Alternating evolution discontinuous Galerkin methods for Hamilton–Jacobi equations

Hailiang Liu, Michael Pollack

Iowa State University, Mathematics Department, Ames, IA 50011, United States

ABSTRACT

In this work, we propose a high resolution Alternating Evolution Discontinuous Galerkin (AEDG) method to solve Hamilton–Jacobi equations. The construction of the AEDG method is based on an alternating evolution system of the Hamilton–Jacobi equation, following the previous work Liu et al. (2013) [31] on AE schemes for Hamilton–Jacobi equations. A semi-discrete AEDG scheme derives directly from a sampling of this system on alternating grids. Higher order accuracy is achieved by a combination of high-order polynomial approximation near each grid and a time discretization with matching accuracy. The AEDG methods have the advantage of easy formulation and implementation, and efficient computation of the solution. For the linear equation, we prove the $L^2$ stability of the method. Numerical experiments for a set of Hamilton–Jacobi equations are presented to demonstrate both accuracy and capacity of these AEDG schemes.

© 2013 Elsevier Inc. All rights reserved.

1. Introduction

In this paper, we develop a new alternating evolution discontinuous Galerkin (AEDG) method to solve time-dependent Hamilton–Jacobi (H–J) equations. We describe the designing principle of our AEDG scheme through the following form:

$$\phi_t + H(x, \phi, \nabla_x \phi) = 0, \quad \phi(x, 0) = \phi_0(x), \quad x \in \mathbb{R}^d, \ t > 0. \quad (1.1)$$

Here, $d$ is the space dimension, the unknown $\phi$ is scalar, and $H : \mathbb{R}^d \rightarrow \mathbb{R}^1$ is a nonlinear Hamiltonian. The Hamilton–Jacobi equation arises in many applications ranging from geometrical optics to differential games. These nonlinear equations typically develop discontinuous derivatives even with smooth initial conditions, the solutions of which are non-unique. In this paper, we are only interested in the viscosity solution [11,12,34], which is the physically relevant solution in some important applications.

The difficulty encountered for the satisfactory approximation of the exact solutions of these equations lies in the presence of discontinuities in the solution derivatives. An important class of finite difference methods for computing the viscosity solution is the class of monotone schemes introduced by Crandall–Lions [13]. Unfortunately, monotone schemes are at most first-order accurate. The need for devising more accurate numerical methods for Hamilton–Jacobi equations has prompted the abundant research in this area in the last two decades, including essentially non-oscillatory (ENO) or weighted ENO (WENO) finite difference schemes, see e.g., [15,17,21,27,28,33,36], and Central or Central-Upwind finite difference schemes, see e.g., [2–4,19,20,22,23], as well as discontinuous Galerkin methods [5,16,25,35].

The preliminary goal of this paper is to design high-order discontinuous Galerkin methods for Hamilton–Jacobi equations by following the alternating evolution framework introduced in [24] for hyperbolic conservation laws; see local AE schemes developed by Saran and Liu [30], see also [1]. The high resolution finite difference scheme under this AE framework was
Further developed for Hamilton–Jacobi equations in [31]. In such a framework, the general setting is to refine the original PDE by an alternating evolution (AE) system which involves two representatives: \( u, v \). To update \( u \), terms involving spatial derivatives are replaced by \( v \)'s derivatives and augmented by an additional relaxation term \((v - u)/\epsilon\), which serves to communicate the two representatives. The AE system for scalar hyperbolic conservation laws was shown in [24] to be capable of capturing the exact solution when initially both representatives are chosen as the given initial data. Such a feature allows for a sampling of two representatives over alternating grids. Using this alternating system as a ‘building base’, we apply standard approximation techniques to the AE system.

