-05	2.00	1.90E-06	3.50	3.37E-05	2.99
$\mathbb{P}^2$								
20	3.69E-04	—	8.66E-04	—	5.82E-03	—	2.57E-02	—
40	4.62E-05	3.00	1.09E-04	3.00	1.37E-04	5.41	7.15E-04	5.17
80	5.78E-06	3.00	1.36E-05	3.00	3.08E-06	5.48	2.54E-05	4.81
160	7.23E-07	3.00	1.70E-06	3.00	6.84E-08	5.49	8.40E-07	4.92
Errors for $du(x)/dx$								
$\mathbb{P}^1$								
20	1.11E-01	—	2.55E-01	—	2.95E-02	—	1.30E-01	—
40	5.60E-02	1.00	1.28E-01	0.99	2.72E-03	3.44	1.54E-02	3.08
80	2.80E-02	1.00	6.41E-02	1.00	2.43E-04	3.49	2.10E-03	2.87
160	1.40E-02	1.00	3.20E-02	1.00	2.15E-05	3.50	2.69E-04	2.96
320	7.01E-03	1.00	1.60E-02	1.00	1.90E-06	3.50	3.40E-05	2.98
$\mathbb{P}^2$								
20	8.69E-03	—	2.52E-02	—	5.82E-03	—	2.57E-02	—
40	2.18E-03	2.00	6.32E-03	1.99	1.37E-04	5.41	7.15E-04	5.17
80	5.45E-04	2.00	1.58E-03	2.00	3.08E-06	5.48	2.56E-05	4.80
160	1.36E-04	2.00	3.95E-04	2.00	6.84E-08	5.49	8.45E-07	4.92
Errors for $d^2u(x)/dx^2$								
$\mathbb{P}^1$								
20	N/A	—	N/A	—	3.06E-02	—	1.51E-01	—
40	N/A	—	N/A	—	3.37E-03	3.18	1.61E-02	3.23
80	N/A	—	N/A	—	5.54E-04	2.60	2.11E-03	2.93
160	N/A	—	N/A	—	1.26E-04	2.13	3.63E-04	2.54
320	N/A	—	N/A	—	3.12E-05	2.02	8.21E-05	2.15
$\mathbb{P}^2$								
20	1.12E-01	—	2.55E-01	—	5.92E-03	—	2.75E-02	—
40	5.61E-02	1.00	1.28E-01	0.99	1.44E-04	5.36	7.18E-04	5.26
80	2.80E-02	1.00	6.41E-02	1.00	3.86E-06	5.22	2.48E-05	4.85
160	1.40E-02	1.00	3.20E-02	1.00	1.57E-07	4.62	8.39E-07	4.89

sizes.

Next, we look at the accuracy enhancement of the post-processor for the derivatives of the numerical solution. The errors and orders of accuracy are listed in Table 3.3. Figure 3.4 plots the errors before and after post-processing for the first and second derivatives in the $\mathbb{P}^2$ case. Clearly, we obtain an improvement for the order of accuracy

TABLE 3.2

One dimensional discontinuous Galerkin approximations to the advection equation over a domain with two different mesh sizes as shown in Figure 3.2. Errors calculated over the entire domain.

	Before post-processing				After post-processing			
mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
				$\mathbb{P}^1$				
20	3.48E-01	—	5.40E-01	—	3.76E-01	—	5.81E-01	—
40	6.20E-02	2.49	1.07E-01	2.34	6.32E-02	2.57	1.09E-01	2.41
80	9.55E-03	2.70	1.89E-02	2.50	1.02E-02	2.63	2.93E-02	1.89
160	1.46E-03	2.71	3.22E-03	2.55	1.25E-03	3.03	4.26E-03	2.78
320	2.82E-04	2.38	9.21E-04	1.80	1.53E-04	3.03	5.80E-04	2.88
				$\mathbb{P}^2$				
20	9.77E-03	—	2.44E-02	—	3.20E-01	—	1.40E-00	—
40	7.95E-04	3.62	3.55E-03	2.78	9.96E-03	5.01	5.88E-02	4.58
80	1.05E-04	2.92	5.08E-04	2.81	2.67E-04	5.22	2.08E-03	4.82
160	1.31E-05	3.01	6.37E-05	2.99	1.03E-05	4.69	8.52E-05	4.61
320	1.68E-06	2.96	8.19E-06	2.96	2.74E-07	5.24	3.00E-06	4.83

