A Parallel $h$-$p$-Adaptive Discontinuous Galerkin Method for Hyperbolic Conservation Laws

Kim S. Bey
Mail Stop 396
NASA Langley Research Center
Hampton, VA 23681

and

J. Tinsley Oden and Abani Patra
The Texas Institute for Computational and Applied Mathematics
University of Texas at Austin
Austin, TX 78712

November 1994

Submitted to Applied Numerical Mathematics, a journal of IMACS
A Parallel $hp$-Adaptive Discontinuous Galerkin Method for Hyperbolic Conservation Laws

Kim S. Bey
NASA Langley Research Center
Hampton, VA 23681

J. Tinsley Oden and Abani Patra
The Texas Institute for Computational and Applied Mathematics
University of Texas at Austin
Austin, TX 78712

November 1994

Abstract

This paper describes a parallel adaptive strategy based on discontinuous $hp$-finite element approximations of linear, scalar, hyperbolic conservation laws. The paper focuses on the development of an effective parallel adaptive strategy for such problems. Numerical experiments suggest that these techniques are highly parallelizable and deliver super-linear rates of convergence, thereby yielding efficiency many times superior to conventional schemes for hyperbolic problems.

1 Introduction

This paper summarizes recent results on the development of parallel $hp$-adaptive discontinuous Galerkin methods for hyperbolic conservation laws. The present study focuses on a model class of scalar linear hyperbolic conservation laws for which it is possible to develop mathematically rigorous $a$ priori and $a$ posteriori error esti-
mates. These estimates form the basis of an $hp$-adaptive strategy designed to deliver a solution with the specified error in three steps.

The notion of discontinuous Galerkin methods for hyperbolic problems originated in the classical work of Lesaint and Raviart [1] over nearly a decade ago. Johnson and Pitkaranta [2] generalized the theory of discontinuous Galerkin methods by introducing mesh-dependent norms and were able to derive a priori error estimates in such norms for linear hyperbolic problems. Discontinuous Galerkin methods were extended to nonlinear hyperbolic conservation laws by Cockburn, Hou, and Shu [3] who developed a local projection strategy to provide nonlinear stability.

The theory of discontinuous Galerkin methods was extended to $hp$-finite element approximations by Bey and Oden [4] who derived a priori and a posteriori error estimates for a class of linear hyperbolic conservation laws. In [4], very high accuracies and convergence rates were observed in applying such schemes to representative test problems. The a posteriori error estimates developed in [4] provide bounds on the global error and were shown to provide accurate estimates of the global and local error for representative test problems.

The local nature of discontinuous Galerkin methods make them ideally suited for parallel architectures, and a parallel $hp$-adaptive strategy was developed by Bey, Patra, and Oden [5]. The $hp$-adaptive strategy was shown to deliver super-linear rates of convergence for smooth solutions [4] and for discontinuous solutions with the discontinuity aligned with the mesh [5]. Combined with an efficient parallel implementation with a special load-balancing strategy designed for $hp$-meshes, the scheme was shown to deliver results with efficiencies several orders of magnitude faster than conventional approaches.
The purpose of this paper is to further investigate the performance of the method when applied to discontinuous solutions. The discontinuous Galerkin method is described in the following section. The a posteriori error estimate used to assess solution accuracy is presented in Section 3. The hp-adaptive strategy and the parallel implementation are summarized in Sections 4 and 5 respectively. Finally, numerical results are presented in Section 6.

2 The Discontinuous Galerkin Method

Consider the following hyperbolic conservation law

\[
\begin{align*}
\beta \cdot \nabla u + au &= f \quad \text{in } \Omega \subset \mathbb{R}^2 \\
\beta \cdot n u &= \beta \cdot n g \quad \text{on } \Gamma_-
\end{align*}
\]

where \( \beta = (\beta_1, \beta_2)^T \) denotes a constant unit velocity vector, \( n \) denotes the unit normal vector pointing outward to the domain boundary \( \partial \Omega \), \( \Gamma_- = \{ x \in \partial \Omega \mid \beta \cdot n(x) < 0 \} \) denotes the inflow boundary, \( a = a(x) \) is a bounded measurable function on \( \Omega \) such that \( 0 < a_0 \leq a(x) \), \( f \in L^2(\Omega) \), and \( g \in L^2(\Gamma_-) \). While this is the simplest of hyperbolic conservation laws, solutions to (1) may contain discontinuities along characteristic lines \( x(s) \) defined by \( \frac{dx}{ds} = \beta \). Solutions to (1) belong to the space of functions \( V(\Omega) = \{ v \in L^2(\Omega) \mid v_g \in L^2(\Omega) \} \) where \( v_g = \beta \cdot \nabla v \).

