Post-processing for the discontinuous Galerkin method over non-uniform meshes

Sean Curtis, Robert M. Kirby,[1]

School of Computing, University of Utah, Salt Lake City, Utah 84112

Jennifer K. Ryan[2]

Department of Mathematics, Virginia Polytechnic Institute and State University

Blacksburg, Virginia 24061-0123

and

Chi-Wang Shu[3]

Division of Applied Mathematics, Brown University, Providence, Rhode Island 02912

ABSTRACT

A post-processing technique based on negative order norm estimates for the discontinuous Galerkin methods was previously introduced by Cockburn, Luskin, Shu, and Süli [4, 5]. The post-processor 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 post-processor that allows for simple implementation using small matrix-vector multiplications. In this paper, we present two alternatives for extending this post-processing 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 post-processed 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 post-processed solution can be achieved. The second method uses

[1]{scurtis, kirby}@cs.utah.edu

[2]Corresponding author. E-mail: jkryan@math.vt.edu

[3]E-mail: shu@dam.brown.edu

the idea of characteristic length based upon the largest element size for the scaling of the post-processing 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, post-processing, discontinuous Galerkin method, hyperbolic equations

AMS(MOS) subject classification: 65M60

1 Introduction

A post-processing technique based on negative order norm estimates for the discontinuous Galerkin methods was previously introduced by Cockburn, Luskin, Shu, and Süli [4, 5]. The post-processor 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 super-convergence of the derivatives, two space dimension, multi-domains with different mesh sizes, and variable and discontinuous coefficient linear hyperbolic equations [13]. Post-processing near a computational domain boundary, discontinuity, or change in mesh size was addressed in [12]. The uniform mesh assumption along with assumptions on the convolution kernel gives the post-processor a local translation invariant form and thus allows for simple implementation using small matrix-vector multiplications. To post-process one element, the matrix size is $(2k'+1) \times (2k'+1)$ where $k' = \lceil \frac{3k+1}{2} \rceil$. For a non-uniform mesh assumption, the size of the matrix could easily grow depending upon the size of the current element being post-processed along with the surrounding elements used to calculate the post-processed solution. In this paper, an extension of this post-processing 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 $\triangle x_i$ for the post-processed 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 post-processing kernel is used. Both of these techniques allow for avoiding an extension of the support of the post-processor and $2k$+1 order accuracy of the post-processed 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 et al. [7, 6, 3, 2, 8], the lecture notes [1] and the review paper [9]. In this paper, linear hyperbolic equations of the form

$$u_t + (au)_x = 0 \tag{1.1}$$

as well as the two-dimensional 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, $\xi_i = \frac{x-x_i}{\triangle x_i}$, on the interval $I_i = (x_i - \frac{\triangle x_i}{2}, x_i + \frac{\triangle x_i}{2})$ where $\triangle x_i$ 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 $u_h(x) \in V_h$ such that

$$\int_{I_i} (u_h)_t v dx = \int_{I_i} a u_h \; v_x dx - \hat{au}^-_{i+1/2} v^-_{i+1/2} + \hat{au}^-_{i-1/2} v^+_{i-1/2} \tag{1.2}$$

for all test functions $v \in V_h$ where the 'numerical flux' is chosen to be an upwind monotone flux and v is taken from inside the cell. The third order 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 post-processor

A brief review of the post-processing technique is presented below. Details of the post-processor implemented in this paper can be found in [4, 5, 13, 12].

The post-processed solution is given by

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

where $K^{2(k+1),k+1}$ is the convolution kernel and $h = \triangle x_i, i = 1, \cdots, N$ represents the uniform element size. Notice that the post-processed solution, $u^*(x)$, is a piecewise polynomial of degree $2k+1$ for the discontinuous Galerkin solution, $u_h(x)$, 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

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

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 post-processor. The kernel also only uses information from its nearest element neighbors. This kernel could be re-evaluated for a nonuniform mesh assumption, but then the computational complexity of implementing the post-processor would increase. For example, using the uniform mesh assumption, the exact evaluation of $u^*(x)$ is done using small matrix-vector multiplications,

$$u^*(x) = \sum_{j=-k'}^{k'} \sum_{l=0}^{k} u_{i+j}^{(l)} \, C(j,l,k,x), \tag{1.5}$$

for $x \in I_i$, where $k' = \lceil (3k+1)/2 \rceil$ and $C(j,l,k,x)$ is a polynomial of degree $2k+1$. The coefficients of the post-processing matrix are given by

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

where $\phi_{i+j}^{(l)}(y)$ represents the basis of the approximation. The application of the post-processing 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 post-processed solution at specific points within an element, such as the Gauss-Legendre points, then we reduce the complexity to a post-processing matrix of numbers that we can store for future use. This post-processing matrix, $C(j,l,k,x)$, only needs to be computed once.

This paper is organized as follows: Section 2 will present extensions of this post-processing 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 post-processor to smoothly varying meshes

One of the basic assumptions in the current implementation of the post-processor 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 post-processing is performed over a locally uniform mesh in order to obtain $2k+1$ convergence gives the post-processor its translation invariance which makes the post-processor a local operation. For general triangulations, even though the solution is still superconvergent in the negative order norm, the post-processor 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 a non-uniform but smooth meshes. In this paper, we propose two strategies for post-processing 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 post-processor. Before discussing these two techniques, we discuss post-processing for a non-uniform mesh.