For Hamilton–Jacobi equations, we consider the following alternating evolution system

\[
\begin{align*}
    u_t + H(x, u, \nabla_x u) &= \frac{1}{\epsilon} (v - u), \\
    v_t + H(x, v, \nabla_x u) &= -\frac{1}{\epsilon} (u - v).
\end{align*}
\]

Here, \( \epsilon > 0 \) is a scale parameter of user’s choice. At \( t = 0 \), we take the initial data

\[ u_0(x) = v_0(x) = \phi_0(x). \]

We sample the above system over alternating grids when performing the spatial discretization, leading to a class of high order semi-discrete AEDG schemes.

The semi-discrete scheme thus obtained has the form

\[
\frac{d}{dt} \int_{I_{\alpha}} \Phi \eta dx + \int_{I_{\alpha}} H(x, \Phi, \nabla_x \Phi^{SN}) \eta dx + \lambda [\Phi] |_{x_0} + \frac{1}{\epsilon} \int_{I_{\alpha}} \Phi \eta dx = \frac{1}{\epsilon} \int_{I_{\alpha}} \Phi^{SN} \eta dx,
\]

where \( \lambda = H_p(x, \Phi(x), \nabla_x \Phi) \), and \( \Phi^{SN} \) are polynomial elements sampled from neighboring cells \( I_{\alpha} \). The scale parameter \( \epsilon \) and the time step size \( \Delta t \) are chosen to stabilize the time discretization. In the one-dimensional case with \( H = \beta p \), the semi-discrete AEDG scheme of first-order becomes

\[
\frac{d}{dt} \Phi_{\alpha} + \beta \left( \Phi_{\alpha+1} - \Phi_{\alpha-1} \right) = \frac{1}{2\Delta x} \left( \Phi_{\alpha+1} - 2\Phi_{\alpha} + \Phi_{\alpha-1} \right),
\]

which, using Euler-forward with \( \epsilon = \Delta t \), reduces to the celebrated Lax–Friedrichs scheme

\[
\Phi_{\alpha}^{n+1} = \frac{1}{2} \left( \Phi_{\alpha+1}^{n} + \Phi_{\alpha-1}^{n} \right) - \epsilon \beta \left( \Phi_{\alpha+1}^{n} - \Phi_{\alpha-1}^{n} \right). \]

We recall that the success of finite difference schemes for Hamilton–Jacobi equations is due to two factors: the local enforcement of the equation in question and the non-oscillatory piecewise polynomial interpolation from evolved grid values. The difference in the AE schemes proposed in [31] and the existing finite difference schemes lies in the local enforcement. The ENO/WENO type schemes [17,21,27,28,32,33,36] are based mainly on some local refinement of H–J equations by

\[
\phi_t + H(x, \phi_x, \phi_{xx}) = 0, \tag{1.2}
\]

where \( \hat{H} \) is the numerical Hamiltonian which needs to be carefully chosen to ensure that the viscosity solution is captured when \( \phi \) becomes discontinuous. In comparison, the central type schemes [2–4,19,20,22,23] choose to evolve the constructed polynomials in smooth regions such that the Taylor expansion may be used in the scheme derivation.

Finite difference methods are quite efficient for Cartesian meshes. However, on unstructured meshes, the scheme is quite complicated. Alternatively, the discontinuous Galerkin (DG) method, originally devised to solve first-order partial differential equations such as conservation laws [6-10,29], has the advantage of flexibility for arbitrarily unstructured meshes, and with the ability to easily achieve arbitrary order of accuracy. However, some new difficulties occur when the existing ideas with finite difference methods are applied toward some discontinuous Galerkin discretization. One main difficulty comes from the non-conservative form of the equation, which precludes the use of integration by parts to establish the cell to cell communication via numerical fluxes as usually done with the DG methods for hyperbolic conservation laws. In spite of this difficulty, some progress has been made. In [16], Hu and Shu proposed a discontinuous Galerkin method to solve the Hamilton–Jacobi equation \( \phi_t + H(\nabla_x \phi) = 0 \). They use the fact that the derivative of the solution \( \phi \) satisfies a conservation law system, and apply the usual discontinuous Galerkin method on this system to advance the derivatives of \( \phi \). The solution \( \phi \) itself is then recovered from these derivatives by a least square procedure for multi-dimensional cases and with an independent evolution of the cell averages of \( \phi \). However, in the multi-dimensional case, a single equation of \( \phi \) is converted to a system of \( w = \nabla_x \phi \), which is only weakly hyperbolic at some points. Also, the resulting algorithm appears indirect and complicated. This motivates the development of another DG method in [5] for directly solving H–J equations; for nonlinear convex Hamiltonians, their numerical experiments demonstrate that the method is stable and accurate, however, entropy correction is needed near the kink formation in some cases. In [35], Yan and Osher apply the LDG discretization to (1.2)
by introducing two additional variables \( p_1 = \phi_1^+ \) and \( p_2 = \phi_2^- \) and then apply the usual discontinuous Galerkin method to advance \((\phi, p_1, p_2)\) all together in the one-dimensional case, with an extension to the two-dimensional case.