approximating the r-th derivative from k+1-r to $2k$+2-r, and the order of accuracy is uniform throughout the computational domain. Also, for derivatives, the one-sided post-processor improves the accuracy even for coarse meshes.

3.3. Discontinuous coefficient hyperbolic equations. The second case we address using the one-sided post-processing technique is that of a linear hyperbolic equation with a discontinuous coefficient

$$u(x,t)_t + (a(x)u(x,t))_x = 0 \tag{3.1}$$

where $a(x)$ is piecewise constant,

$$a(x) = \begin{cases} 1, & x \in [-a,a]\backslash(-b,b), \\ \frac{1}{2}, & x \in (-b,b) \end{cases}$$

with $0 < b < a$ as given in [15].

The first example is one with two stationary shocks located at $x = \pm\frac{1}{2}$ for the exact solution. The initial condition is given by

$$u(x,0) = \begin{cases} -2\cos(4\pi x), & x \in (-\frac{1}{2},\frac{1}{2}), \\ \cos(2\pi x), & \text{otherwise in [-1,1]} \end{cases}$$

and extended periodically outside [-1,1]. Here $a = 1$ and $b = \frac{1}{2}$, and the boundary condition is 2-periodic. In [15], when a symmetric post-processor was used, the errors were calculated 0.4 away from the stationary shocks at a final time $T = 12.5$. Here we implement the one-sided version of the post-processing technique at the boundaries and the regions near the stationary shocks whenever necessary. In Table 3.4 we can see that the post-processing gives us ($2k$+1)-th order accuracy throughout the entire domain, including the regions containing the discontinuities. The errors for the $\mathbb{P}^2$-polynomial case are also plotted in Figure 3.5 in logarithmic scale both before and after implementing

Table 3.3

One dimensional discontinuous Galerkin approximations to the advection equation over a domain with two different mesh sizes as shown in Figure 3.2. Errors in the first and second derivatives calculated over the entire domain.

	Before post-processing				After post-processing			
mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
First Derivative								
$\mathbb{P}^1$								
20	1.16E-00	—	1.98E-00	—	9.57E-01	—	1.54E-00	—
40	4.02E-01	1.52	1.18E-00	0.75	2.03E-01	2.24	4.71E-01	1.71
80	1.95E-01	1.04	6.46E-01	0.87	2.68E-02	2.93	6.00E-02	2.97
160	9.78E-02	1.00	3.30E-01	0.97	3.31E-03	3.01	5.23E-03	3.52
320	4.93E-02	0.99	1.67E-01	0.98	4.27E-04	2.96	7.22E-04	2.86
$\mathbb{P}^2$								
20	1.32E-01	—	6.40E-01	—	1.35E-00	—	4.30E-00	—
40	3.49E-02	1.92	1.73E-01	1.88	4.64E-02	4.86	2.03E-01	4.40
80	9.42E-03	1.89	4.75E-02	1.87	1.40E-03	5.05	8.26E-03	4.62
160	2.36E-03	2.00	1.19E-02	2.00	1.77E-05	6.31	1.91E-04	5.43
320	6.00E-04	1.98	3.03E-03	1.97	5.27E-07	5.07	6.04E-06	4.98
Second Derivative								
$\mathbb{P}^1$								
20	N/A	—	N/A	—	3.41E-00	—	5.48E-00	—
40	N/A	—	N/A	—	5.77E-01	2.56	9.99E-01	2.46
80	N/A	—	N/A	—	9.86E-02	2.55	2.66E-01	1.91
160	N/A	—	N/A	—	1.43E-02	2.78	4.71E-02	2.50
320	N/A	—	N/A	—	2.63E-03	2.45	8.28E-03	2.51
$\mathbb{P}^2$								
20	2.04E-00	—	6.37E-00	—	2.99E-00	—	1.30E+01	—
40	1.04E-00	0.97	3.39E-00	0.91	9.06E-02	5.04	5.22E-01	4.64
80	5.38E-01	0.96	1.77E-00	0.94	2.56E-03	5.15	1.80E-02	4.85
160	2.69E-01	1.00	8.86E-01	1.00	1.16E-04	4.46	1.30E-03	3.79
320	1.36E-01	0.99	4.49E-01	0.98	2.10E-05	2.47	2.45E-04	2.41