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

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

where j indicates the cells that fall within the support of the post-processor, which may no longer be symmetric with respect to the element being post-processed. 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, $\eta = \frac{y-x}{\triangle x_i}$, for post-processing element i. This change of variables results in

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

Notice that the post-processing coefficients now depend on the mesh size of those elements within the support of the post-processor, thus requiring the post-processing coefficients to be recomputed for each element. We wish to avoid this re-computation and take advantage of the small pre-computed 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 post-processing matrix coefficients involves a *local* L^2-projection of the mesh to a uniform mesh and an application of the post-processor 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 post-processing element $I_i = (x_i - \frac{\triangle x_i}{2}, x_i + \frac{\triangle x_i}{2})$. We project the approximation coefficients to a locally uniform mesh of size $h = \triangle x_i$ (see Figure 2.1).

Let us denote our original nonuniform mesh by X. To create the approximation that will allow us to use the uniform mesh post-processor, we proceed in the following manner: First, create a locally uniform mesh of $2k'+1$ elements, $k' = \lceil (3k+1)/2 \rceil$, of size $h = \triangle x_i$, denote this mesh by Y_i. Next, identify which elements from the original mesh fall within the support of the cell being post-processed. Project the approximation at the final time, $u_h(x, T)$, to the locally uniform mesh, Y_i, for all x in the post-processing region (see Figure 2.1). Call this projection $u_n(x, T)$. We then use these coefficients of the projected solution to find post-processed solution on element I_i. That is, the post-processed solution is now given

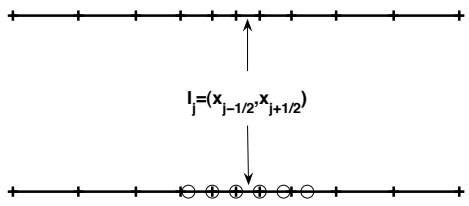

Figure 2.1: Diagram demonstrating the projection from the nonuniform mesh to the locally uniform mesh. ∘ indicates the created locally uniform mesh for post-processing element I_j onto which we project the approximation.

by

$$u^*(x) = \sum_{j=-k'}^{k'} \sum_{l=0}^{k} u^{(l)}_{n_{(i+j)}} C(j,l,k,x).$$

We emphasize that with this implementation the post-processing matrix does not change, only the vector that multiplies the post-processing 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 post-processing for all elements. Using this characteristic length idea, we can write the post-processed solution as

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

where L is the characteristic length of the B-spline, and the i^{th} element is the current element being post-processed. Using a characteristic length that is equal to the largest element size

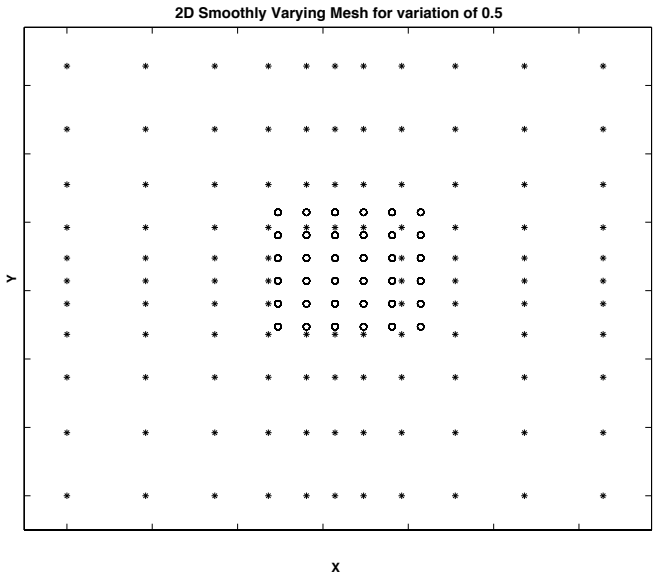

Figure 2.2: Diagram demonstrating the projection from the nonuniform mesh to the locally uniform mesh for 2D approximations. ∘ indicates the created locally uniform mesh for post-processing element $I_{i,j}$ onto which we project the approximation.

(i.e. $L = \max \triangle x_i,\ i = 1, \cdots, N$) guarantees that the support of the post-processor is wide enough for any element being post-processed. Small elements contain more information on the approximation in terms of element size and hence only require 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 post-processing large elements and a larger-than-necessary kernel when post-processing 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 post-processing solution:

- The mesh spacing enters into the scaling of the B-spline functions.
- The polynomial element order determines the order of the B-spline used in the convo-

lution kernel as well as the number of linear combinations of B-splines that are to be used.

In the uniform mesh case, post-processing of the approximation over the entire domain can be constructed as a matrix-vector multiply (as well as post-processing over one element). Since the post-processing only increases the order per element (and does not change the spatial configuration of the mesh), one can envisage post-processing 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 post-processed solution), C is the post-processing matrix with the pre-computed convolution operator and u_h consists of a vector of modes of the original approximation.

Consider a one-dimensional 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 post-processor, 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 that 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, $M << N$, and hence the local post-processing takes $\mathcal{O}(N)$ operations to convert the entire DG 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 post-process the discontinuous Galerkin approximation over the entire domain. However, in some cases it may only be necessary to

post-process 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" post-processing are not known a priori – they must be tabulated based upon the evaluation point of interest.

In the non-uniform case, we are exploring two options:

1. local L^2-projection, and

2. characteristic length.