The starting point for the discontinuous Galerkin methods is to develop an appropriate weak formulation of (1) defined on a partition of \( \Omega \) into elements, denoted by \( \mathcal{P}_h \). Here the elements \( K \in \mathcal{P}_h \) are general quadrilaterals of diameter \( h_K \). The
space of admissible solutions is extended to the partition using the broken space
\[ V(P_h) = \bigcup_{K \in P_h} V(K). \]

The standard conventions in finite element meshing are assumed to be in force: \( P_h \) is a family of partitions \( \mathcal{F}_h \) and each element \( K \) of \( P_h \) is the image of an invertible map \( F_K \) of a master element \( \hat{K} = [-1, 1]^2 \). The partitions \( P_h \in \mathcal{F}_h \) are regular and, in the present study, it is sufficient to take \( F_K \) as affine maps. For each partition \( P_h \), approximate solutions are sought in the subspace
\[ V_p(P_h) = \{ v \in L^2(\Omega) \mid v|_K \circ F_K^{-1} \in Q^{p_K}(\hat{K}) \} \]
where \( Q^{p_K}(\hat{K}) \) denotes the space of functions formed by tensor products of Legendre polynomials of degree \( p_K \) on the master element \( \hat{K} \). Note that the polynomial degree, \( p_K \), may vary over different elements in the mesh and that functions \( v^p_h \in V_p(P_h) \) are discontinuous across element interfaces. The approximation properties of such spaces are typified by local interpolation estimates of the following type (see [6]): if \( u \in H^s(K) \), there exists a constant \( C \), independent of \( h_K = \text{diam}(K) \) and \( p_K \) (the minimal order of the polynomial shape functions for \( K \)), and a polynomial \( w \) of degree \( p_K \), such that
\[
\|u - w\|_{r,K} \leq C \frac{h_K^{\min(p_K+1,s)}}{p_K^{s-r}} \|u\|_{s,K} \quad ; \quad r = 0,1
\]
where \( \| \cdot \|_{r,K} \) denotes the usual Sobolev norm.

The following notation is used for functions \( v \in V(P_h) \):
\[
\begin{align*}
\nu^\pm &= \lim_{\epsilon \to 0} \nu(\mathbf{x} \pm \epsilon \mathbf{b}) \\
\langle v, w \rangle_K &= \int_K v w \, d\mathbf{x} \quad ; \quad \|v\|_K = \sqrt{\langle v, v \rangle_K} \\
\langle v, w \rangle_{\partial K} &= \int_{\partial K} v w \mid \mathbf{b} \cdot \mathbf{n}_K \mid \, ds \quad ; \quad \langle \langle v \rangle \rangle_{\partial K} = \sqrt{\langle v, v \rangle_{\partial K}}
\end{align*}
\]
The discontinuous Galerkin method applied to (1) is written in the following abstract form:

Find \( \hat{u} \in V_p(\mathcal{P}_h) \) such that

\[
\sum_{K \in \mathcal{P}_h} B_K(\hat{u}, \hat{v}) = \sum_{K \in \mathcal{P}_h} L_K(\hat{v}), \quad \text{for every } \hat{v} \in V_p(\mathcal{P}_h)
\]

(5)

where (see [4])

\[
B_K(\hat{u}, \hat{v}) \overset{\text{def}}{=} (\hat{u}, \hat{v} + \frac{h_K}{p_K^2} \hat{v})_K + (1 + \frac{h_K}{p_K^2})(\hat{u}, \hat{v})_{\partial K \setminus \Gamma_-} + (\hat{u}, \hat{v}^+)_{\partial K \cap \Gamma_-}
\]

