

## POSTPROCESSING FOR THE DISCONTINUOUS GALERKIN METHOD OVER NONUNIFORM MESHES\*

SEAN CURTIS<sup>†</sup>, ROBERT M. KIRBY<sup>†</sup>, JENNIFER K. RYAN<sup>‡</sup>, AND CHI-WANG SHU<sup>§</sup>

**Abstract.** A postprocessing technique based on negative order norm estimates for the discontinuous Galerkin methods was previously introduced by Cockburn, Luskin, Shu, and Süli [*Proceedings of the International Symposium on Discontinuous Galerkin Methods*, Springer, New York, pp. 291–300; *Math. Comput.*, 72 (2003), pp. 577–606]. The postprocessor allows improvement in accuracy of the discontinuous Galerkin method for time-dependent linear hyperbolic equations from order  $k+1$  to order  $2k+1$  over a uniform mesh. Assumptions on the convolution kernel along with uniformity in mesh size give a local translation invariant postprocessor that allows for simple implementation using small matrix-vector multiplications. In this paper, we present two alternatives for extending this postprocessing technique to include smoothly varying meshes. The first method uses a simple local  $L^2$ -projection of the smoothly varying mesh to a locally uniform mesh and uses this projected solution to compute the postprocessed solution. By using this local  $L^2$ -projection, recalculating the convolution kernel for every element can be avoided, and  $2k+1$  order accuracy of the postprocessed solution can be achieved. The second method uses the idea of characteristic length based upon the largest element size for the scaling of the postprocessing kernel. These two methods, local projection and characteristic length, are also applied to approximations over a mesh with elements that vary in size randomly. We discuss the computational issues in using these two techniques and demonstrate numerically that we obtain the  $2k+1$  order of accuracy for the smoothly varying meshes, and that although the  $2k+1$  order of accuracy is not fully realized for random meshes, there is significant improvement in the  $L^2$ -errors.

**Key words.** accuracy enhancement, postprocessing, discontinuous Galerkin method, hyperbolic equations

**AMS subject classification.** 65M60

**DOI.** 10.1137/070681284

**1. Introduction.** A postprocessing technique based on negative order norm estimates for the discontinuous Galerkin methods was previously introduced by Cockburn, Luskin, Shu, and Süli [4, 5]. The postprocessor allows improvement in accuracy of the discontinuous Galerkin method for time-dependent linear hyperbolic equations from order  $k+1$  to order  $2k+1$  over a uniform mesh, where  $k$  is the largest degree polynomial used in the approximation. This improvement in accuracy was extended to include superconvergence of the derivatives, two space dimensions, multi-domains with different mesh sizes, and variable and discontinuous coefficient linear hyperbolic equations [13]. Postprocessing near a computational domain boundary, discontinuity, or change in mesh size was addressed in [12]. The uniform mesh assumption along

---

\*Received by the editors January 29, 2007; accepted for publication (in revised form) May 23, 2007; published electronically December 19, 2007.

<http://www.siam.org/journals/sisc/30-1/68128.html>

<sup>†</sup>School of Computing, University of Utah, Salt Lake City, UT 84112 (scurtis@cs.utah.edu, kirby@cs.utah.edu). The first author would like to acknowledge the NSF REU support provided as part of NSF Career Award (Kirby) NSF-CCF0347791. The second author would like to acknowledge the support of ARO grant W911NF-05-1-0395.

<sup>‡</sup>Department of Mathematics, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061-0123 (jkryan@vt.edu). This author was supported by the Householder Fellowship in Scientific Computing sponsored by the DOE Applied Mathematical Sciences program, program of Oak Ridge National Laboratory (ORNL), managed by UT-Battelle, LLC, for the U.S. Department of Energy under Contract DE-AC05-00OR22725.

<sup>§</sup>Division of Applied Mathematics, Brown University, Providence, RI 02912 (shu@dam.brown.edu). This author was supported by ARO grant W911NF-04-1-0291 and NSF grant DMS-0510345.

with assumptions on the convolution kernel gives the postprocessor a local translation invariant form and thus allows for simple implementation using small matrix-vector multiplications. To postprocess one element, the matrix size is  $(2k' + 1) \times (2k' + 1)$ , where  $k' = \lceil \frac{3k+1}{2} \rceil$ . For a nonuniform mesh assumption, the size of the matrix could easily grow depending upon the size of the current element being postprocessed along with the surrounding elements used to calculate the postprocessed solution. In this paper, an extension of this postprocessing technique to include meshes with element size that vary smoothly, where the mesh is defined by a continuous function, is addressed by two techniques. In the first technique, we perform a simple *local*  $L^2$ -projection of the smoothly varying or nonuniform mesh to a uniform mesh containing  $(2k' + 1)$  elements of size  $\Delta x_i$  for the postprocessed solution. In the second method, a characteristic length based upon the size of the largest element over the smoothly varying mesh for the scaling of the postprocessing kernel is used. Both of these techniques allow for avoiding an extension of the support of the postprocessor, and  $2k+1$  order accuracy of the postprocessed solution can be achieved. These methods are also applied to approximations over a random mesh. We demonstrate numerically that although the  $2k+1$  order of accuracy is not fully realized for a random mesh, there is significant improvement in the  $L^2$ -errors using both methods.

**1.1. The discontinuous Galerkin method.** Details of the discontinuous Galerkin method for solving hyperbolic conservation laws can be found in the series of papers of Cockburn and coworkers [7, 6, 3, 2, 8], the lecture notes [1], and the review paper [9]. In this paper, linear hyperbolic equations of the form

$$(1.1) \quad \begin{aligned} u_t + (au)_x &= 0, & t &\geq 0, \\ u(x, 0) &= u_0(x), \end{aligned}$$

as well as the two-dimensional (2D) equivalent are considered. The basis for the approximation space,  $V_h$ , consists of piecewise polynomials of degree less than or equal to  $k$ , where  $k+1$  is the order of accuracy of the approximation. For the numerical studies included in this paper, the basis is taken to be the monomials,  $\xi_i = \frac{x-x_i}{\Delta x_i}$ , on the interval  $I_i = (x_i - \frac{\Delta x_i}{2}, x_i + \frac{\Delta x_i}{2})$ , where  $\Delta x_i$  is the element size. The approximation is based on a weak formulation of (1.1):

$$\int_{I_i} u_t v dx = \int_{I_i} au v_x dx - (au)_{i+1/2} v_{i+1/2} + (au)_{i-1/2} v_{i-1/2}$$

for all smooth functions  $v$ . The numerical scheme is then given by: Find  $u_h(\cdot, t) \in V_h$  such that

$$(1.2) \quad \int_{I_i} (u_h)_t v dx = \int_{I_i} au_h v_x dx - a(x_{i+1/2}^-, t) \hat{u}_{i+1/2} v_{i+1/2}^- + a(x_{i-1/2}^-, t) \hat{u}_{i-1/2} v_{i-1/2}^+$$

for all test functions  $v \in V_h$ , where the “numerical flux”  $\hat{u}_{i\pm 1/2} = u_h(x_{i\pm 1/2}^-, t)$  is chosen to be an upwind monotone flux and  $v$  is taken from inside the cell. The third order strong stability preserving total variation diminishing (SSP TVD) Runge–Kutta method is used for the time discretization [14, 10, 11], and the time step is taken so that spatial errors dominate.