In both of these cases – if one wants to post-process the DG 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 post-processing. However, there will be differences in the computational cost and storage.

The L^2-projection method will require less flops in general than the characteristic length formulation (based upon the largest element). This is because, in the case of the post-processing 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 post-processing the entire domain but rather the DG approximation at specific points), the characteristic length procedure may require less 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 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 demonstrating the effectiveness of the two methods, local L^2-projection and characteristic length based upon the size of the largest element, are shown to improve 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 post-processor, we neglect the use of one-sided or partially one-sided post-processing by assuming periodic boundary conditions. The L^2-errors, computed using a six-point Gauss quadrature, for the discontinuous Galerkin solution as well as the post-processed solution are given for both methods. We note that we use a mesh size variation of fifty percent in all examples. Similar results are obtained for mesh size variations ranging from ten to ninety percent.

Example 1: 1-dimensional approximation level errors

To first test the effectiveness of the nonuniform mesh post-processor we examine

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

over smoothly varying and random meshes. In all examples, the smoothly varying mesh is defined by $x = \xi + \frac{1}{2}\sin\xi$. As shown in Table 3.1 the $k+1$ order of accuracy is improved to better than $2k+1$ with both the local L^2-projection used before the post-processor and the characteristic length post-processing. 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 as seen in Table 3.2. For a randomly

Smoothly Varying: variation of $\frac{1}{2}$

	DG		L2P		CL	
N	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

Table 3.1: The L^2- approximation level errors and order of accuracy for $u(x) = \sin(x)$. Results are shown for before and after post-processing for the smoothly varying mesh defined by $x = \xi + b\sin(\xi)$ where ξ is the uniform mesh variable. L^2 projection versus characteristic length using largest element size.

varying mesh size both methods again improve the errors, however this is without order improvement. The characteristic length post-processing 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 post-processing only mildly improves the errors for the randomly varying mesh, but again we note the improvement in the error numbers (see Figure 3.1 and Table 3.2).

Example 2: 1-dimensional linear convection equation

Random: variation of $\frac{1}{2}$

			L2P		CL	
N	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

Table 3.2: The L^2- approximation level errors and order of accuracy for $u(x) = \sin(x)$. Results are shown for before and after post-processing for the randomly varying mesh. L^2 projection versus characteristic length using largest element size.