In [25], Li and Yakovlev proposed a central discontinuous Galerkin method that directly solves the Hamilton–Jacobi equations using two polynomial representatives solved on two sets of overlapping meshes [26]. The AEDG scheme proposed in this paper only has one polynomial representative associated with each grid point, even in multi-dimensional case. The scheme that we design here is also different in that it is based on alternatively sampling the AE system of the Hamilton–Jacobi equation with spatial accuracy enhanced by interlaced polynomials. However, the AEDG is also similar to the central DG scheme in the sense that neighboring polynomials in the AEDG (the other representative polynomials in the central DG) are used whenever a calculation of the Hamiltonian is needed. For the one-dimensional Hamiltonian \( H(x, \phi, \phi_x) \), the two schemes are almost equivalent when rescaling the mesh size \( \Delta x \) to \( 2\Delta x \), except that in the AEDG scheme the neighboring polynomials are used to compute only the solution gradient \( \phi_x \), while the central DG method uses the other representative polynomials to compute the entire \( H(x, \phi, \phi_x) \). The main advantage of the AEDG scheme is that in the multi-dimensional case, the alternating evolution framework results in easy formulation of the computation of the Hamiltonian. For example, in the two-dimensional case, AEDG evaluates the gradient using the closest polynomials.

It should be noted that even though our AEDG schemes are derived based on sampling the alternating evolution system, we do not solve the system directly. The AE system simply provides a systematic way for developing numerical schemes of both semi-discrete and fully discrete form for the original problem, instead of as an approximation system at the continuous level.

The article is organized as follows: in Section 2, we formulate the AEDG method for one-dimensional Hamilton–Jacobi equations. We also discuss the use of kink detector, and reconstruction of polynomial elements to make the approximation less oscillatory, so that the computed solution is the viscosity solution as desired. We then give a rigorous proof of \( L^2 \) stability by the AEDG method for \( H = \alpha p \) in Section 3. Extensions to the multi-D case is given in Section 4. In Section 5, we show numerical results, which includes both one and two-dimensional problems. The results illustrate accuracy, efficiency, and high resolution near kinks. The last Section 6 ends this paper with our concluding remarks.

2. Alternating evolution DG methods

Our numerical schemes for H–J equations consist of a semi-discrete formulation based on sampling of the AE system on alternating grids and a fully discrete version by using an appropriate Runge–Kutta solver.

To illustrate, we start with the one-dimensional Hamilton–Jacobi equation of the form

\[
\phi_t + H(x, \phi, \phi_x) = 0.
\]

The ‘building base’ is the following AE system

\[
\begin{align*}
  u_t + H(x, u, v_x) &= \frac{1}{\epsilon} (v - u), \\
  v_t + H(x, v, u_x) &= \frac{1}{\epsilon} (u - v).
\end{align*}
\]