(6)

\[
L_K(\hat{v}) \overset{\text{def}}{=} (f, \hat{v} + \frac{h_K}{p_K} \hat{v})_K + (1 + \frac{h_K}{p_K})(g, \hat{v})_{\partial K \cap \Gamma_-}
\]

(7)

The error in the discontinuous Galerkin solution satisfies the following a priori estimate [4]:

Theorem 1 Let \( u \in H^s(\Omega) \) be a solution to (1), let \( \hat{u} \) be a solution to (5), and let (3) hold. Then there exists a positive constant \( C \), independent of \( h_K, p_K, \) and \( u \), such that the approximation error, \( e = u - \hat{u} \), satisfies the following estimate

\[
|||e|||_{h,p,\theta} \leq C \left\{ \sum_{K \in \mathcal{P}_h} \left[ \frac{h_K^{2\mu_K}}{2\nu_K} \max \left( 1, \frac{h_K}{p_K^2} \right) |||u|||_{2,K}^2 \right] \right\}^{\frac{1}{2}}
\]

(8)

where \( \mu_K = \min(p_K + 1, s) - \frac{1}{2}, \nu_K = s - 1, \) and

\[
|||e|||_{h,p,\theta} = \left\{ \sum_{K \in \mathcal{P}_h} \left[ \frac{h_K^{2\mu_K}}{p_K^2} |||e^h|||_K^2 + |||e|||_K^2 + (\langle e^+ - e^- \rangle_{\partial K \setminus \Gamma_-})^2 + (\langle e \rangle_{\partial K \cap \partial \Omega})^2 \right] \right\}^{\frac{1}{2}}
\]

(9)
The *a priori* estimate (8) establishes convergence of the method and is useful for predicting how the error in numerical solutions behaves with \( h \)-refinement or \( p \)-enrichment. Unfortunately, its usefulness in assessing the accuracy of a given numerical solution is limited since the estimate involves unknown constants and the exact solution.

### 3 A Posteriori Error Estimation

*A posteriori* error estimates used here are based on extensions of an element residual method of Ainsworth and Oden [7]. Element error indicators are computed by solving a suitably-constructed local problem with the element residual as data. These local indicators are used in the adaptive strategy to assess the accuracy of the solution in an element. Moreover, they contribute to a global error estimate which is accurate enough to provide a reliable assessment of the quality of the approximate solution. Detailed derivation of the *a posteriori* estimate and detailed numerical results demonstrating its effectiveness for hyperbolic problems can be found in Bey and Oden [4].

The local problem is constructed to result in an upper bound on the error. Let \( \psi_K \) be the solution to the following local problem,

\[
A_K^u(\psi_K, v_K) = B_K(e_K, v_K) = L_K(v_K) - B_K(\hat{u}_K, v_K) \quad \forall v \in V(\mathcal{P}_h)
\]  

where

\[
A_K^v(\psi_K, v_K) \overset{\text{def}}{=} \frac{h_K}{p_K^2}(\beta \cdot \nabla \psi_K, \beta \cdot \nabla v_K)_K + \bar{a}(\psi_K, v_K)_K
\]  

(11)
and $a > 0$ is a constant. The solution to the local problem, measured in the norm,

$$\|\psi_K\|_{A^0\psi(K)} = \sqrt{A^0\psi(\psi_K, \psi_K)}$$

serves as an element error indicator in the adaptive strategy. The global error estimate is given by

$$\|\psi\|_{A^0\psi} = \sqrt{\sum_{K \in \mathcal{P}_h} \|\psi_K\|_{A^0\psi(K)}^2}$$

The solution to the local problem (10) provides an upper bound on the global error in the following sense [4]:

**Lemma 1** Let $\psi \in V(\mathcal{P}_h)$ be the solution to the following problem:

$$\sum_{K \in \mathcal{P}_h} A^0\psi(\psi_K, v) = \sum_{K \in \mathcal{P}_h} B(e, v) \quad \forall v \in V(\mathcal{P}_h)$$