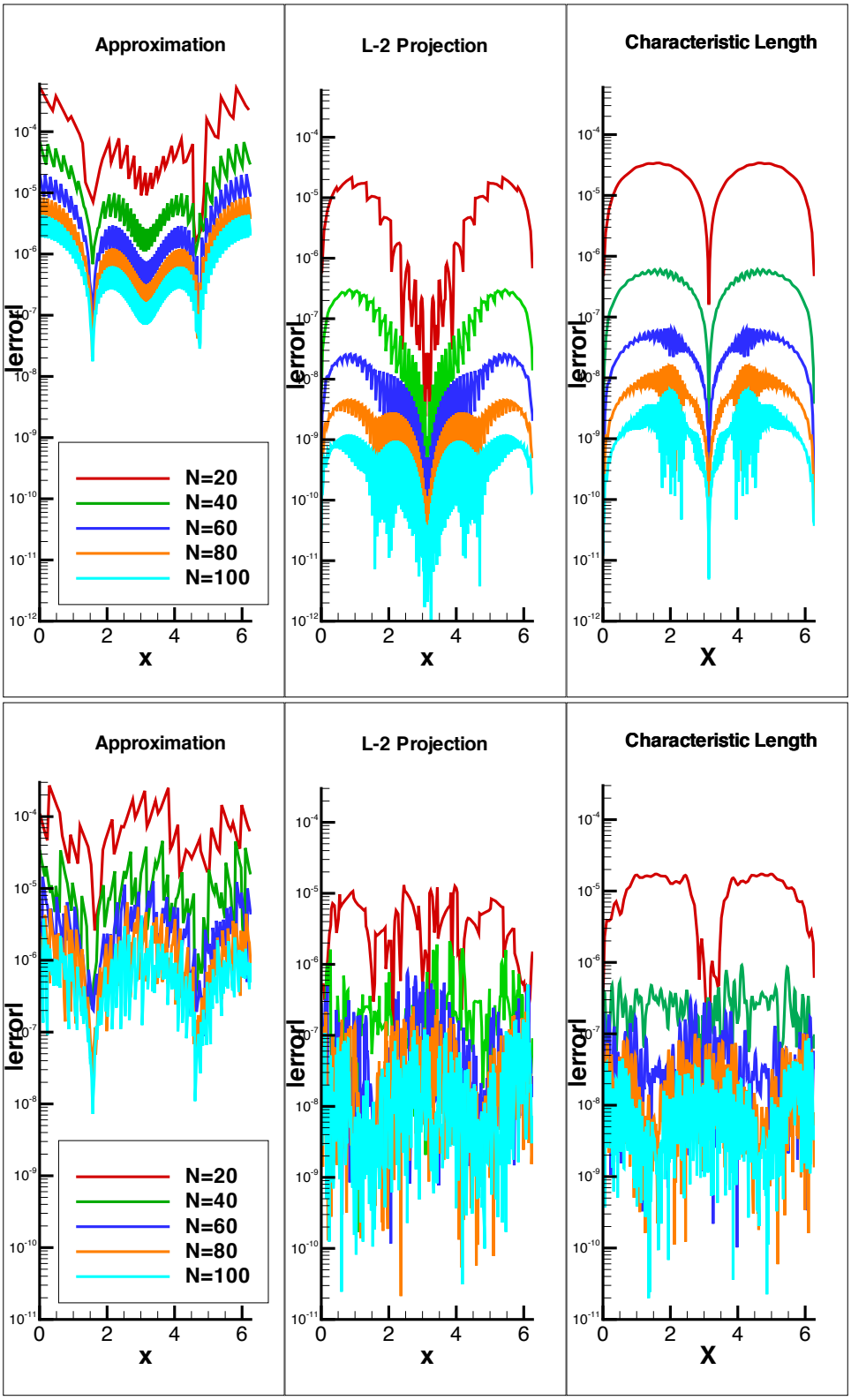

Figure 3.1: Pointwise errors in log scale for the L^2 approximation of $u(x) = \sin(x)$ for smoothly varying (top) and random (bottom) meshes where the smoothly varying mesh is defined by $x = \xi + b\sin(\xi)$ with $b = 0.5$ and $x \in (0, 2\pi)$.

In this case, we consider the one-dimensional linear convection equation

$$u_t + u_x = 0 \tag{3.1}$$

$$u(x,0) = \sin(x)$$

for $x \in (0, 2\pi)$ and $T = 12.5$ as in [5]. The error results are shown in Tables 3.3 and 3.4. For this equation, the smoothly varying mesh produces $2k+1$ order accuracy in the post-processed solution using the L^2-projection combined with the post-processor and will at least initially show order improvement for the non-uniform post-processing using characteristic length. For the smoothly varying mesh (Figure 3.2), the oscillations in the DG 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 3.4), there is significant improvement in the errors.

Example 3: 1-dimensional system

In this example, the local L^2-projection combined with the post-processor are tested on the wave equation written as a one-dimensional system

$$\begin{pmatrix} u \\ v \end{pmatrix}_t + \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix}_x = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{3.2}$$

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 3.5 and 3.6. 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.

Example 4: 1-dimensional variable coefficient equation

In the last of the one-dimensional examples, We look at the variable coefficient equation

$$u(x,t)_t + (a(x)u(x,t))_x = f(x,t). \tag{3.3}$$

Smoothly Varying: variation of $\frac{1}{2}$

	DG		L2P		CL	
N	L^2 error	order	L^2 error	order	L^2 error	order
	$\mathbb{P}^1$					
10	5.3634E-02	—	5.0774E-02	—	5.36946E-02	—
20	9.0628E-03	2.57	6.7109E-03	2.92	6.96207E-03	2.95
40	1.7995E-03	2.33	8.4615E-04	2.99	8.63550E-04	3.01
60	7.5280E-04	2.15	2.5080E-04	3.00	2.54640E-04	3.01
80	4.1368E-04	2.08	1.0578E-04	3.00	1.07232E-04	3.01
100	2.6179E-04	2.05	5.4140E-05	3.00	5.49007E-05	3.00
	$\mathbb{P}^2$					
10	1.9174E-03	—	9.2349E-04	—	8.71056E-04	—
20	2.4012E-04	3.00	2.1026E-05	5.46	3.59106E-05	4.60
40	3.0109E-05	3.00	5.1198E-07	5.36	7.75477E-07	5.53
60	8.9282E-06	3.00	6.1014E-08	5.25	8.61597E-08	5.42
80	3.7677E-06	3.00	1.3728E-08	5.19	1.90227E-08	5.25
100	1.9294E-06	3.00	4.3525E-09	5.15	6.25800E-09	4.98
	$\mathbb{P}^3$					
20	5.3906E-06	—	6.4215E-07	—	1.63951E-06	—
40	3.3871E-07	3.99	2.6784E-09	7.91	6.99678E-09	7.87
60	6.6953E-08	4.00	1.0966E-10	7.88	2.81901E-10	7.92
80	2.1189E-08	4.00	1.1659E-11	7.79	3.10608E-11	7.67
100	8.6796E-09	4.00	2.1156E-12	7.65	7.10490E-12	6.61

Table 3.3: The L^2-errors and order of accuracy for the solution to $u_t + u_x = 0$ with initial condition $u(x) = \sin(x)$. Results are shown for before and after post-processing for the smoothly varying mesh defined by $x = \xi + b\sin(\xi)$ where ξ is the uniform mesh variable. L^2 projection versus characteristic length using largest element size.

Random: variation of $\frac{1}{2}$

N	DG L^2 error	order	L2P L^2 error	order	CL L^2 error	order
			$\mathbb{P}^1$			
10	3.9288E-02	—	3.5696E-02	—	3.76980E-02	—
20	6.4910E-03	2.60	4.4325E-03	3.01	4.66228E-03	3.02
40	1.4038E-03	2.21	5.9097E-04	2.91	6.33290E-04	2.88
60	5.6738E-04	2.23	1.8322E-04	2.89	1.87807E-04	3.00
80	3.1493E-04	2.05	8.5120E-05	2.66	8.49880E-05	2.76
100	2.0449E-04	1.94	4.9641E-05	2.42	4.78369E-05	2.58
			$\mathbb{P}^2$			
10	1.0364E-03	—	5.5511E-04	—	8.71056E-04	—
20	1.4054E-04	2.88	1.6553E-05	5.07	1.80160E-05	5.60
40	1.7473E-05	3.01	1.9171E-06	3.11	7.92770E-07	4.51
60	5.4834E-06	2.86	7.0102E-07	2.48	2.96891E-07	2.42
80	2.2763E-06	3.06	2.5657E-07	3.49	1.31103E-07	2.84
100	1.2407E-06	2.72	1.7533E-07	1.71	6.61734E-08	3.06
			$\mathbb{P}^3$			
20	3.0230E-06	—	2.0912E-07	—	6.41058E-07	—
40	2.3168E-07	3.71	1.0021E-08	4.38	4.20790E-09	7.25
60	3.9623E-08	4.36	1.5744E-09	4.56	5.80172E-10	4.89
80	1.2867E-08	3.91	5.8957E-10	3.41	2.35268E-10	3.14
100	5.6505E-09	3.69	2.6617E-10	3.56	9.71225E-11	3.96