**1.2. The postprocessor.** A brief review of the postprocessing technique is presented below. Details of the postprocessor implemented in this paper can be found in [4, 5, 13, 12].

The postprocessing is performed only at the final time,  $T$ , and is given by

$$(1.3) \quad u^*(x) = \frac{1}{h} \int_{-\infty}^{\infty} K^{2(k+1),k+1} \left( \frac{y-x}{h} \right) u_h(y, T) dy,$$

where  $K^{2(k+1),k+1}$  is the convolution kernel;  $h = \Delta x_i$ ,  $i = 1, \dots, N$ , represents the uniform element size; and  $u_h(y, T)$  is the discontinuous Galerkin approximation given at the final time  $T$ . Notice that the postprocessed solution,  $u^*(x)$ , is a piecewise polynomial of degree  $2k+1$  for the discontinuous Galerkin solution,  $u_h(x, T)$ , using  $\mathbb{P}^k$  elements. The form of  $K^{2(k+1),k+1}$  for the discontinuous Galerkin approximation using  $\mathbb{P}^k$  elements in one dimension is

$$(1.4) \quad K^{2(k+1),k+1}(x) = \sum_{\gamma=-k}^k c_{\gamma}^{2(k+1),k+1} \psi^{(k+1)}(x - \gamma),$$

where  $\psi^{(1)} = \chi_{(-1/2, 1/2)}$  and  $\psi^{(n)} = \psi^{(n-1)} * \chi_{(-1/2, 1/2)}$  for  $n \geq 2$  and  $c_{\gamma}^{2(k+1),k+1} \in \mathbb{R}$ .

It is important to emphasize that the uniform mesh assumption gives the kernel a particularly simple form that allows for the translation invariance of the postprocessor. The kernel also uses information only from its nearest element neighbors. This kernel could be re-evaluated for a nonuniform mesh assumption, but then the computational complexity of implementing the postprocessor would increase. For example, using the uniform mesh assumption, the exact evaluation of  $u^*(x)$  is done using small matrix-vector multiplications,

$$(1.5) \quad u^*(x) = \sum_{j=-k'}^{k'} \sum_{l=0}^k u_{i+j}^{(l)}(T) C(j, l, k, x),$$

for  $x \in I_i$ , where  $k' = \lceil (3k+1)/2 \rceil$ ,  $C(j, l, k, x)$  is a polynomial of degree  $2k+1$ , and  $u_{i+j}^{(l)}(T)$  are the coefficients in the discontinuous Galerkin approximation at the final time. The coefficients of the postprocessing matrix are given by

$$(1.6) \quad C(j, l, k, x) = \frac{1}{h} \sum_{\gamma=-k}^k c_{\gamma}^{2(k+1),k+1} \int_{I_{i+j}} \psi^{(k+1)} \left( \frac{y-x}{h} - \gamma \right) \phi_{i+j}^{(l)}(y) dy,$$

where  $\phi_{i+j}^{(l)}(y)$  represents the basis of the approximation. The application of the postprocessing step is done only at the final time, after the numerical solution has been computed, or whenever increased accuracy or regularity is needed as in visualization. If we choose to evaluate the postprocessed solution at specific points within an element, such as the Gauss–Legendre points, then we reduce the complexity to a postprocessing matrix of numbers that we can store for future use. This postprocessing matrix,  $C(j, l, k, x)$ , needs to be computed only once.

This paper is organized as follows: section 2 will present extensions of this postprocessing technique to smoothly varying and nonuniform meshes. These extensions, an  $L^2$ -projection as well as a characteristic length, will be discussed along with computational considerations. Numerical examples of the effectiveness of this local  $L^2$ -projection are presented in section 3. We summarize the material in section 4.

**2. Extending the postprocessor to smoothly varying meshes.** One of the basic assumptions in the current implementation of the postprocessor is that it is applied to an approximation over a uniform mesh. This implementation assumption can

be restrictive for many applications. However, the assumption that the postprocessing is performed over a locally uniform mesh in order to obtain  $2k+1$  convergence gives the postprocessor its translation invariance, which makes the postprocessor a local operation. For general triangulations, even though the solution is still superconvergent in the negative order norm, the postprocessor to extract the optimal  $2k+1$  convergence rate is no longer local. We wish to take advantage of the local feature captured by the uniform mesh assumption and try to capture the  $2k+1$  order accuracy for nonuniform but smooth meshes. In this paper, we propose two strategies for postprocessing over nonuniform meshes. The first idea uses a *local*  $L^2$ -projection to a *locally* uniform mesh. The second method uses a characteristic length for the B-splines of the postprocessor. Before discussing these two techniques, we discuss postprocessing for a nonuniform mesh.

If we write the postprocessed solution in a nonuniform mesh form for the basis of the approximation being the monomials, we have

$$u^*(x) = \frac{1}{\Delta x_i} \sum_j \sum_{l=0}^k u_{i+j}^{(l)}(T) \sum_{\gamma=-k}^k c_\gamma^{2(k+1),k+1} \int_{I_{i+j}} \psi^{(k+1)} \left( \frac{y-x}{\Delta x_i} - \gamma \right) \left( \frac{y-x_{i+j}}{\Delta x_{i+j}} \right)^l dy,$$

where  $j$  indicates the cells that fall within the support of the postprocessor, which may no longer be symmetric with respect to the element being postprocessed. This integral is simplified when there is a uniform mesh assumption using the change of variables,  $\eta = \frac{y-x}{h}$ , where  $h$  was the size of the uniform mesh elements. Now, we choose similarly,  $\eta = \frac{y-x}{\Delta x_i}$ , for postprocessing element  $i$ . This change of variables results in

$$(2.1) \quad u^*(x) = \sum_j \sum_{l=0}^k u_{i+j}^{(l)}(T) \sum_{\gamma=-k}^k c_\gamma^{2(k+1),k+1} \int_{y=I_{i+j}} \psi^{(k+1)}(\eta - \gamma) \left( \frac{x-x_{i+j}}{\Delta x_{i+j}} + \frac{\Delta x_i}{\Delta x_{i+j}} \eta \right)^l dy.$$

Notice that the postprocessing coefficients now depend on the mesh size of those elements within the support of the postprocessor, thus requiring the postprocessing coefficients to be recomputed for each element. We wish to avoid this recomputation and take advantage of the small precomputed matrix-vector multiplications the uniform mesh assumption provides. We consider two ideas below.