Then there exists a positive constant $k$ such that

$$\|\psi\|_{A^0\psi} \geq k \|e\|_{hp, \beta}$$

An approximate solution to the local problem (10) in the corresponding norm serves as a local error indicator for the element. Since the discontinuous Galerkin solution satisfies the orthogonality condition,

$$B_K(e, v) = 0 \quad \forall v \in Q^h(K)$$
the error indicator must be approximated with a polynomial of degree $p_K + \sigma_K$ where $\sigma_K \geq 1$ in order for the discrete local problem to have a non-trivial solution. If a complete polynomial of degree $p_K+\sigma_K$ (on the master element) is used to approximate the solution to the local problem, then the discrete local problem requires the solution of a system of order $(p_K + \sigma_K + 1)^2$. This system can be fairly large compared to the system of order $(p_K + 1)^2$ equations used to obtain the approximate solution for which we are estimating the error. Since $(p_K + 1)^2$ terms on the right hand side of the discrete local problem (corresponding to (16)) are zero, a simplification is made by approximating the solution to the local problem in the space $Q^{p_K+\sigma_K}(K) \setminus Q^{p_K}(K)$. In other words, the solution to the local problem is approximated with incomplete polynomials of degree $p_K + \sigma_K$ by neglecting the terms associated with polynomials of degree $p_K$. This simplification results in a system of $\sigma_K(\sigma_K + 2p_K + 2)$ equations for each element.

4 The $hp$-Adaptive Strategy

The $hp$-adaptive strategy used here is based on the 3-step strategy developed by Oden, Patra, and Feng [8] for a large class of elliptic problems and, in several applications, was shown to yield exponential rates of convergence with respect to both CPU time and the number of unknowns [9]. Extensions of the $hp$-adaptive strategy to hyperbolic problems were made by Bey, Patra, and Oden [5].

The goal of the adaptive strategy is to deliver a solution with the specified level of error in three adaptive steps: (1) estimate the error in the solution obtained on an initial mesh (2) construct a new mesh using $h$-refinement of the initial mesh, solve the problem on the new mesh, and estimate the error, and (3) enrich the approximation in
regions where the solution is smooth by increasing the spectral order of the elements in the mesh from step (2), and if necessary, perform $h$-refinement in regions where the solution is of low regularity. If the error after step (3) exceeds the specified level, it is necessary to repeat steps (2) and (3) until the desired error is attained.

The $hp$-adaptive strategy is based on the assumption that the $a$ posteriori estimate is a reasonable approximation to the actual error in a particular solution. The $a$ priori estimate (8) and some additional assumptions (see [5]) lead to expressions for estimating the local regularity of the solution and for predicting the mesh required to reduce the error to the specified level. The entire procedure is outlined below.

Detailed development of the $hp$-adaptive strategy can be found in [5].

(i) Specify a target normalized error, $\eta_T$. The target error is normalized by the solution in the same norm. Specify the parameter $\alpha$ to determine the intermediate target error, $\eta_i = \alpha \eta_T$. Specify the parameters $\alpha_s$ and $\alpha_n$ which establish reduction factors for the error in smooth and non-smooth regions as described below. Specify the parameters $\mu_\kappa$ and $\nu_\kappa$ in the $a$ priori estimate (8). Formally these parameters depend on the global regularity of the solution. While there is little theoretical justification, local values can be used by computing the rate of convergence of the local error for a uniform $h$-refinement and $p$-enrichment of a coarse mesh.

(ii) Construct an initial mesh $\mathcal{P}_0$ containing $N(\mathcal{P}_0)$ elements. The elements in $\mathcal{P}_0$ may have uniform $p_\kappa = p_0$ and essentially uniform $h_\kappa \approx h_0$. Find
the approximate solution $\hat{u}_0 \in V_{p_0}(\mathcal{P}_0)$. Estimate the error $\theta_0$ where

$$\theta_0 = \sqrt{\sum_{K \in \mathcal{P}_0} \frac{\theta_{0,K}^2}{\theta_{i}^2}} = \sqrt{\sum_{K \in \mathcal{P}_0} \frac{||\psi_K||^2}{\theta_{i}^2}}$$  \hspace{1cm} (17)

and $\psi_K$ is the solution to the local problem (10).