Table 3.4: The L^2-errors and order of accuracy for the solution to $u_t + u_x = 0$ with initial condition $u(x) = \sin(x)$. Results are shown for before and after post-processing for the randomly varying mesh. L^2 projection versus characteristic length using largest element size.

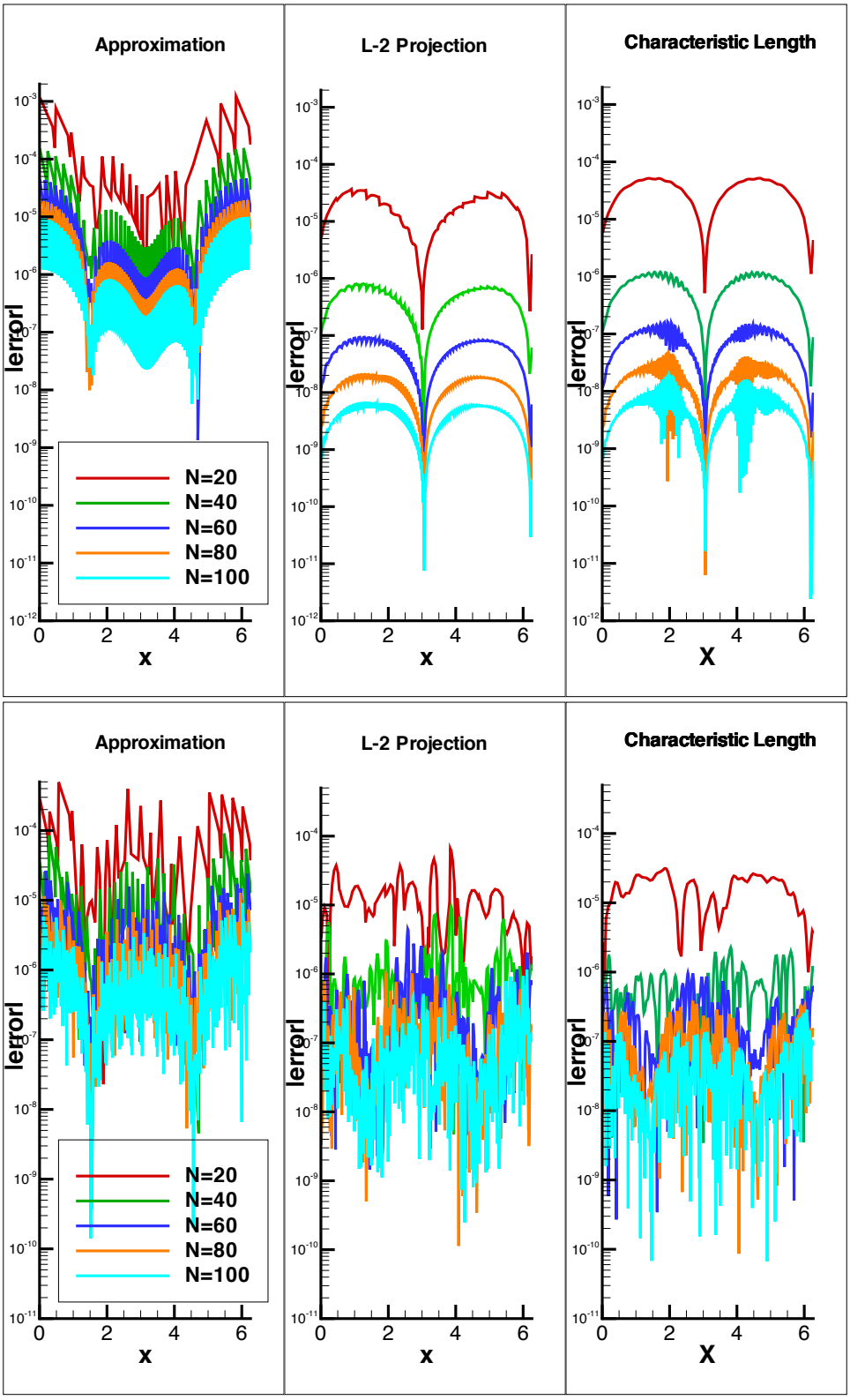

Figure 3.2: Pointwise errors in log scale for the approximation to the solution of $u_t + u_x = 0$ with $u(x,0) = \sin(x)$ for smoothly varying (top) and random (bottom) meshes where the smoothly varying mesh is defined by $x = \xi + b\sin(\xi)$ with $b = 0.5$ and $x \in (0, 2\pi)$.