2.1. Semi-discrete AEDG formulation

We develop an AE discontinuous Galerkin (AEDG) method for the H–J equation subject to initial data \( \phi_0(x) \). We divide the spatial domain to form a uniform grid, and let \( \{x_j\} \) be the grid point, with \( \Delta x = x_{j+1} - x_j \). We denote \( I_j = (x_{j-1}, x_{j+1}) \) for \( j = 2, \ldots, N - 1 \) with \( I_1 = (x_1, x_2) \) and \( I_N = (x_{N-1}, x_N) \). Centered at each grid \( x_j \), the numerical approximation is a polynomial \( \Phi_{|I_j} = \Phi_j(x) \in P^k \), where \( P^k \) denotes a linear space of all polynomials of degree at most \( k \):

\[
P^k := \left\{ p \mid p(x)|_{I_j} = \sum_{0 \leq i \leq k} a_i (x - x_j)^i, \ a_i \in \mathbb{R} \right\}.
\]

Note \( \dim(P^k) = k + 1 \). We denote \( v(x^+) = \lim_{\epsilon \to 0^+} v(x + \epsilon) \), and \( v_j^\pm = v(x_j^\pm) \). The jump at \( x_j \) is \( [v]_j = v(x_j^+) - v(x_j^-) \), and the average \( \langle v \rangle_j = \frac{1}{2}(v_j^+ + v_j^-) \). Integrating the AE system (2.1) over \( I_j \) against the test function \( \eta \in P^k(I_j) \), we obtain the semi-discrete AEDG scheme

\[
\int_{I_j} (\partial_t \Phi_j + H(x, \Phi_j, \partial_x \Phi_j^{|SN})) \eta \, dx + \lambda \sum_{x_j} \left[ \Phi_j^{|SN} \right] |_{x_j} \eta(x_j) = \frac{1}{\epsilon} \int_{I_j} \langle \Phi_j^{|SN} \rangle \eta \, dx - \int_{I_j} \Phi_j \eta \, dx,
\]

where \( \lambda = H_p(x_j, \Phi_j(x_j), \partial_x \Phi_j(x_j)) \) and \( \Phi_j^{|SN} \) are sampled from neighboring polynomials \( \Phi_{j \pm 1} \) in the following way:
\[
\int_{I_j} H(x, \Phi_j, \partial_x \Phi_j^{SN}) \eta \, dx = \int_{x_{j-1}}^{x_j} H(x, \Phi_j, \partial_x \Phi_{j-1}) \eta \, dx + \int_{x_j}^{x_{j+1}} H(x, \Phi_j, \partial_x \Phi_{j+1}) \eta \, dx,
\]

\[
\int_{I_j} \Phi_j^{SN} \eta \, dx = \int_{x_{j-1}}^{x_j} \Phi_{j-1} \eta \, dx + \int_{x_j}^{x_{j+1}} \Phi_{j+1} \eta \, dx,
\]

\[
[\Phi_j^{SN}]_{x_j} = \Phi_{j+1}(x_j^+) - \Phi_{j-1}(x_j^-).
\]

To update each grid-centered polynomial element \( \Phi \), we write the compact form of the semi-discrete scheme

\[
\frac{d}{dt} \int_{I_j} \Phi_j \eta \, dx = L[\Phi_j; \Phi_j^{SN}, \eta](t),
\]

where

\[
L[\Phi_j; \Phi_{j\pm1}, \eta](t) = \frac{1}{\epsilon} \int_{I_j} (\Phi_j^{SN} - \Phi_j) \eta \, dx - \int_{I_j} H(x, \Phi_j, \partial_x \Phi_j^{SN}) \eta \, dx - \lambda [\Phi_j^{SN}]_{x_j} \eta(x_j).
\]

For cells near the boundary we treat with care. For a computational domain \([a, b] \) with \( x_1 = a, x_N = b \) and \( \Delta x = (b - a)/(N - 1) \), the two equations near boundary may be given as

\[
\int_{x_1}^{x_2} (\partial_t \Phi_1 + H(x, \Phi_1, \partial_x \Phi_2)) \eta \, dx + \frac{1}{2} [\Phi]_{x_1} \eta(x_1) = \frac{1}{\epsilon} \int_{x_1}^{x_2} (\Phi_2 - \Phi_1) \eta \, dx,
\]

\[
\int_{x_{N-1}}^{x_N} (\partial_t \Phi_N + H(x, \Phi_N, \partial_x \Phi_{N-1})) \eta \, dx + \frac{1}{2} [\Phi]_{x_N} \eta(x_N) = \frac{1}{\epsilon} \int_{x_{N-1}}^{x_N} (\Phi_{N-1} - \Phi_N) \eta \, dx.
\]