**2.1. Extension via local  $L^2$ -projections.** The first method that we implement to avoid recomputing the postprocessing matrix coefficients involves a *local*  $L^2$ -projection of the mesh to a uniform mesh and an application of the postprocessor to the projected coefficients. If the mesh is defined by a continuous function of a uniform mesh, we can still reduce oscillations in the errors for the discontinuous Galerkin method and obtain  $2k+1$  order accuracy.

Assume we are postprocessing element  $I_i = (x_i - \frac{\Delta x_i}{2}, x_i + \frac{\Delta x_i}{2})$ . We project the approximation coefficients to a locally uniform mesh of size  $h = \Delta x_i$  (see Figure 1).

Let us denote our original nonuniform mesh by  $X$ . To create the approximation that will allow us to use the uniform mesh postprocessor, we proceed in the following

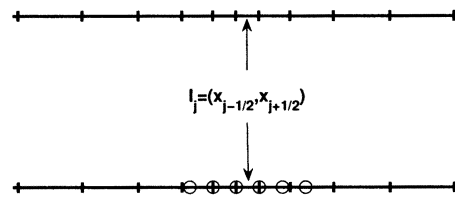


FIG. 1. Diagram illustrating the projection from the nonuniform mesh to the locally uniform mesh.  $\circ$  indicates the created locally uniform mesh for postprocessing element  $I_j$  onto which we project the approximation.

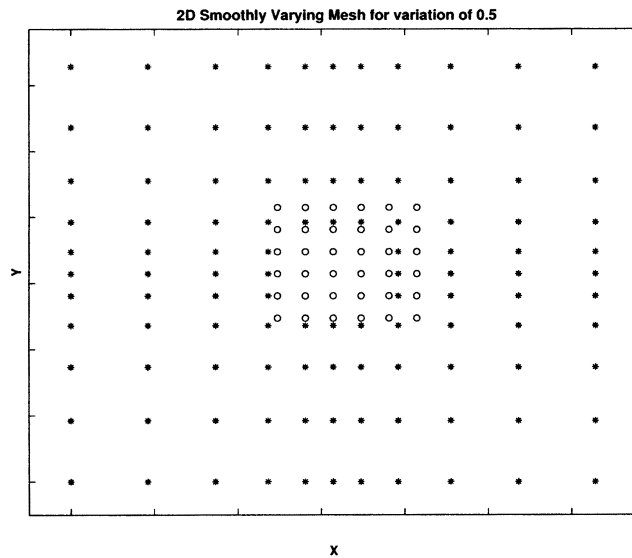


FIG. 2. Diagram illustrating the projection from the nonuniform mesh to the locally uniform mesh for 2D approximations.  $\circ$  indicates the created locally uniform mesh for postprocessing element  $I_{i,j}$  onto which we project the approximation.

manner: First, create a locally uniform mesh of  $2k'+1$  elements,  $k' = \lceil (3k+1)/2 \rceil$ , of size  $h = \Delta x_i$ , and denote this mesh by  $Y_i$ . Next, identify which elements from the original mesh fall within the support of the cell being postprocessed. Project the approximation at the final time,  $u_h(x, T)$ , to the locally uniform mesh,  $Y_i$ , for all  $x$  in the postprocessing region (see Figure 1 for 1D approximations and Figure 2 for 2D approximations). Call this projection  $u_n(x, T)$ . We then use these coefficients of the projected solution to find the postprocessed solution on element  $I_i$ . That is, the postprocessed solution is now given by

$$u^*(x) = \sum_{j=-k'}^{k'} \sum_{l=0}^k u_{n(i+j)}^{(l)}(T) C(j, l, k, x).$$

We emphasize that with this implementation, the postprocessing matrix does not change; only the vector that multiplies the postprocessing matrix changes. That is, we multiply the matrix  $C$  by the coefficients obtained from the local  $L^2$ -projection,  $u_n$ , instead of the coefficients from the approximation,  $u_h$ .

**2.2. Extension via characteristic length.** For this implementation, we forgo the extra computation required to perform the local  $L^2$ -projections and instead use a characteristic length based on the largest element size to scale the B-splines. This idea for using a characteristic length is to ensure enough information is captured from neighboring elements to perform the postprocessing for all elements. Using this characteristic length idea, we can write the postprocessed solution as

$$u^*(x) = \frac{1}{L} \sum_j \sum_{l=0}^k u_{i+j}^{(l)}(T) \sum_{\gamma=-k}^k c_\gamma^{2(k+1),k+1} \times \int_{I_{i+j}} \psi^{(k+1)} \left( \frac{y-x}{L} - \gamma \right) \left( \frac{y-x_{i+j}}{\Delta x_{i+j}} \right)^l dy,$$

where  $L$  is the characteristic length of the B-spline and the  $i$ th element is the current element being postprocessed. Using a characteristic length that is equal to the largest element size (i.e.,  $L = \max \Delta x_i, i = 1, \dots, N$ ) guarantees that the support of the postprocessor is wide enough for any element being postprocessed. Small elements contain more information on the approximation in terms of element size and hence require only small kernels. Large elements contain less information and require the convolution kernel to be larger. In this way, we obtain a sufficiently large kernel for postprocessing large elements and a larger-than-necessary kernel when postprocessing small elements. Although we cannot provably maintain this order of accuracy, we show computationally that for coarse meshes we can obtain  $2k+1$  order of accuracy. However, as we discuss below, for certain cases this may be more efficient computationally than the previous method.

**2.3. Computational considerations.** Here we briefly discuss the computational considerations of implementing both methods. First, we note a few things about the convolution kernel used in the postprocessing solution:

1. The mesh spacing enters into the scaling of the B-spline functions.
2. The polynomial element order determines the order of the B-spline used in the convolution kernel as well as the number of linear combinations of B-splines that are to be used.