Smoothly Varying: variation of $\frac{1}{2}$

	DG		L2P		CL	
N	L^2 error	order	L^2 error	order	L^2 error	order
	$\mathbb{P}^1$					
10	3.7925E-02	—	5.0774E-02	—	3.79678E-02	—
20	6.4084E-03	2.57	6.7109E-03	2.92	4.92293E-03	2.95
40	1.2725E-03	2.33	8.4615E-04	2.99	6.10622E-04	3.01
60	5.3231E-04	2.15	2.5080E-04	3.00	1.80058E-04	3.01
80	2.9251E-04	2.08	1.0578E-04	3.00	7.58247E-05	3.01
100	1.8511E-04	2.05	5.4140E-05	3.00	3.88207E-05	3.00
	$\mathbb{P}^2$					
10	1.3558E-03	—	9.2349E-04	—	—	—
20	1.6979E-04	3.00	2.1026E-05	5.46	2.53926E-05	—
40	2.1290E-05	3.00	5.1198E-07	5.36	5.48345E-07	5.53
60	6.3132E-06	3.00	6.1014E-08	5.25	6.09241E-08	5.42
80	2.6642E-06	3.00	1.3728E-08	5.19	1.34511E-08	5.25
100	1.3643E-06	3.00	4.3525E-09	5.15	4.42507E-09	4.98
	$\mathbb{P}^3$					
20	3.8117E-06	—	6.4215E-07	—	1.15931E-06	—
40	2.3950E-07	3.99	2.6784E-09	7.91	4.94747E-09	7.87
60	4.7343E-08	4.00	1.0966E-10	7.88	1.99334E-10	7.92
80	1.4983E-08	4.00	1.1659E-11	7.79	2.19634E-11	7.67
100	6.1374E-09	4.00	2.1156E-12	7.65	5.02394E-12	6.61

Table 3.5: The L^2-errors and order of accuracy for the wave equation written as a one-dimensional system with $u(x,0) = \sin(x)$ calculated at final time, T=12.5. Results are shown for before and after post-processing for the smoothly varying mesh defined by $x = \xi + b\sin(\xi)$ where ξ is the uniform mesh variable. L^2 projection versus characteristic length using largest element size.

Random: variation of $\frac{1}{2}$

	DG		L2P		CL	
N	L^2 error	order	L^2 error	order	L^2 error	order
	$\mathbb{P}^1$					
10	2.8103E-02	—	3.5654E-02	—	2.66599E-02	—
20	4.5860E-03	2.62	4.4442E-03	3.00	3.29707E-03	3.02
40	9.9064E-04	2.21	5.8598E-04	2.92	4.47100E-04	2.88
60	4.0057E-04	2.23	1.7898E-04	2.93	1.32845E-04	2.99
80	2.2328E-04	2.03	8.3025E-05	2.67	6.04966E-05	2.73
100	1.4523E-04	1.93	4.9729E-05	2.30	3.39610E-05	2.59
	$\mathbb{P}^2$					
10	6.7673E-04	—	5.6663E-04	—	6.15856E-04	—
20	9.9537E-05	2.77	1.6937E-05	5.06	1.27342E-05	5.60
40	1.2421E-05	3.00	1.8908E-06	3.16	1.02438E-06	3.64
60	3.8858E-06	2.87	6.6401E-07	2.58	3.45023E-07	2.68
80	1.6019E-06	3.08	2.4758E-07	3.43	1.36709E-07	3.22
100	8.7105E-07	2.73	1.8076E-07	1.42	6.80892E-08	3.12
	$\mathbb{P}^3$					
20	2.1355E-06	—	2.0901E-07	—	6.37371E-07	—
40	1.6278E-07	3.72	9.3177E-09	4.49	7.77741E-09	6.36
60	2.7931E-08	4.35	1.5057E-09	4.50	8.59372E-10	5.43
80	9.1654E-09	3.87	5.6320E-10	3.42	2.25153E-10	4.66
100	4.0393E-09	3.67	2.6826E-10	3.32	8.26038E-11	4.49

Table 3.6: The L^2-errors and order of accuracy for the wave equation written as a one-dimensional system with $u(x,0) = \sin(x)$ calculated at final time, T=12.5 Results are shown for before and after post-processing for the randomly varying mesh. L^2 projection versus characteristic length using largest element size.

with $a(x) = 2 + \sin(x)$ and initial condition $u(x,0) = \sin(x)$. A 2π 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 3.7, 3.8 and Figure 3.3 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: 2-dimensional system

Next, we extend the local L^2-projection combined with the post-processor to two-dimensions using a tensor product. We consider the linear system

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

with initial conditions

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

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

and 2π periodic boundary conditions in both directions. This is the second order wave equation written as a first order linear system. In Table 3.9 we list the L^2-errors 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 post-processing 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 one-dimensional case.

4 Concluding Remarks

We have demonstrated numerically two tools for enhancing the accuracy of the 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 post-processor, the other, scaling the post-processing kernel based upon largest element size. For the local L^2-projection combined with the post-processor, we can provably obtain order improvement

Smoothly Varying: variation of $\frac{1}{2}$

	DG		L2P		CL	
N	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

Table 3.7: 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 post-processing for the smoothly varying mesh defined by $x = \xi + b\sin(\xi)$ where ξ is the uniform mesh variable. L^2 projection versus characteristic length using largest element size.