(iii) Construct a mesh $\mathcal{P}_1$ by subdividing each element in $\mathcal{P}_0$ into the number of elements, $n_K$, required to equally distribute the error and reduce it to $\theta_i = \eta_i(||\hat{u}_0||_{A,U} + \theta_0)$. The number of elements, $n_K$, is obtained by iteratively solving the following two equations:

$$n_K = \left(\frac{\theta_{0,K}}{\theta_{i}^2} N(\mathcal{P}_1)\right)^{\frac{1}{n_{K+1}}}$$  \hspace{1cm} (18)

$$N(\mathcal{P}_1) = \sum_{K \in \mathcal{P}_0} n_K$$  \hspace{1cm} (19)

Find the approximate solution $\hat{u}_1 \in V_{p_0}(\mathcal{P}_1)$ and estimate the error $\theta_1$.

(iv) Estimate the local regularity of the solution by computing the rate of convergence of the local error

$$\mu_K = \frac{\log \theta_{0,K} - \log \sqrt{\sum_{L=1}^{n_K} \theta_{h,L}^2}}{\log h_K - \log \frac{h_K}{\sqrt{A_K}}}, \hspace{0.5cm} K = 1, \ldots, N(\mathcal{P}_0)$$  \hspace{1cm} (20)

The value of $\mu_K$ given by (20) is associated with an element $K$ in the initial mesh and is simply inherited by the new elements generated by
subdividing the element $K$. The expected rate of convergence for smooth solutions is $p_K + \frac{1}{2}$, according to the \textit{a priori} estimate (8). Divide the error into two contributions according to the value of $\mu_K$:

$$
\theta_D = \sqrt{\sum_{K \in \Omega_D} \theta_{1,K}^2}; \quad \Omega_D = \{ K \in P_1 : \mu_K < p_K + \frac{1}{2} \} \tag{21}
$$

$$
\theta_S = \sqrt{\sum_{K \in \Omega_S} \theta_{1,K}^2}; \quad \Omega_S = P_1 \setminus \Omega_D \tag{22}
$$

Subdivide the elements in $\Omega_D$ into the number of elements required to equally distribute the error and reduce it to $\alpha_D \theta_D$. Enrich the approximation in $\Omega_S$ to equally distribute the error and reduce it to $\alpha_s \theta_S$ by increasing $p_K$ according to

$$
p_K = p_0 \left( \frac{\theta_{1,K} \sqrt{N(\Omega_S)}}{\alpha_s \theta_S} \right)^{\frac{1}{p_K}} \tag{23}
$$

Find the approximate solution on the new mesh and estimate the error.

(v) If the estimated error in (iv) is larger than the target error, repeat step (iii) and (iv) until the target error is reached.

In the current implementation, $h$-refinement is accomplished by successive bisection of an element and is limited to two levels for a particular adaptive step. The $h$-refinement in (iv) is necessary only when the error $\theta_D$ exceeds the target error.
5 Parallel Implementation

With the aim of solving more general problems in mind, the linear model problem is solved in a way that is easily extendible to the nonlinear case, takes full advantage of the discontinuous approximation, and is amenable to parallel computations; that is, by solving the time-dependent conservation law for the steady-state solution. Let

\[ u^{n+1} = \hat{u}(t_{n+1}) \text{ where } t_{n+1} = (n + 1)\Delta t \text{ and } \Delta t \text{ is the time step increment.} \]

Assuming the solution at time \( t_n \) is known, then the solution to (1) can be obtained using, for example, the forward Euler time marching scheme:

\[
(u^{n+1}, v)_K = (u^n, v)_K + \Delta t [ L_K(v) - B_K(u^n, v)] \quad \forall v \in V_p(P_h), \forall K \in P_h
\]  

Time-marching versions of the discontinuous Galerkin methods, such as (24), fall naturally into the class of single program multiple data (SPMD) parallel applications. This class of applications are ideally suited for multi-processor distributed-memory architectures.

The primary issue in a parallel implementation of discontinuous Galerkin methods is to balance the workload among the available processors while minimizing the communication between processors, thereby optimizing the utilization of the multi-processor environment. For a machine with \( P \) processors, this is accomplished by decomposing the domain into \( P \) subdomains and assigning the elements contained in a particular subdomain to a particular processor. To minimize communications, the interface of the subdomain boundaries should have as small a measure as possible.