At the left boundary, \( \eta \in p^k(x_1, x_2) \), and on the right boundary, \( \eta \in p^k(x_{N-1}, x_N) \), so that the length of the interval is \( \Delta x \) instead of \( 2\Delta x \).

If the flow is incoming at \( x = a \), one has to impose a boundary condition \( \phi(a, t) = g_1(t) \). As a result, one is required to modify (2.6) by changing \( [\Phi] \) to \( \Phi_2(x_1^+) - g_1(t) \); for the outflow case, one may take \( [\Phi] = 0 \) in (2.6). Similarly, at \( x = b \), the inflow boundary condition \( \phi(b, t) = g_2(t) \) can be incorporated in (2.7) by replacing \( [\Phi] \) by \( g_2(t) - \Phi_{N-1}(x_N^-) \); and for outgoing flow, replacing \( [\Phi] \) by \( 0 \). The determination of the inflow or outflow of the boundary condition may be obtained by checking the sign of the characteristic speed \( \partial_t H(x, \phi, p) \). In the case of periodic boundary conditions, \( [\Phi] \) can be computed as \( \Phi_2(x_1^+) - \Phi_{N-1}(x_N^-) \) at \( x = x_1, x_N \), and \( \Phi_N(x) \) is regarded to be identical to \( \Phi_1(x) \).

The fully discrete scheme follows from applying an appropriate Runge–Kutta solver to (2.4). We summarize the algorithm as follows.

**Algorithm.**

1. Initialization: in any cell \( I_j \), compute the initial data by the local \( L^2 \)-projection

\[
\int_{I_j} (\Phi^0 - \phi_0) \eta \, dx = 0, \quad \eta \in p^k(I_j).
\]

2. Alternating evaluation: take polynomials \( \Phi_{j\pm1}(x) = \Phi|_{I_{j\pm1}} \), and then sample in \( I_j \) to get \( L[\Phi_j; \Phi_{j\pm1}, \eta] \) as defined in (2.5).
3. Evolution: obtain \( \Phi^{n+1} \) from \( \Phi^n \) by some Runge–Kutta type procedure to solve the ODE system (2.4).

In the AEDG schemes, \( \epsilon \) is chosen such that the stability condition,

\[
\Delta t \leq \epsilon \leq Q \frac{\Delta x}{\max |H_p(x, p)|},
\]

is satisfied. The choice of \( Q \) depends on the order of the scheme, similar to that used in the AE finite difference scheme introduced in [31].
2.2. Discrete ODEs

We now present AE schemes with polynomial elements of degree \( k \) for H–J equations. Let \( \{ \phi_l(\xi) \}_{l=1}^{k+1} \) be the basis in the master domain \([-1, 1]\), then in each cell \( I_j \) we can express

\[
\Phi_j(x) = \sum_{l=1}^{k+1} a_j^l \phi_l(\xi) \quad \xi = \frac{x - x_j}{\Delta x},
\]

then

\[
\Phi_{j+1}(x) = \sum_{l=1}^{k+1} a_{j+1}^l \phi_l(\xi + 1) = \phi^T(\xi + 1) a_{j+1}.
\]