In the uniform mesh case, postprocessing of the approximation over the entire domain can be constructed as a matrix-vector multiply (as well as postprocessing over one element). Since the postprocessing only increases the order per element (and does not change the spatial configuration of the mesh), one can envisage postprocessing as being the following operation:

$$u^* = C * u_h,$$

where  $u^*$  is a vector containing the new modal degrees of freedom per element (for the postprocessed solution),  $C$  is the postprocessing matrix with the precomputed convolution operator, and  $u_h$  consists of a vector of modes of the original approximation.

Consider a one-dimensional (1D) example. Suppose we have a mesh consisting of  $N$  elements and use a  $k$ th-degree polynomial approximation on each element. Then the total number of entries in  $u_h$  will be  $N * (k + 1)$ . The total number of entries in  $u^*$  will be  $N * (2k + 1)$ .  $C$  will be a linear operator of size  $(N * (2k + 1)) \times (N * (k + 1))$ .

Because of the local nature of the postprocessor, the matrix-vector multiply above is not  $\mathcal{O}(N^2)$  operations—it scales as  $\mathcal{O}(MN)$ , where  $M$  is a number not dependent

on  $N$  but dependent on the polynomial order per element. Note that in the uniform mesh case  $M$  is not dependent on the mesh size. Regardless of the element size, the number of neighboring elements used in constructing the kernel is the same.

In general,  $M \ll N$ , and hence the local postprocessing takes  $\mathcal{O}(N)$  operations to convert the entire discontinuous Galerkin approximation.

Note that the matrix  $C$  is dependent on the mesh (element configuration) and the polynomial order per element, and not on the solution. Hence, given a particular mesh and polynomial order, one can create  $C$  a priori.

The discussion above assumes that one wants to postprocess the discontinuous Galerkin approximation over the entire domain. However, in some cases it may only be necessary to postprocess at a finite number of points, such as in the computation of streamlines. In this case, the operation count effectively is the same as a vector dot-product where the scaling is  $\mathcal{O}(M)$ , where  $M$  is a number dependent on the extent of the local filter. Note that, unlike the above procedure, the coefficients needed for this “on the fly” postprocessing are not known a priori—they must be tabulated based upon the evaluation point of interest.

In the nonuniform case, we are exploring two options:

1. local  $L^2$ -projection, and
2. characteristic length.

In both of these cases—if one wants to postprocess the discontinuous Galerkin approximation over the entire domain, a formulation as mentioned above can be used. In the local  $L^2$ -projection case, the matrix  $C$  will consist of both the local projection and the postprocessing. However, there will be differences in the computational cost and storage.

The  $L^2$ -projection method will require fewer flops in general than the characteristic length formulation (based upon the largest element). This is because, in the case of the postprocessing of a small element, the  $L^2$ -projection is to a small number of elements, and hence has less (spatial) extent. The discrepancy in the flop count per element is a function of the element size discrepancy (i.e., if a mesh contains very large elements and very small elements, the characteristic length concept will always use large extent and hence use more flops). It should be noted, however, that the algorithmic scaling is still  $\mathcal{O}(N)$  for both algorithms. The additional flops for the characteristic length formulation of a global system will not be significant.

When accomplishing evaluation of individual points (i.e., not postprocessing the entire domain but rather the DG approximation at specific points), the characteristic length procedure may require fewer flops, as it does not require the projection of the field. There is a break-even point as to which method wins, depending on the ratio of maximum element size to minimum element size. If the ratio of maximum to minimum element size is very large, the  $L^2$ -projection will win (as the cost of the  $L^2$ -projection will be balanced by the number of elements that are involved). If the ratio of element sizes is not large, the characteristic length will win in terms of computational cost.

**3. Numerical examples.** In this section numerical examples demonstrate the effectiveness of the two methods, local  $L^2$ -projection and characteristic length based upon the size of the largest element, at improving the errors in the approximation. We numerically demonstrate that for a mesh defined by an analytic function we are able to increase the order of accuracy of the discontinuous Galerkin solution from  $k+1$  to  $2k+1$  and obtain improvement in the errors for the approximation over random mesh. To further simplify the application of the postprocessor, we neglect the use of one-sided or partially one-sided postprocessing by assuming periodic boundary conditions.

TABLE 1

The  $L^2$ -approximation level errors and order of accuracy for  $u(x) = \sin(x)$  for piecewise linear, quadratic, and cubic polynomials. Results are shown for before and after postprocessing for the smoothly varying mesh defined by  $x = \xi + b\sin(\xi)$ , where  $\xi$  is the uniform mesh variable.  $L^2$ -projection (L2P) versus characteristic length (CL) using largest element size. DG stands for discontinuous Galerkin.

N	DG		L2P		CL	
	$L^2$ -error	Order	$L^2$ -error	Order	$L^2$ -error	Order
$\mathbb{P}^1$						
10	1.5408E-02	—	4.2732E-02	—	5.4817E-03	—
20	3.9029E-03	1.98	2.2069E-04	3.86	3.9910E-04	3.78
40	9.7975E-04	1.99	1.5756E-05	3.80	3.6183E-05	3.46
60	4.3578E-04	2.00	3.5240E-06	3.69	1.2661E-05	2.59
80	2.4519E-04	2.00	1.2682E-06	3.55	6.7321E-06	2.20
100	1.5694E-04	2.00	5.9011E-07	3.43	4.2358E-06	2.08
$\mathbb{P}^2$						
10	1.2175E-03	—	6.1679E-04	—	6.87657E-04	—
20	1.5490E-04	2.97	1.0484E-05	5.88	2.40147E-05	4.84
40	1.9448E-05	2.99	1.6048E-07	6.03	3.97387E-07	5.92
60	5.7672E-06	3.00	1.3538E-08	6.10	3.58077E-08	5.94
80	2.4337E-06	3.00	2.3281E-09	6.12	6.85311E-09	5.75
100	1.2462E-06	3.00	5.9596E-10	6.11	2.15325E-09	5.19
$\mathbb{P}^3$						
20	3.5540E-06	—	6.3206E-07	—	1.62783E-06	—
40	2.2351E-07	3.99	2.6035E-09	7.92	6.90137E-09	7.88
60	4.4200E-08	4.00	1.0506E-10	7.92	2.73896E-10	7.96
80	1.3991E-08	4.00	1.0965E-11	7.86	2.79935E-11	7.93
100	5.7317E-09	4.00	1.9409E-12	7.76	5.07763E-12	7.65

The  $L^2$ -errors, computed using a six-point Gauss quadrature, for the discontinuous Galerkin solution as well as the postprocessed solution are given for both methods. We note that we use a mesh size variation of 50% in all examples. Similar results are obtained for mesh size variations ranging from 10–90%.