the one-sided form of the post-processor. From Figure 3.5 we can see that the one-sided post-processor works even near the shock regions.

We next consider the more complex situation when there are two moving shocks in addition to the two stationary shocks. The equation (3.1) is with $a = 2$, $b = 1$ and the initial condition is given by

$$u(x,0) = \begin{cases} \cos(\frac{\pi}{2}x), & x \in [-2,2]\backslash(-1,1), \\ \frac{2}{3}\sin(\pi x) & x \in (-1,1). \end{cases}$$

and extended periodically outside [-2,2]. We use a 4-periodic boundary condition and compute up to the final time $T = 1$, before the shocks cross each other. The two stationary shocks are located at cell interfaces, whereas the two moving shocks are located inside the cells. In this example, the discontinuous Galerkin method has no problem handling the stationary shocks, as they are at cell interfaces. However, the moving shocks are located inside the cells and will degenerate the accuracy nearby. In this case, the errors for the symmetric version of the post-processor were calculated 1.1 away from

Table 3.4

Errors for the discontinuous method applied to the linear hyperbolic equation with discontinuous coefficient with two stationary shocks. Errors calculated over the entire domain.

	Before post-processing				After post-processing			
mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
				$\mathbb{P}^1$				
20	8.55E-01	—	1.56E+01	—	8.17E-01	—	1.55E+01	—
40	1.93E-01	2.15	3.80E-01	2.04	1.80E-01	2.18	3.50E-01	2.15
80	2.72E-02	2.83	5.84E-02	2.70	2.58E-02	2.80	4.90E-02	2.84
160	3.69E-03	2.88	8.84E-03	2.72	3.34E-03	2.95	6.21E-03	2.98
320	5.67E-04	2.70	1.46E-03	2.60	4.21E-04	2.99	7.87E-04	2.98
				$\mathbb{P}^2$				
40	1.45E-03	—	5.65E-03	—	2.04E-02	—	9.85E-02	—
80	1.54E-04	3.24	7.29E-04	2.95	4.48E-04	5.51	3.04E-03	5.02
160	1.90E-05	3.02	9.22E-05	2.98	5.87E-06	6.25	5.81E-05	5.71
320	2.37E-06	3.00	1.16E-05	3.00	7.26E-08	6.34	9.72E-07	5.90
640	2.96E-07	3.00	1.44E-06	3.00	1.71E-09	5.41	1.61E-08	5.91

Table 3.5

Errors for the discontinuous method applied to the linear hyperbolic equation with a discontinuous coefficient with two stationary and two moving shocks. Errors calculated outside a radius of 0.4 of the moving shocks.