A simple calculation of (2.3) gives

\[
M \dot{a}_j = -\frac{1}{\epsilon} M a_j + \frac{1}{\epsilon} L_j,
\]

where

\[
M = h \int_{-1}^{1} \phi(\xi) \phi^T(\xi) d\xi, \quad L_j = C_- a_{j-1} + C_+ a_{j+1} - \epsilon H_j
\]

with

\[
C_- = h \int_{-1}^{0} \phi(\xi + 1) \phi^T(\xi) d\xi + \epsilon \lambda \phi(0) \phi^T(1),
\]

\[
C_+ = h \int_{0}^{1} \phi(\xi - 1) \phi^T(\xi) d\xi - \epsilon \lambda \phi(0) \phi^T(-1),
\]

\[
H_j = \int_{-1}^{1} \phi(\xi) H(x_j + h\xi, \phi^T(\xi) a_j, h^{-1} \partial_\xi \phi^T(\xi + 1)a_{j-1}) d\xi
\]

\[
+ \int_{0}^{1} \phi(\xi) H(x_j + h\xi, \phi^T(\xi) a_j, h^{-1} \partial_\xi \phi^T(\xi - 1)a_{j+1}) d\xi.
\]

The Hamiltonian \( H \) is generally nonlinear, some quadrature rule must be applied to evaluate \( H_j \). The parameter \( \lambda \) is used to ensure the stability. The following two options are acceptable:

\[
\lambda = \lambda_j = H_p \left( x_j, \Phi_j(x_j), \frac{1}{2} (\partial_x \Phi_{j-1} + \partial_x \Phi_{j+1})(x_j) \right)
\]

or

\[
\lambda = H_p (x_j, \Phi_j(x_j), \partial_x \Phi_j(x_j)).
\]

2.3. Troubled cell detection and limiters

It is known that near singularities one may need to impose some limiters so that the proposed scheme can capture the viscosity solution. To determine which cells are troubled cells and require an application of a limiter, we use a strategy that is based on a strong superconvergence as in [18]. We construct the troubled cell detector in one-dimensional case:

\[
I_j = \frac{\left| \partial_x \phi_{j-1}(x_j^-) - \partial_x \phi_{j+1}(x_j^+) \right|}{h^{k+1/2} \| \phi \|_{L^2(I_j)}}.
\]

The order of the numerator can be determined by the continuity of \( \phi_x \) at the interface \( x = x_j \),

\[
\left| \partial_x \phi_{j-1}(x_j^-) - \partial_x \phi_{j+1}(x_j^+) \right| = \begin{cases} O(h^k), & \text{if } \phi_x \text{ is smooth}, \\ O(1), & \text{if } \phi_x \text{ is discontinuous}. \end{cases}
\]
so that
\[ I_j = \begin{cases} O(h^{3}), & \text{if } \phi_k \text{ is smooth}, \\ O(h^{-\frac{1}{3}}), & \text{if } \phi_k \text{ is discontinuous}. \end{cases} \]
for \( k > 1 \). Then, \( I_j \to 0 \) as \( h \to 0 \) in smooth regions and \( I_j \to \infty \) as \( h \to 0 \) in nonsmooth regions. So, if \( I_j > 1 \), then \( \phi_k \) is discontinuous and the cell \( I_j \) is identified as a troubled cell. A limiter is then applied to that cell to obtain \( \phi_j^{\text{new}} \). If \( I_j < 1 \), then \( \phi_k \) is continuous so that \( \phi_j^{\text{new}} = \phi_j \).

In order to capture the viscosity solution, nonlinear limiters are applied (see [25]) to the detected troubled cells. For \( k = 1 \), \( \phi_j \) is written as
\[ \phi_j = a_j^1 + a_j^2 \left( \frac{x - x_j}{\Delta x} \right). \]
In order to update the \( a_j^2 \) term, we consider two new candidates for the slope of \( \phi_j \),
\[ s_j^L = a_j^1 - a_{j-1}^1, \quad s_j^R = a_{j+1}^1 - a_j^1. \]
We then compute
\[ \tilde{a}_j^2 = \text{MM}(a_j^2, s_j^L, s_j^R). \]
Here \( \text{MM} \) denotes the Min–Mod nonlinear limiter
\[ \text{MM}(b_1, b_2, \ldots) = \begin{cases} \min_j \{b_j\} & \text{if } b_j > 0, \forall j, \\ \max_j \{b_j\} & \text{if } b_j < 0, \forall j, \\ 0 & \text{otherwise}. \end{cases} \]
If \( |a_j^2 - \tilde{a}_j^2| \leq \epsilon \) for small \( \epsilon \), then no change is needed. Otherwise, \( a_j^2 = \tilde{a}_j^2 \).