*Example 1* (1D approximation level errors). To first test the effectiveness of the nonuniform mesh postprocessor we examine the  $L^2$ -projection of the function

$$u(x) = \sin(x), \quad x \in (0, 2\pi),$$

to smoothly varying and randomly varying meshes. In all examples, the smoothly varying mesh is defined by  $x = \xi + \frac{1}{2} \sin \xi$ , so that the element sizes vary by at most 50% (or one-half) from each other. As shown in Table 1, the  $k+1$  order of accuracy is improved to better than  $2k+1$  with both the local  $L^2$ -projection used before the postprocessor and the characteristic length postprocessing. The errors for the local  $L^2$ -projection over a smoothly varying mesh are slightly better than that of the characteristic length. These methods were also applied to a mesh of randomly varying mesh size (up to 50%), as seen in Table 2. For a randomly varying mesh size both methods again improve the errors; however, this is without order improvement. The characteristic length postprocessing provides slightly better results than the local  $L^2$ -projection. For the smoothly varying mesh, the plots of the errors show that most of the oscillations from the discontinuous Galerkin solution are reduced and there is order improvement. The postprocessing only mildly improves the errors for the randomly varying mesh, but again we note the improvement in the error numbers (see Figure 3 and Table 2).

TABLE 2

The  $L^2$ -approximation level errors and order of accuracy for  $u(x) = \sin(x)$ . Results are shown for before and after postprocessing for the randomly varying mesh.  $L^2$ -projection versus characteristic length using largest element size. (Abbreviations as in Table 1.)

N	DG		L2P		CL	
	$L^2$ -error	Order	$L^2$ -error	Order	$L^2$ -error	Order
$\mathbb{P}^1$						
10	1.3497E-02	—	2.6845E-03	—	3.92033E-03	—
20	2.9586E-03	2.19	2.5385E-04	3.40	2.96326E-04	3.73
40	7.8404E-04	1.92	1.0193E-04	1.32	7.17296E-05	2.05
60	3.3135E-04	2.12	3.6936E-05	2.50	2.62094E-05	2.48
80	1.8848E-04	1.96	2.2705E-05	1.69	1.53670E-05	1.86
100	1.2365E-04	1.89	1.7024E-05	1.29	9.84808E-06	1.99
$\mathbb{P}^2$						
10	5.9665E-04	—	4.0600E-04	—	6.87657E-04	—
20	9.0912E-05	2.71	5.4703E-06	6.21	1.20052E-05	5.84
40	1.1331E-05	3.00	4.2420E-07	3.69	2.98438E-07	5.33
60	3.5481E-06	2.86	1.4570E-07	2.64	7.65188E-08	3.36
80	1.4619E-06	3.08	5.1184E-08	3.64	2.97707E-08	3.28
100	7.9487E-07	2.73	3.9110E-08	1.21	1.53304E-08	2.97
$\mathbb{P}^3$						
20	1.9964E-06	—	2.0545E-07	—	6.37260E-07	—
40	1.5233E-07	3.71	3.2787E-09	5.97	3.11111E-09	7.68
60	2.6092E-08	4.35	4.0221E-10	5.17	1.48046E-10	7.51
80	8.5628E-09	3.87	1.5402E-10	3.34	3.17030E-11	5.36
100	3.7721E-09	6.67	7.3011E-11	3.35	1.20790E-11	4.32

*Example 2* (1D linear convection equation). In this case, we consider the 1D linear convection equation

$$(3.1) \quad \begin{aligned} u_t + u_x &= 0, \\ u(x, 0) &= \sin(x) \end{aligned}$$

for  $x \in (0, 2\pi)$  and  $T = 12.5$  as in [5]. The error results are shown in Tables 3 and 4. For this equation, the smoothly varying mesh produces  $2k+1$  order accuracy in the postprocessed solution using the  $L^2$ -projection combined with the postprocessor and will at least initially show order improvement for the nonuniform postprocessing using characteristic length. For the smoothly varying mesh (Figure 4), the oscillations in the discontinuous Galerkin method are clearly reduced, and there is order improvement. Although the order of accuracy is not clear for the mesh containing elements of randomly varying sizes (Table 4), there is significant improvement in the errors.

*Example 3* (1D system). In this example, the local  $L^2$ -projection combined with the postprocessor is tested on the wave equation written as a 1D system,

$$(3.2) \quad \begin{pmatrix} u \\ v \end{pmatrix}_t + \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}_x = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

where  $x \in (0, 2\pi)$  and  $T = 12.5$  as in [4]. The initial condition is  $u(x, 0) = \sin(x)$  and  $v(x, 0) = 0$ . The  $L^2$ -errors for the smooth and random mesh case are shown in Tables 5 and 6. The error results are similar to those in Example 2. That is, for a mesh defined by an analytic function, we obtain order improvement, improvement in errors, and a reduction of oscillations. For the mesh with elements of random size, we obtain improvement in the error numbers.