	Before post-processing				After post-processing			
mesh	L^2 error	order	L^∞ error	order	L^2 error	order	L^∞ error	order
				$\mathbb{P}^1$				
40	1.36E-03	—	8.88E-03	—	2.05E-03	—	1.17E-02	—
80	3.35E-04	2.02	2.31E-03	1.94	9.57E-05	4.42	7.82E-04	3.91
160	8.30E-05	2.01	5.87E-04	1.98	4.61E-06	4.38	4.66E-05	4.07
320	2.07E-05	2.01	1.48E-04	1.99	3.40E-07	3.76	2.49E-06	4.23
640	5.16E-06	2.00	3.71E-05	2.00	3.87E-08	3.14	1.32E-07	4.24
				$\mathbb{P}^2$				
40	3.79E-05	—	2.45E-04	—	1.90E-04	—	1.00E-03	—
80	4.77E-06	2.99	3.08E-05	2.99	2.44E-06	6.28	1.89E-05	5.73
160	5.98E-07	3.00	3.85E-06	3.00	2.80E-08	6.45	3.09E-07	5.93

the shocks in [15]. However, implementing the one-sided post-processor in the region of the discontinuities allows us to calculate the error in a larger region which includes the stationary shocks. Since the discontinuous Galerkin approximation itself does not give good information near the moving shocks, we do not expect the accuracy of the post-processor to improve upon this information, even in the case of implementing the one-sided post-processor. The errors contained in Table 3.5 show that outside a radius of 0.4 of the moving shocks, the one-sided post-processed solution improves the accuracy to $2k+1$. The errors for the $\mathbb{P}^2$ case are also plotted in Figure 3.6 in logarithmic scale both before and after post-processing.

4. Concluding Remarks.

We have presented a one-sided version of a local post-processing technique that enhances the accuracy of any approximation with high order negative norm error estimates, with application to the discontinuous Galerkin solution for linear hyperbolic equations. The one-sided nature of the post-processor allows us to enhance the accuracy throughout the entire domain and not just away from the boundaries, discontinuities, or interfaces of different mesh sizes. Numerical examples are given to verify such accuracy enhancement capability of the post-processor for the discontinuous Galerkin solution and its derivatives throughout the entire domain for multi-domain problems with different mesh sizes and discontinuous coefficient equations.

REFERENCES

[1] J.H. Bramble and A.H. Schatz, *Higher order local accuracy by averaging in the finite element method*, Mathematics of Computation, 31 (1977), pp. 94–111.

[2] W. Cai, D. Gottlieb and C.-W. Shu, *On one-sided filters for spectral Fourier approximations of discontinuous functions*, SIAM Journal on Numerical Analysis, 29 (1992), pp. 905–916.

[3] B. Cockburn, *Discontinuous Galerkin methods for convection-dominated problems*, in *High-Order Methods for Computational Physics*, T.J. Barth and H. Deconinck, editors, Lecture Notes in Computational Science and Engineering, volume 9, Springer, 1999, pp. 69–224.

[4] B. Cockburn, S. Hou and C.-W. Shu, *The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: the multidimensional case*, Mathematics of Computation, 54 (1990), pp. 545–581.

[5] B. Cockburn, S.-Y. Lin and C.-W. Shu, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: one dimensional systems*, Journal of Computational Physics, 84 (1989), pp. 90–113.

[6] B. Cockburn, M. Luskin, C.-W. Shu and E. Süli, *Post-processing of Galerkin methods for hyperbolic problems*, Proceedings of the International Symposium on Discontinuous Galerkin Methods, Newport, May 1999, G. Karniadakis, B. Cockburn and C.-W. Shu, editors, Lecture Notes in Computational Science and Engineering, Springer, 11 (1999), pp. 291–300.

[7] B. Cockburn, M. Luskin, C.-W. Shu and E. Süli, *Enhanced accuracy by post-processing for finite element methods for hyperbolic equations*, Mathematics of Computation, 72 (2003), pp. 577–606.

[8] B. Cockburn and C.-W. Shu, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: general framework*, Mathematics of Computation, 52 (1989), pp. 411–435.

[9] B. Cockburn and C.-W. Shu, *The Runge-Kutta local projection P^1-discontinuous-Galerkin finite element method for scalar conservation laws*, Mathematical Modelling and Numerical Analysis (M^2AN), 25 (1991), pp. 337–361.