Random: variation of $\frac{1}{2}$

	DG		L2P		CL	
N	L^2 error	order	L^2 error	order	L^2 error	order
	$\mathbb{P}^1$					
10	2.2959E-02	—	1.0717E-02	—	1.05522E-02	—
20	4.9890E-03	2.20	1.5387E-03	2.80	1.37002E-03	2.95
40	1.2894E-03	1.95	3.2431E-04	2.25	2.69588E-04	2.35
60	5.4391E-04	2.13	1.1644E-04	2.53	9.01755E-05	2.70
80	3.0760E-04	1.98	6.4140E-05	2.07	4.97856E-05	2.06
100	2.0130E-04	1.90	4.2805E-05	1.81	—	—
	$\mathbb{P}^2$					
10	1.0398E-03	—	4.6250E-04	—	7.14988E-04	—
20	1.4013E-04	2.89	1.5527E-05	4.90	1.43903E-05	5.63
40	1.7506E-05	3.00	1.9061E-06	3.03	1.02438E-06	3.81
60	5.4792E-06	2.86	6.8815E-07	2.51	3.45023E-07	2.68
80	2.2757E-06	3.05	2.5205E-07	3.49	1.36709E-07	3.22
100	1.2419E-06	2.71	1.7345E-07	1.67	6.80892E-08	3.12
	$\mathbb{P}^3$					
10	6.3658E-05	—	7.6557E-05	—	—	—
20	3.0291E-06	4.39	2.2402E-07	8.42	6.37371E-07	—
40	2.3150E-07	3.71	1.2726E-08	4.14	7.77741E-09	6.36
60	3.9588E-08	4.36	1.4673E-09	5.33	8.59372E-10	5.43
80	1.2854E-08	3.91	5.1104E-10	3.67	2.25153E-10	4.66
100	5.6454E-09	3.69	2.3984E-10	3.39	8.26038E-11	4.49

Table 3.8: 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 post-processing for the randomly varying mesh. L^2 projection versus characteristic length using largest element size.

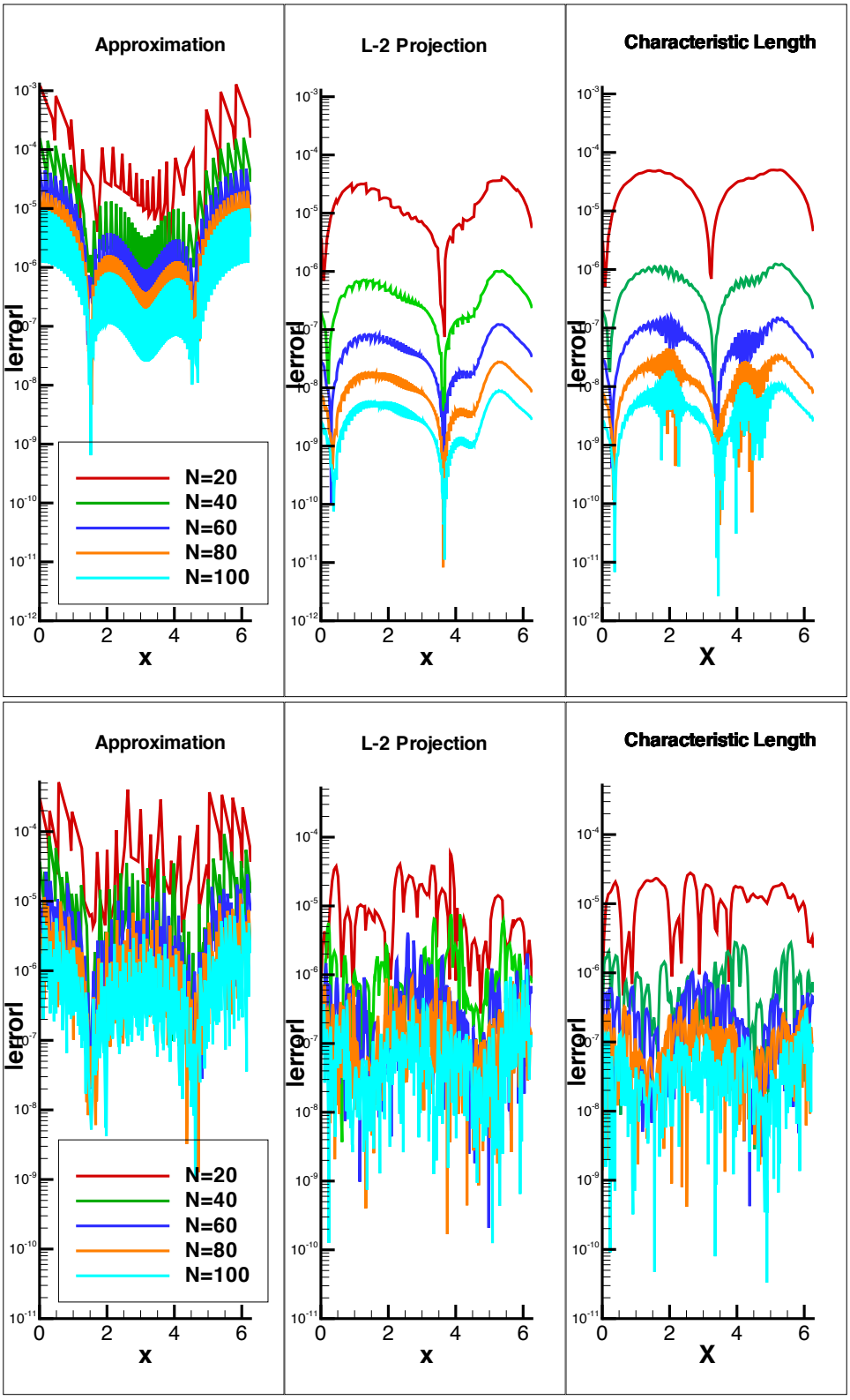

Figure 3.3: Pointwise errors in log scale for $u_t + (a(x,t)u)_x = f(x,t)$ over $x \in (0, 2\pi)$ with $a(x,t) = 2 + \sin(x+t)$ and $f(x,t)$ chosen such that $u(x,t) = \sin(x-t)$ for smoothly varying (top) and random (bottom) meshes where the smoothly varying mesh is defined by $x = \xi + b\sin(\xi)$ with $b = 0.5$ and $x \in (0, 2\pi)$.