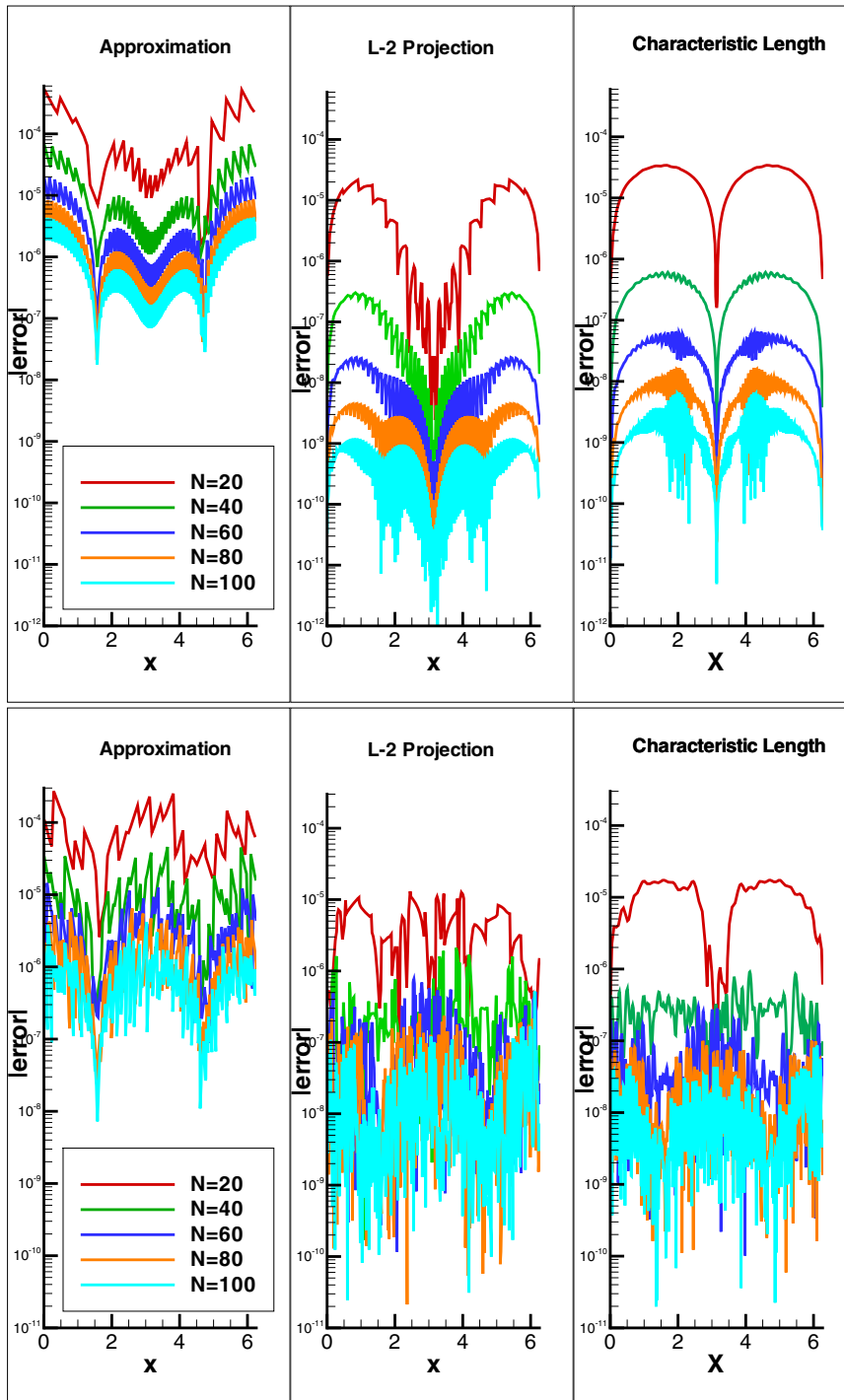


FIG. 3. Pointwise errors in log scale for the  $L^2$  approximation of  $u(x) = \sin(x)$  for smoothly varying (top) and random (bottom) meshes, where the smoothly varying mesh is defined by  $x = \xi + b\sin(\xi)$  with  $b = 0.5$  and  $x \in (0, 2\pi)$ .



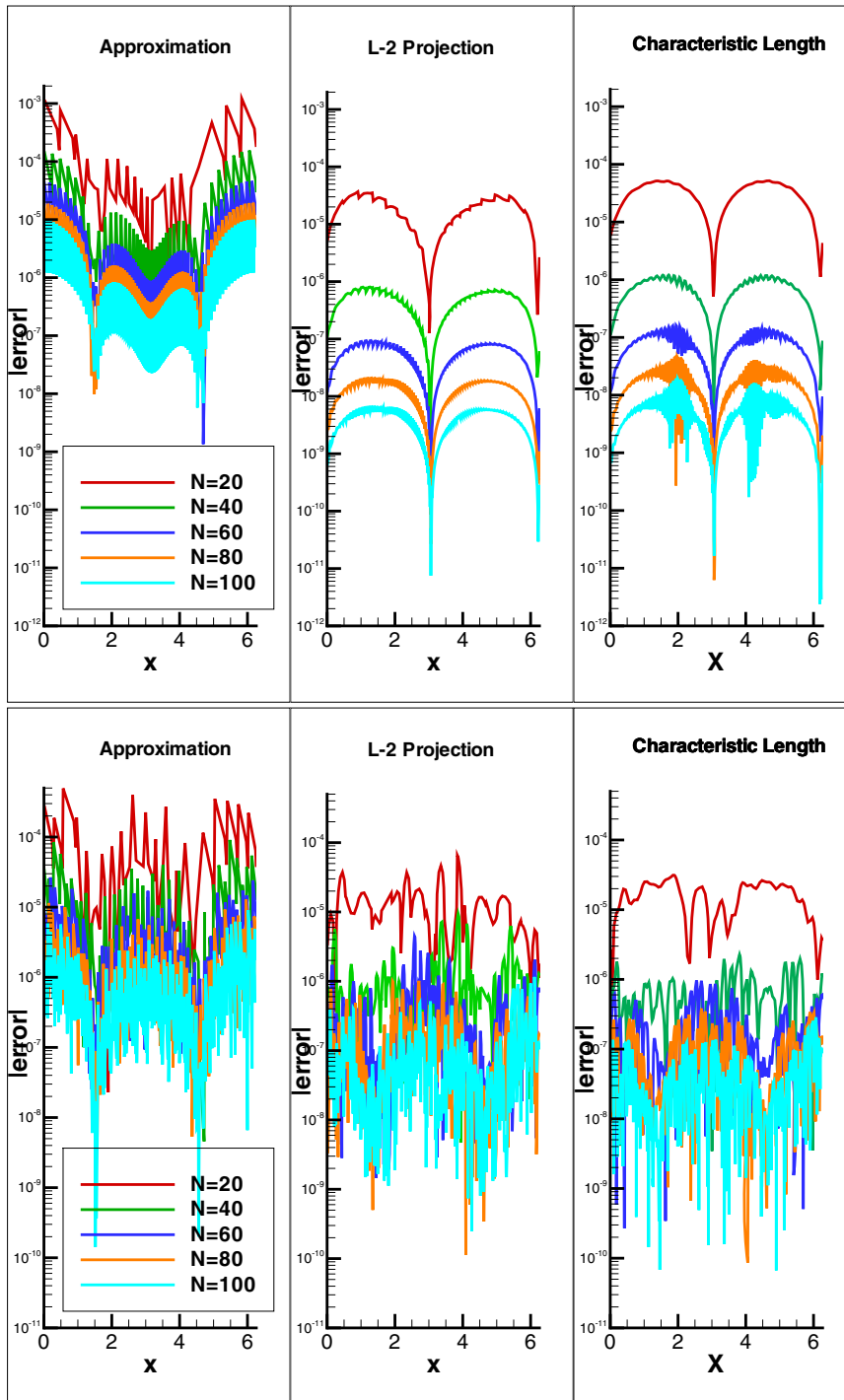


FIG. 4. Pointwise errors in log scale for the approximation to the solution of  $u_t + u_x = 0$  with  $u(x, 0) = \sin(x)$  for smoothly varying (top) and random (bottom) meshes, where the smoothly varying mesh is defined by  $x = \xi + b \sin(\xi)$  with  $b = 0.5$  and  $x \in (0, 2\pi)$ .