For \( k = 2 \),
\[ \phi_j = a_j^1 + a_j^2 \left( \frac{x - x_j}{\Delta x} \right) + a_j^3 \left( \frac{x - x_j}{\Delta x} \right)^2. \]
The application of the limiter to \( a_j^2 \) is the same as in the \( k = 1 \) case. Once the \( a_j^2 \) terms have been updated, the \( a_j^3 \) terms are updated in a similar fashion, but for the polynomial \( \Delta x \partial_x \phi_j = a_j^2 + 2a_j^3 \left( \frac{x - x_j}{\Delta x} \right) \). The second derivative approximations can be computed as
\[ (s_j^L)^L = a_j^2 - a_{j-1}^2, \quad (s_j^R)^R = a_{j+1}^2 - a_j^2. \]
We then compute
\[ \tilde{a}_j^3 = \text{MM}(2a_j^3, (s_j^L)^L, (s_j^R)^R). \]
If \( |a_j^3 - \tilde{a}_j^3| \leq \epsilon \) for small \( \epsilon \), then no change is needed. Otherwise, \( a_j^3 = \tilde{a}_j^3/2 \).

### 2.4. Time discretization

We now turn to time discretization of (2.9), following [28]. Let \( \{t^n\}, n = 0, 1, \ldots, K \) be a uniform partition of the time interval \([0, T]\). Let \( \phi^0 = P\phi_0 \) be the piecewise \( L^2 \) projection of \( \phi_0(x) \) defined in (2.8), and \( A = [a_1, \ldots, a_N] \), then
\[ A_t = G(A). \]
We use the third order explicit SSP Runge–Kutta method [14] for time discretization. In details, let \( A^n \) be the solution at time level \( n \),
\begin{align*}
A^{(1)} &= A^n + \Delta t G(A^n), \\
A^{(2)} &= \frac{3}{4} A^{(0)} + \frac{1}{4} (A^{(1)} + \Delta t G(A^{(1)})), \\
A^{n+1} &= \frac{1}{2} A^{(0)} + \frac{2}{3} (A^{(2)} + \Delta t G(A^{(2)})).
\end{align*} (2.10)
3. Stability analysis

In this section we show that the semi-discrete AEDG scheme is $L^2$ stable for linear Hamiltonian. In particular, when $H = \alpha p$, $\alpha = \text{const.}$, we have the following.

**Theorem 3.1.** Let $\Phi$ be computed from the AEDG scheme (2.4) for the Hamilton–Jacobi equation $\phi_t + H(\phi_x) = 0$ with linear Hamiltonian $H(p) = \alpha p$ and periodic boundary conditions. Then

$$\frac{d}{dt} \left( \sum_{j=1}^{N-1} \int_{x_j}^{x_{j+1}} \frac{\Phi_{j+1}^2 + \Phi_j^2}{2} \, dx \right) = -\frac{1}{\epsilon} \sum_{j=1}^{N-1} \int_{x_j}^{x_{j+1}} (\Phi_{j+1} - \Phi_j)^2 \, dx.$$ 