[10] B. Cockburn and C.-W. Shu, *The Runge-Kutta discontinuous Galerkin method for conservation laws V: multidimensional systems*, Journal of Computational Physics, 141 (1998), pp. 199–224.

[11] B. Cockburn and C.-W. Shu, *The local discontinuous Galerkin method for time-dependent convection-diffusion systems*, SIAM Journal on Numerical Analysis, 35 (1998), pp. 2440–2463.

[12] B. Cockburn and C.-W. Shu, *Runge-Kutta Discontinuous Galerkin methods for convection-dominated problems*, Journal of Scientific Computing, 16 (2001), pp. 173–261.

[13] D. Gottlieb, C.-W. Shu, A. Solomonoff and H. Vandeven, *On the Gibbs phenomenon I: recovering exponential accuracy from the Fourier partial sum of a nonperiodic analytic function*, Journal of Computational and Applied Mathematics, 43 (1992), pp. 81–98.

[14] M.S. Mock and P.D. Lax, *The computation of discontinuous solutions of linear hyperbolic equations*, Communications on Pure and Applied Mathematics, 31 (1978), pp. 423–430.

[15] J. Ryan, C.-W. Shu and H.L. Atkins, *Extension of a post-processing technique for the discontinuous Galerkin method for hyperbolic equations with application to an aeroacoustic problem*, submitted to SIAM Journal on Scientific Computing.

[16] C.-W. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, Journal of Computational Physics, 77 (1988), pp. 439–471.

[17] J. Yan and C.-W. Shu, *Local discontinuous Galerkin methods for partial differential equations with higher order derivatives*, Journal of Scientific Computing, 17 (2002), pp. 27–47.