TABLE 7

The  $L^2$ -errors and order of accuracy for  $u_t + (a(x, t)u)_x = f(x, t)$  over  $x \in [0, 2\pi]$  with  $a(x, t) = 2 + \sin(x + t)$  and  $f(x, t)$  chosen such that  $u(x, t) = \sin(x - t)$ . Results are shown for before and after postprocessing for the smoothly varying mesh defined by  $x = \xi + b\sin(\xi)$ , where  $\xi$  is the uniform mesh variable.  $L^2$ -projection versus characteristic length using largest element size. (Abbreviations as in Table 1.)

N	DG		L2P		CL	
	$L^2$ -error	Order	$L^2$ -error	Order	$L^2$ -error	Order
$\mathbb{P}^1$						
10	2.9101E-02	—	1.8275E-02	—	2.02363E-02	—
20	6.7746E-03	2.10	2.6027E-03	2.81	2.72060E-03	2.89
40	1.6318E-03	2.05	3.4299E-04	2.92	3.49734E-04	2.96
60	7.1857E-04	2.01	1.0297E-04	2.97	1.04870E-04	2.97
80	4.0275E-04	2.01	4.3674E-05	2.98	4.47473E-05	2.96
100	2.5731E-04	2.01	2.2424E-05	2.99	2.32122E-05	2.94
$\mathbb{P}^2$						
10	2.0195E-03	—	8.9665E-04	—	4.30827E-02	—
20	2.4683E-04	3.03	2.0051E-05	5.48	3.35356E-05	—
40	3.0528E-05	3.02	4.8283E-07	5.38	7.08985E-07	5.56
60	9.0107E-06	3.01	5.7334E-08	5.26	7.84258E-08	5.43
80	3.7938E-06	3.01	1.2887E-08	5.19	1.74144E-08	5.23
100	1.9400E-06	3.01	4.0850E-09	5.15	5.80739E-09	4.92
$\mathbb{P}^3$						
10	8.6750E-05	—	1.4038E-04	—	—	—
20	5.4529E-06	3.99	6.5376E-07	7.75	1.65056E-06	—
40	3.4038E-07	4.00	2.7579E-09	7.89	7.08132E-09	7.86
60	6.7175E-08	4.00	1.1413E-10	7.86	2.86698E-10	7.91
80	2.1242E-08	4.00	1.2253E-11	7.76	3.16230E-11	7.66
100	8.6971E-09	4.00	2.2363E-12	7.62	7.18275E-12	6.64

Example 4 (1D variable coefficient equation). In the last of the 1D examples, we look at the variable coefficient equation

$$(3.3) \quad u(x, t)_t + (a(x)u(x, t))_x = f(x, t),$$

with  $a(x) = 2 + \sin(x)$  and initial condition  $u(x, 0) = \sin(x)$ . A  $2\pi$ -periodic boundary condition is used, and a suitable forcing function  $f(x, t)$  is taken so that the exact solution is  $u(x, t) = \sin(x - t)$  as in [13]. The error Tables 7, 8 and Figure 5 show that there is order improvement to  $2k+1$  for a smoothly varying mesh for the approximation. For the random mesh, we are able to reduce the errors in the approximation.

Example 5 (2D system). Next, we extend the local  $L^2$ -projection combined with the postprocessor to two dimensions using a tensor product. We consider the linear system

$$(3.4) \quad \begin{pmatrix} u \\ v \end{pmatrix}_t + \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}_x + \begin{pmatrix} 0 & -1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}_y = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

with initial conditions

$$u(x, y, 0) = \frac{1}{2\sqrt{2}} (\sin(x + y) - \cos(x + y)),$$

$$v(x, y, 0) = \frac{1}{2\sqrt{2}} ((\sqrt{2} - 1) \sin(x + y) + (1 + \sqrt{2}) \cos(x + y))$$

and  $2\pi$ -periodic boundary conditions in both directions. This is the second order wave equation written as a first order linear system. In Table 9 we list the  $L^2$ -errors