Smoothly Varying: variation of $\frac{1}{2}$

	DG		L2P		CL	
N	L^2 error	order	L^2 error	order	L^2 error	order
	$\mathbb{P}^1$					
10	2.5646E-01	—	9.3597E-02	—	2.6147E-01	—
20	4.6170E-02	2.47	2.9566E-02	1.66	4.5489E-02	2.52
40	6.7617E-03	2.77	5.0229E-03	2.56	6.1035E-03	2.90
60	2.2637E-03	2.70	1.6096E-03	2.81	1.8268E-03	2.98
80	1.0827E-03	2.56	7.0378E-04	2.88	7.77288E-04	2.97
100	6.2764E-04	2.44	3.6777E-04	2.91	3.9614E-04	3.02
	$\mathbb{P}^2$					
20	8.0988E-04	—	9.4633E-05	—	2.5239E-04	—
40	9.7808E-05	3.05	4.7729E-06	4.31	7.3263E-06	5.11
60	2.8909e-05	3.01	7.1507E-07	4.68	9.3603e-07	5.07
80	1.2189e-05	3.00	1.7936E-07	4.81	2.189e-07	5.05
100	6.2397e-06	3.00	6.0610E-08	4.86	7.13e-08	5.03
	$\mathbb{P}^3$					
20	5.0590E-05	—	2.5667E-07	—	3.9587e-06	—
40	3.3662E-06	3.91	5.0348E-09	5.67	1.9928e-08	7.63
60	6.7469e-07	3.96	3.7718E-10	6.39	9.2025e-10	7.58
80	2.1463e-07	3.98	5.5094E-11	6.69	1.0677e-10	7.49
100	8.8139e-08	3.99	1.2036E-11	6.82	2.0881e-11	7.31

Table 3.9: The L^2-errors and order of accuracy for the wave equation written as a two-dimensional system calculated at final time, T=12.5. Results are shown for before and after post-processing for the smoothly varying mesh defined by $x = \xi + b\sin(\xi)$ where ξ is the uniform mesh variable. L^2 projection versus characteristic length using largest element size.

from $k+1$ to $2k+1$ for meshes whose variation are defined by an analytic function. Although computationally we do see this same order improvement if we choose a characteristic length for our post-processor 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 use of small matrix-vector multiplications that result from the uniform mesh assumption. Computationally, the preferred method depends upon the ratio of maximum element size to minimum element size. Using these methods allows for using the existing post-processing matrix and avoidance of significant increase in computational complexity.

Acknowledgments

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 number W911NF-05-1-0395. The third author would like to acknowledge the support of 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 No. DE-AC05-00OR22725. The fourth author would like to acknowledge the support of ARO grant W911NF-04-1-0291 and NSF grant DMS-0510345.

References

[1] B. Cockburn, *Discontinuous Galerkin methods for convection-dominated problems*, vol. 9, Springer, 1999.

[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*, Mathematics of Computation, 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*, Journal of Computational Physics, 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, 1999, pp. 291–300.

[5] B. Cockburn, M. Luskin, C.-W. Shu, and E. Suli, *Enhanced accuracy by post-processing for finite element methods for hyperbolic equations*, Mathematics of Computation, 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*, Mathematics of Computation, 52 (1989), pp. 411–435.

[7] ——, *The Runge-Kutta local projection P^1-discontinuous-Galerkin finite element method for scalar conservation laws*, Mathematical Modeling and Numerical Analysis (M^2AN), 25 (1991), pp. 337–361.

[8] ——, *The Runge-Kutta discontinuous Galerkin method for conservation laws v: multidimensional systems*, Journal of Computational Physics, 141 (1998), pp. 199–224.

[9] ——, *Runge-Kutta discontinuous Galerkin methods for convection-dominated problems*, Journal of Scientific Computing, 16 (2001), pp. 173–261.

[10] S. Gottlieb and C.-W. Shu, *Total variation diminishing Runge-Kutta schemes*, Mathematics of Computation, 67 (1998), pp. 73–85.

[11] S. Gottlieb, C.-W. Shu, and E. Tadmor, *Strong stability preserving high-order time discretization methods*, SIAM Review, 43 (2001), pp. 89–112.

[12] J. Ryan and C.-W. Shu, *On a one-sided post-processing technique for the discontinuous Galerkin methods*, Methods and Applications of Analysis, 10 (2003), pp. 295–307.

[13] J. Ryan, C.-W. Shu, and H. Atkins, *Extension of a post-processing technique for the discontinuous Galerkin method for hyperbolic equations with application to an aeroacoustic problem*, SIAM Journal on Scientific Computing, 26 (2005), pp. 821–843.

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