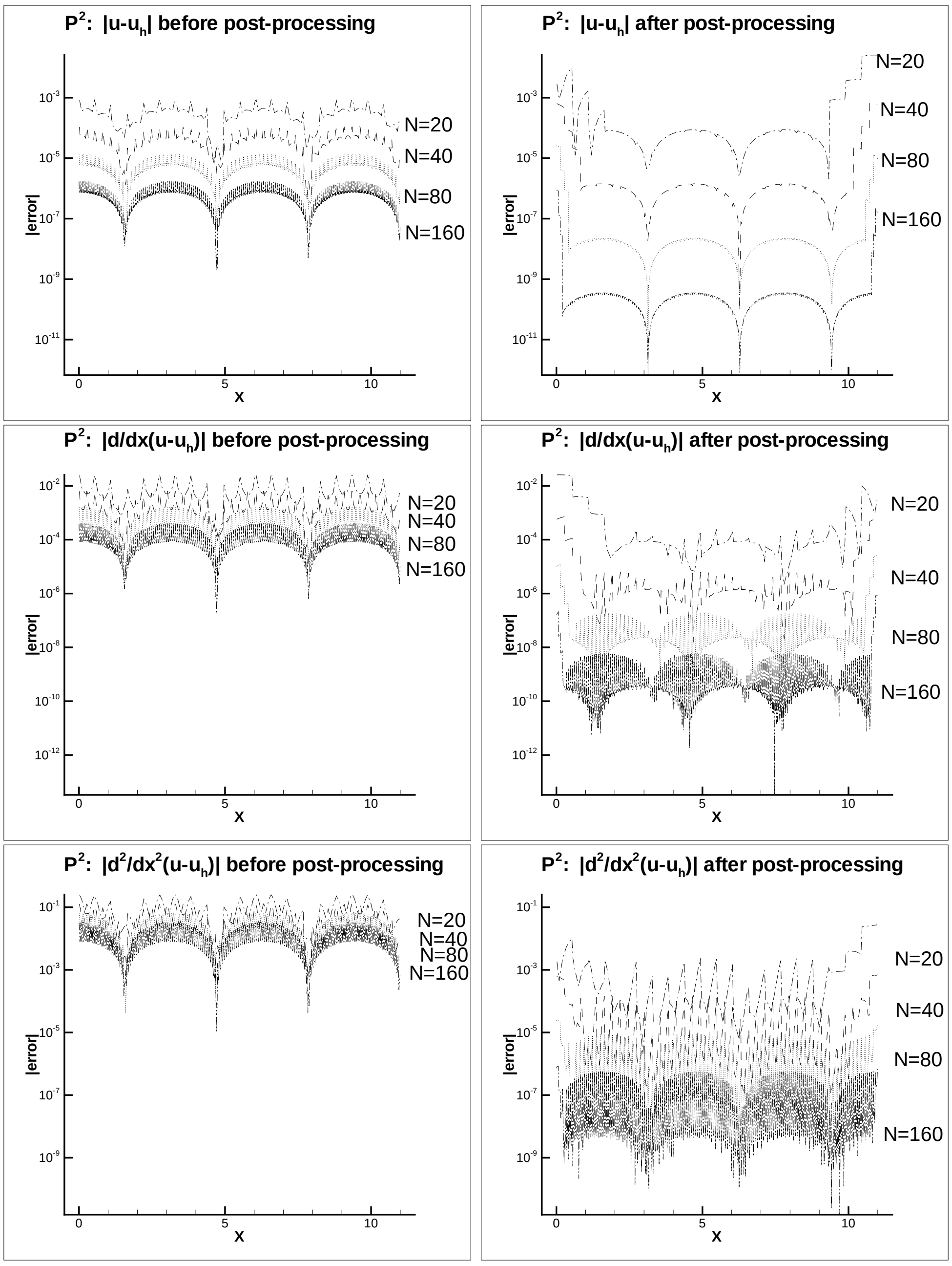

FIG. 3.1. *Pointwise errors in logarithmic scale when an L^2-projection onto the piecewise $\mathbb{P}^2$ polynomials is done. Left: before post-processing. Right: after post-processing. Top: for the function; middle: for the first derivative; bottom: for the second derivative.*

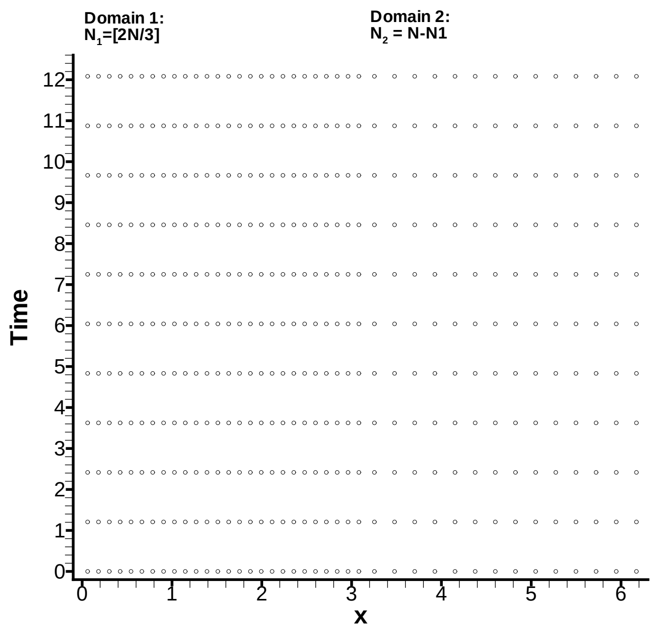

FIG. 3.2. *Mesh structure for solving the linear advection equation. The left half of the domain contains approximately two-thirds of the elements; the right contains approximately one-third of the elements.*