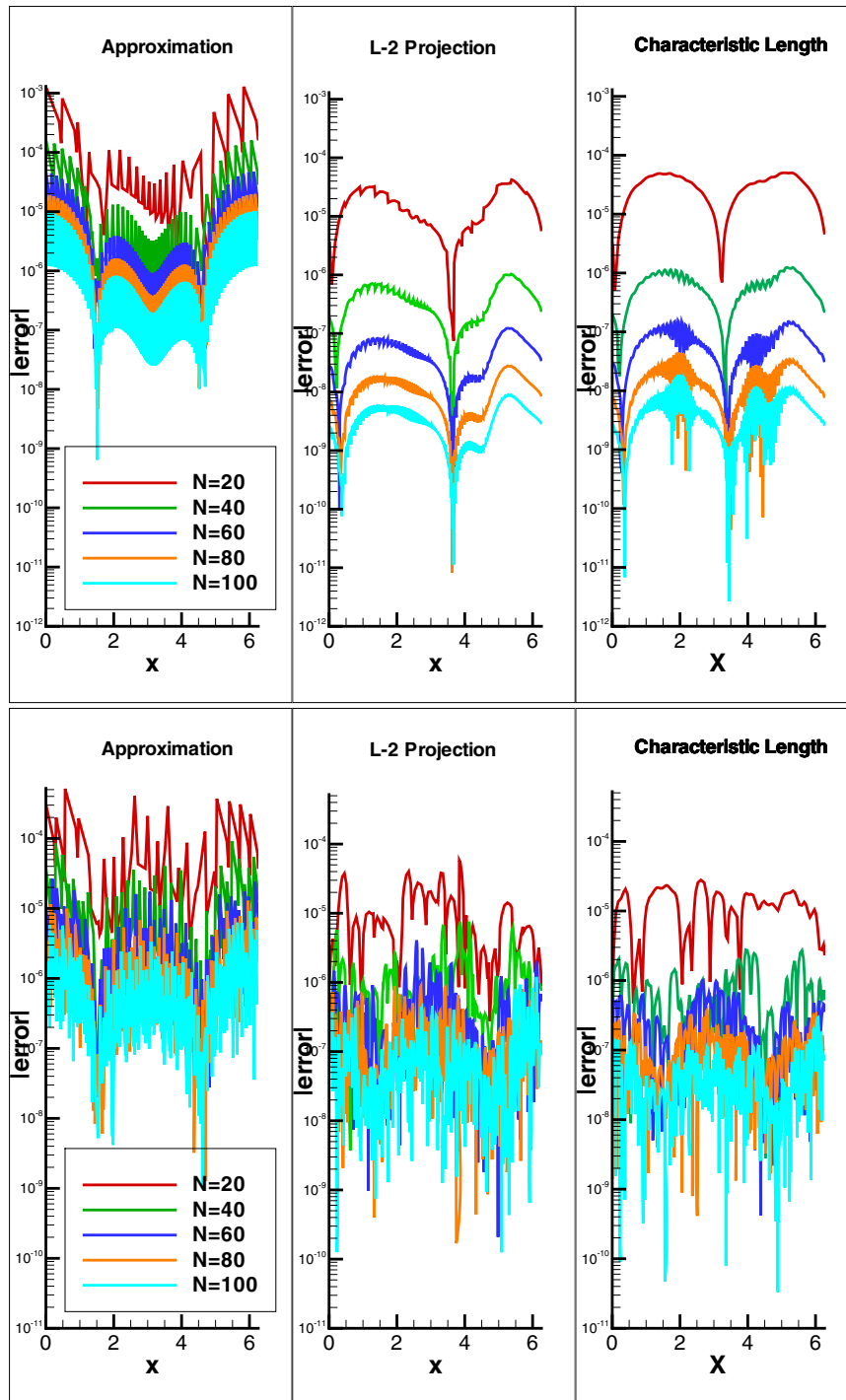


FIG. 5. Pointwise errors in log scale for  $u_t + (a(x,t)u)_x = f(x,t)$  over  $x \in (0, 2\pi)$  with  $a(x,t) = 2 + \sin(x+t)$  and  $f(x,t)$  chosen such that  $u(x,t) = \sin(x-t)$  for smoothly varying (top) and random (bottom) meshes, where the smoothly varying mesh is defined by  $x = \xi + b\sin(\xi)$  with  $b = 0.5$  and  $x \in (0, 2\pi)$ .



at  $T = 12.5$  using the  $\mathbb{P}^k$  polynomial basis. The  $(k+1)$ th order accuracy before post-processing becomes  $(2k+1)$ th order accuracy after postprocessing for the smoothly varying mesh case. For the random mesh, we see improvement in the magnitude of the errors, but no order improvement as in the 1D case.

**4. Concluding remarks.** We have demonstrated numerically two tools for enhancing the accuracy of discontinuous Galerkin approximations over meshes with elements whose size varies smoothly as well as randomly. One is a simple, *local*  $L^2$ -projection combined with the postprocessor, the other involves scaling the postprocessing kernel based upon largest element size. For the local  $L^2$ -projection combined with the postprocessor, we can provably obtain order improvement from  $k+1$  to  $2k+1$  for meshes whose variation is defined by an analytic function. Although computationally we do see this same order improvement if we choose a characteristic length for our postprocessor equal to the largest element size, we cannot guarantee that this order improvement will be maintained. This improvement from order  $k+1$  to order  $2k+1$  is the same improvement that we obtain for the uniform mesh case. Although there is not firm order of accuracy improvement for the case where the element sizes vary randomly, there is significant improvement in the magnitude of the errors. Both methods allow for the use of small matrix-vector multiplications that result from the uniform mesh assumption. Computationally, the method preferred depends upon the ratio of maximum element size to minimum element size. Using these methods allows for use of the existing postprocessing matrix and avoidance of significant increases in computational complexity. Although the effectiveness of these two techniques has not been tested when combined with the one-sided postprocessing [12], we expect similar results.

#### REFERENCES

- [1] B. COCKBURN, *Discontinuous Galerkin methods for convection-dominated problems*, in High-Order Methods for Computational Physics, Lecture Notes in Comput. Sci. Engrg. 9, Springer, New York, 1999, pp. 69–224.
- [2] 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*, Math. Comput., 54 (1990), pp. 545–581.
- [3] 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*, J. Comput. Phys., 84 (1989), pp. 90–113.
- [4] 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, Springer, New York, 1999, pp. 291–300.
- [5] B. COCKBURN, M. LUSKIN, C.-W. SHU, AND E. SÜLI, *Enhanced accuracy by post-processing for finite element methods for hyperbolic equations*, Math. Comput., 72 (2003), pp. 577–606.
- [6] B. COCKBURN AND C.-W. SHU, *TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework*, Math. Comput., 52 (1989), pp. 411–435.
- [7] B. COCKBURN AND C.-W. SHU, *The Runge-Kutta local projection  $P^1$ -discontinuous-Galerkin finite element method for scalar conservation laws*,  $M^2AN$  Math. Model. Numer. Anal., 25 (1991), pp. 337–361.
- [8] B. COCKBURN AND C.-W. SHU, *The Runge-Kutta discontinuous Galerkin method for conservation laws V: Multidimensional systems*, J. Comput. Phys., 141 (1998), pp. 199–224.
- [9] B. COCKBURN AND C.-W. SHU, *Runge-Kutta discontinuous Galerkin methods for convection-dominated problems*, J. Sci. Comput., 16 (2001), pp. 173–261.
- [10] S. GOTTLIEB AND C.-W. SHU, *Total variation diminishing Runge-Kutta schemes*, Math. Comput., 67 (1998), pp. 73–85.
- [11] S. GOTTLIEB, C.-W. SHU, AND E. TADMOR, *Strong stability-preserving high-order time discretization methods*, SIAM Rev., 43 (2001), pp. 89–112.

- [12] J. RYAN AND C.-W. SHU, *On a one-sided postprocessing technique for the discontinuous Galerkin methods*, *Methods Appl. Anal.*, 10 (2003), pp. 295–307.
- [13] J. RYAN, C.-W. SHU, AND H. 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.
- [14] C.-W. SHU AND S. OSHER, *Efficient implementation of essentially non-oscillatory shock-capturing schemes*, *J. Comput. Phys.*, 77 (1988), pp. 439–471.