Most domain decomposition methods have been developed and analyzed for \( h-\)
type meshes where the number of degrees of freedom, and hence the computational effort, is the same for every element in the mesh. For these types of meshes, equally distributing the elements among the available processors will usually result in a balanced load. Among the most successful domain decomposition methods for $h$-meshes are those based on recursive bisectioning of either the coordinates or an ordering of the elements. In the recursive bisectioning of the coordinates, trial separators define possible subdomain configurations. The selection of a particular separator as a subdomain interface is based on the resulting load-balance as well as interface size. Vavasis [10] has obtained theoretical bounds on the achievable load balance and interface size for such partitionings based on nodal coordinates. One disadvantage of this approach is that it can be computationally expensive for multiple space dimensions.

Recursive bisectioning methods based on an ordering of the elements have a computational advantage since the bisectioning is performed on a one-dimensional list of elements, regardless of the spatial dimension of the domain. One difficulty with this approach is constructing an ordering which preserves the locality of the elements in the mesh. A locality-preserving ordering is necessary to avoid multiply connected or disconnected subdomains and to minimize interface size. Pothen, Simon, and Liou [11] construct such an ordering by using the second eigenvector of the Laplacian matrix associated with the graph of the mesh. However, the computational cost of deriving the ordering can be high.

For $hp$ meshes, in which the number of degrees of freedom (and hence the computational effort) vary from element to element, the domain decomposition must include some measure of the computational work for each element. The recursive load-based bisection of an ordering (RLBBO) introduced by Patra and Oden [12] is used in this
work. The ordering is generated by the relative location of element centroids on a unit interval obtained by a mapping of a space-filling curve connecting element centroids. The recursive bisection process seeks to balance the load using some local measure of the computational work. In this study, empirical element-computational-time data is used as a measure of its computational load.

For very complex adaptive \( hp \) meshes, the static load balancing algorithms outlined above may not balance the load well, especially if the subdomain sizes become very small. A simple partition smoothing technique is used to improve the load balance. After performing the static decomposition, we carry out a few steps (normally 5) of the time integration and monitor the actual time spent in element solution computations on each processor. If the measured time on processor \( i \) exceeds the measured time on any interface processor \( j \) by 20 percent, then the interface elements on processor \( i \) are re-assigned to processor \( j \).

The obvious drawback in this procedure is that unless the interface elements comprise a small enough part of the total partition, migrating them may actually make the load imbalance worse. Thus the requirement for this procedure to be effective is that the subdomain size (i.e., the problem size) be large enough and scale with the number of processors to maintain partitioning efficiency. Such a requirement in a parallel computing application is common.

Communication between processors on distributed memory machines can significantly effect the overall performance of the parallel implementation, particularly if a processor must wait to receive information from another processor before proceeding with a calculation. To compute the solution at time level \( t_{n+1} \) on a particular subdomain, the solution at \( t_n \) is needed from interface elements on neighboring subdo-
mains. Using synchronous communication strategies, that is, requesting information from neighboring processors at the time that it is needed, leads to unnecessary waiting by all processors. Moreover, communication conflicts are likely to occur since two-way communication is required across interior subdomain boundaries.

Asynchronous communications are used to minimize this wait time. In this type of communication scheme, a request for information is posted before it is actually required and information is sent to all processors that will require it as soon as it is computed. The uncoupled element-by-element nature of the discontinuous Galerkin computations allows for this type of asynchronous communications which completely overlap communication with computations, i.e., a processor computes the solution on another element while the communications required for previous element solutions are being carried out. This overlap is maximized by first computing the solution on the partition interface elements, and then processing those elements on the subdomain interior while the interface element information is simultaneously being communicated to the necessary processors. Hence the communication time which is usually a major bottleneck in scalable parallel computations is almost eliminated.