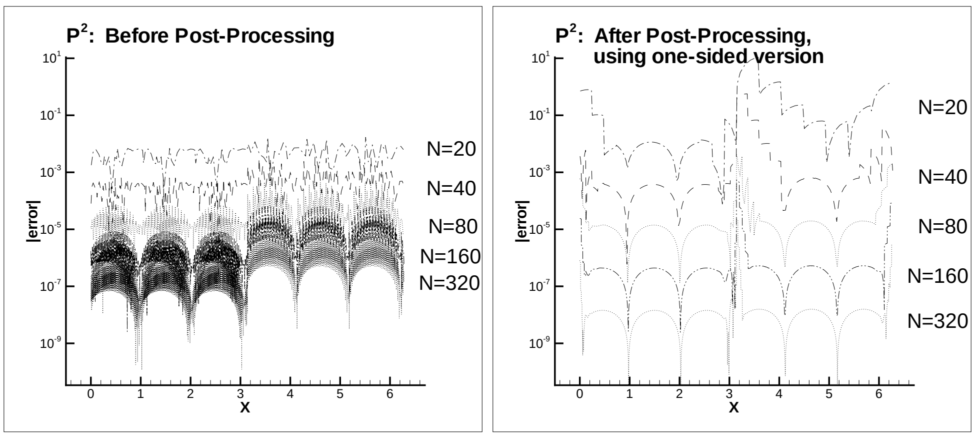

FIG. 3.3. *Pointwise errors in logarithmic scale when the $\mathbb{P}^2$ discontinuous Galerkin method is used to solve the linear advection equation over a domain with two different mesh sizes as shown in Figure 3.2. Left: before post-processing. Right: after post-processing. The graphs are scaled the same.*

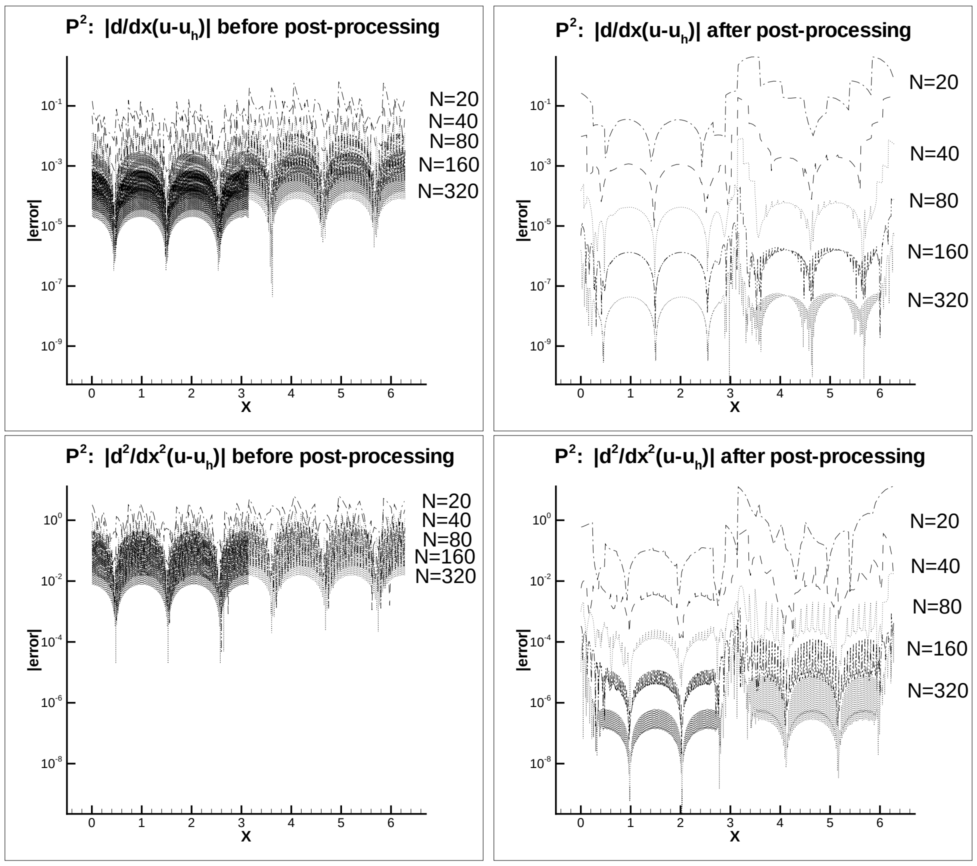

FIG. 3.4. *Pointwise errors in the first and second derivatives in logarithmic scale when the $\mathbb{P}^2$ discontinuous Galerkin method is used to solve the linear advection equation over a domain with two different mesh sizes as shown in Figure 3.2. Left: before post-processing. Right: after post-processing. Top: first derivative; bottom: second derivative.*