6 Numerical Results

The parallel implementation of the discontinuous Galerkin method was performed on the Intel Paragon XP/S-5, a distributed-memory machine with 72 processors arranged in a $12 \times 6$ mesh. The method is used to solve the model problem (1) to assess the reliability of the error estimate described in Section 3 and to investigate the

\footnote{The use of trademarks or names of manufacturers in this report is for accurate reporting and does not constitute an official endorsement, either expressed or implied, of such products or manufacturers by the National Aeronautics and Space Administration.}
performance of the hp-adaptive strategy described in Section 4 for a problem with a discontinuous solution. Other numerical examples which verify the a priori estimate and demonstrate the effectiveness of a posteriori estimate can be found in [4]. Other numerical examples demonstrating the effectiveness of the hp-adaptive strategy and the parallel performance can be found in [5].

The following data is used in (1):

(i) $\Omega = (-1, 1) \times (-1, 1)$

(ii) $\beta = (\tfrac{\sqrt{2}}{2}, \tfrac{\sqrt{2}}{2})^T$

(iii) $a(x) = 1.0$

(iv) $g(x, y) = \begin{cases} 
5e^{-[\frac{1}{4}+y^2]} + 3e^{-[1+(y-\frac{1}{2})^2]} & x = -1 \\
-1 - 8e^{-5(x-\frac{1}{2})^2+\frac{1}{4}} & y = -1 
\end{cases}$

The source term $f$ in (1) is chosen so that the exact solution is a function which is discontinuous along the domain diagonal given by

$$u(x, y) = \begin{cases} 
5e^{-[(x+\frac{1}{2})^2+y^2]} + 3e^{-[x^2+(y-\frac{1}{2})^2]} & \text{if } y > x \\
-1 - 8e^{-5(x-\frac{1}{2})^2+(y+\frac{1}{2})^2} & \text{otherwise} 
\end{cases}$$

and shown in Fig. 1.

The global estimated error for a sequence of uniform refinements and for several adaptive hp-meshes is shown in Fig. 2. The labels hp-adaptive in Fig. 2 refer to the strategy described in Section 4 with only $p$-enrichment in the third adaptive step. The labels hhp-adaptive refer to the strategy with both $h$-refinement and $p$-enrichment in the third step. The hp-adaptive strategy delivers nearly linear rates of convergence with respect to the number of degrees of freedom. The rates of convergence (the
Figure 1: Exact solution.

<table>
<thead>
<tr>
<th>u</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
Figure 2: Rates of convergence of the error with the number of unknowns.
slope of the lines in Fig. 2) for the adaptive strategy are higher than the rates of convergence for uniform refinement, indicating that a more accurate solution is obtained with far fewer degrees of freedom when using the hp-strategy. The rate of convergence obtained with the adaptive strategy determines the efficiency of the overall process, and as seen in Fig. 2, the rate of convergence depends significantly on the target and intermediate error specified.

The error history for the hp-adaptive solution denoted by the solid triangles in Fig. 2 is listed in Table 1. The target error was nearly achieved at each step in the adaptive process. The effectivity index (the ratio of the estimated error to the exact error) is on the order of 0.6, quite good for a discontinuous solution, but indicating that the actual error is larger than the estimated error.

Recall that the global error is a sum of element error indicators. The primary source of the under-estimation of the global error is the under-estimation of the element error indicators as shown in Figs. 3 - 5. While the local error estimate provides a qualitative measure of the error at the discontinuity, the low local effectivity index

<table>
<thead>
<tr>
<th>Adaptive step</th>
<th>Target normalized error</th>
<th>Achieved normalized error</th>
<th>Effectivity index</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial 16 x 16 element $p = 1$ mesh</td>
<td>—</td>
<td>0.0722</td>
<td>0.62</td>
</tr>
<tr>
<td>$h$-step</td>
<td>0.036</td>
<td>0.0410</td>
<td>0.58</td>
</tr>
<tr>
<td>$hp$-step</td>
<td>0.022</td>
<td>0.028</td>
<td>0.57</td>
</tr>
</tbody>
</table>

Table 1: Error history for an hp-adaptive solution.
Figure 3: Reliability of the error estimate for the initial mesh.
Figure 4: Reliability of the error estimate for the $h$-step of the adaptive strategy.
Figure 5: Reliability of the error estimate for the $hp$-step of the adaptive strategy.
indicates some severe under-estimation of the error in that region. Note that the local error estimate in smooth regions is very accurate with effectivity indices near unity.

The parallel speedup, defined here as the ratio of the CPU time on 1 processor to the CPU time on \( n \) processors, obtained for the meshes above is shown in Fig. 6. The parallel efficiency of the method, ranging from 95% on 2 processors to 63% on 16 processors, is quite good for \( hp \)-meshes. For the uniform mesh, the RLBBO domain decomposition scheme results in a balanced workload in the sense that it evenly divides the number of elements among the available processors. The departure from ideal speedup for the uniform mesh is caused by the more complex communications patterns that are created by increasing the number of subdomains. The departure from ideal speedup for the \( h \)- and \( hp \)-meshes is caused by a combination of communications and a load imbalance. Exploded views of the RLBBO domain decomposition for the adaptive \( h \)-mesh and \( hp \)-mesh on 8 processors are shown in Figs. 7 and 8 respectively. While the RLBBO domain decomposition results in simply-connected subdomains, complex communications patterns are created at subdomain interfaces due to different levels of \( h \)-refinement and varying spectral orders. The number of elements per processor for the \( h \)-mesh gives a good indication of the load imbalance, since the workload is the same for each element. The maximum number of elements in a subdomain is 151 and the minimum number is 62. For the \( hp \)-mesh, a simple element count or degree-of-freedom count is not adequate to assess the load imbalance, and the actual computational time must be measured. At the end of the dynamic load smoothing, the maximum CPU time for a processor was 19 seconds and the minimum CPU time for a processor was 7 seconds. A more sophisticated migration strategy (see for example Devine [13]) is needed to further improve parallel
Figure 6: Parallel speedup for an adaptive $hp$-mesh.
Figure 7: RLBBO domain decomposition on 8 processors for the mesh obtained after the $h$-step of the adaptive strategy.
Figure 8: RLBBO domain decomposition on 8 processors for the mesh obtained after the $hp$-step of the adaptive strategy.
performance.

7 Concluding Remarks

The development of a parallel $hp$-adaptive discontinuous Galerkin method for hyperbolic conservation laws is presented in this work. The emphasis of the work is on a model class of linear hyperbolic conservation laws for which it is possible to develop $a$ priori error estimates and reliable $a$ posteriori estimates which provide bounds on the actual error. These estimates are obtained using a mesh-dependent norm which reflects the dependence of the error on the local element size and the local order of the approximation.

The $hp$-adaptive strategy is designed to deliver solutions to a specified error level in an efficient way. This is accomplished using a three-step procedure in which the $a$ posteriori estimate is used to determine the error in the solution at a particular adaptive step and the $a$ priori estimate is used to predict the mesh required to deliver a solution with the specified level of error. The $hp$-adaptive strategy makes further use of the $a$ priori estimate to provide detection of discontinuities in the solution thereby identifying regions where $h$-refinement and $p$-enrichment are appropriate.

A parallel implementation of the discontinuous Galerkin method is presented which takes full advantage of the local character of the method and includes a special load-balancing strategy for $hp$-meshes, Recursive Load-Based Bisectioning of an Ordering. This parallel implementation results in nearly optimal speedups when the ratio of interior elements to subdomain interface elements is sufficiently large. The load-balancing strategy, however, becomes less effective as the number of processors is increased.
Numerical experiments demonstrate the effectiveness of the *a posteriori* estimates in providing reliable estimates of the actual error in the numerical solution. While local error estimates near discontinuities under-estimate the actual error, the local error estimates are very accurate in smooth regions. The numerical examples also illustrate the ability of the *hp*-adaptive strategy to deliver a final solution with the specified error. While the *hp*-adaptive strategy provides super-linear convergence rates with respect to the number of unknowns in the problem, the rate of convergence depends on the levels of error requested. More numerical experiments are needed to provide guidelines for selecting the optimum user-specified parameters.

**Acknowledgement** The authors J. Tinsley Oden and Abani Patra acknowledge support of this work by ARPA under Contract number DABT63-92-C-0042 and ARO under Contract number DAAH04-93-G-0109.

**References**