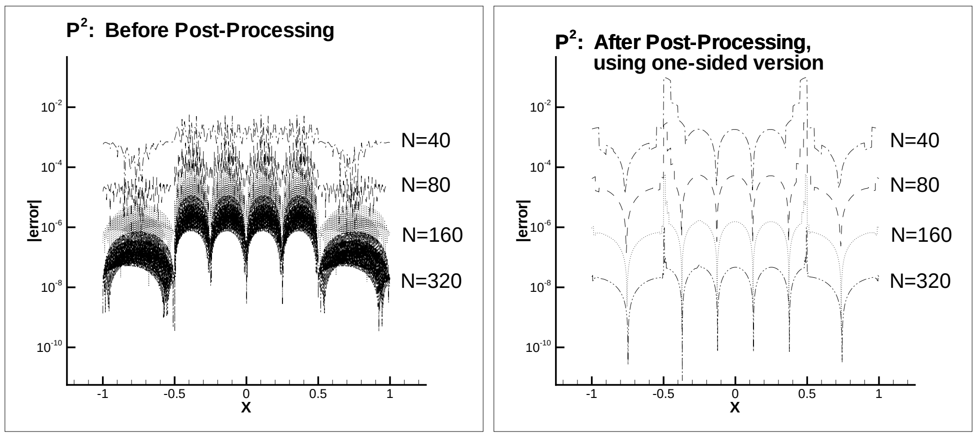

FIG. 3.5. *Pointwise errors in logarithmic scale when the $\mathbb{P}^2$ discontinuous Galerkin method is used to solve the discontinuous coefficient equation with two stationary shocks. Left: before post-processing. Right: after post-processing.*

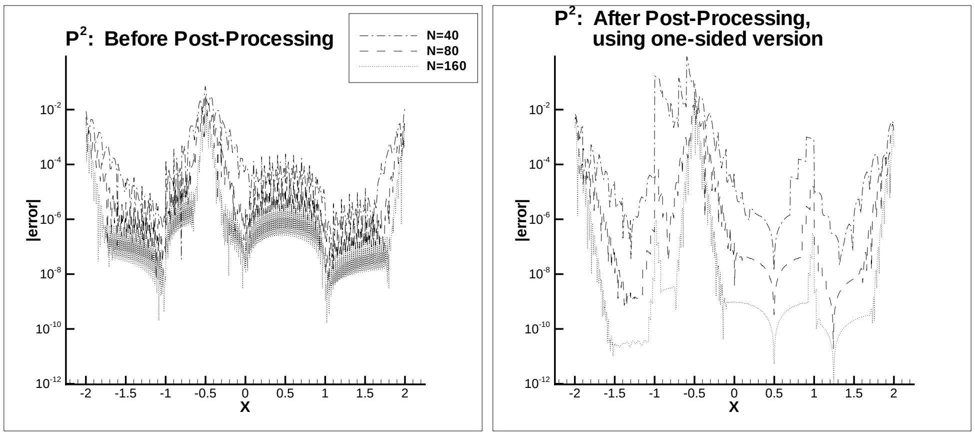

FIG. 3.6. *Pointwise errors in logarithmic scale when the $\mathbb{P}^2$ discontinuous Galerkin method is used to solve the discontinuous coefficient equation with two stationary shocks and two moving shocks. Left: before post-processing. Right: after post-processing.*