**Proof.** Taking $\eta = \Phi_{2i} =: u$ in (2.4) for $j = 2i$, and $\eta = \Phi_{2i+1} =: v$ in (2.4) for $j = 2i + 1$, respectively, it gives

$$\frac{d}{dt} \int_{l_{2i}}^{l_{2i+1}} \frac{u^2}{2} \, dx + \int_{l_{2i}}^{l_{2i+1}} H(x, u, v, u_x) \, dx + \lambda [v] u|_{x_{2i}} = \frac{1}{\epsilon} \left( \int_{l_{2i}}^{l_{2i+1}} uv \, dx - \int_{l_{2i}}^{l_{2i+1}} u^2 \, dx \right).$$

$$\frac{d}{dt} \int_{l_{2i+1}}^{l_{2i+2}} \frac{v^2}{2} \, dx + \int_{l_{2i+1}}^{l_{2i+2}} H(x, v, u_x) \, dx + \lambda [u] v|_{x_{2i+1}} = \frac{1}{\epsilon} \left( \int_{l_{2i+1}}^{l_{2i+2}} uv \, dx - \int_{l_{2i+1}}^{l_{2i+2}} v^2 \, dx \right).$$

Note that

$$[a, b] = \bigcup_{i=1}^{N/2-1} l_{2i} + [x_{N-1}, x_N] = [x_1, x_2] + \bigcup_{i=1}^{N/2-1} l_{2i+1},$$

we sum over all index and use (2.6)–(2.7) to obtain

$$\frac{d}{dt} \int_{a}^{b} \frac{u^2}{2} \, dx + \int_{a}^{b} \left( \sum_{i=1}^{N/2-1} \int_{l_{2i}}^{l_{2i+1}} H(x, u, v, u_x) \, dx + \sum_{i=1}^{N/2-1} \lambda [v] u|_{x_{2i}} + \frac{1}{2} \lambda [v] u|_{x_N} \right) = \frac{1}{\epsilon} \int_{a}^{b} (uv - u^2) \, dx,$$

$$\frac{d}{dt} \int_{a}^{b} \frac{v^2}{2} \, dx + \int_{a}^{b} \left( \sum_{i=1}^{N/2-1} \int_{x_{2i+1}}^{x_{2i+2}} H(x, v, u_x) \, dx + \frac{1}{2} \lambda [u] v|_{x_1} + \sum_{i=1}^{N/2-1} \lambda [u] v|_{x_{2i+1}} \right) = \frac{1}{\epsilon} \int_{a}^{b} (uv - v^2) \, dx.$$ 

Adding these two relations up, we obtain

$$\frac{d}{dt} \int_{a}^{b} \frac{u^2 + v^2}{2} \, dx + J_1 + J_2 = -\frac{1}{\epsilon} \int_{a}^{b} (u - v)^2 \, dx,$$

where for $H(p) = \alpha p$ all other terms in $J_1$ and $J_2$ are expressed as follows:

$$J_1 = \sum_{i=1}^{N/2-1} \left( \int_{l_{2i}}^{l_{2i+1}} H(x, u, v, u_x) \, dx + \int_{l_{2i}}^{l_{2i+1}} \lambda [v] u|_{x_{2i}} \right) - \alpha \Phi_2(x_1^+) \Phi_1(x_1) - \alpha \Phi_N(x_N) \Phi_{N-1}(x_N),$$

and

$$J_2 = \lambda \left( \sum_{i=1}^{N/2-1} \int_{x_{2i+1}}^{x_{2i}} H(x, v, u_x) \, dx + \int_{x_{2i+1}}^{x_{2i}} \lambda [u] v|_{x_{2i+1}} \right) + \lambda \left( \Phi_1(x_1) + \Phi_N(x_N) \right) \Phi_2(x_1^+) - \Phi_{N-1}(x_N).$$

Using the periodic boundary condition $\Phi_1(x_1) = \Phi_N(x_N)$, we have

$$J_1 + J_2 = 0$$

when $\lambda = \alpha$. $\Box$
4. The multi-dimensional case

By similar procedures we can construct AEDG schemes for multi-dimensional H-J equations:

\[ \phi_t + H(x, \phi, \nabla_x \phi) = 0, \quad x := (x^1, \ldots, x^d) \in \mathbb{R}^d. \]

We start with the AE formulation

\[ u_t + \frac{1}{\epsilon} u = \frac{1}{\epsilon} v - H(x, u, \nabla_x v). \quad (4.1) \]

Let \( \{x_\alpha\} \) be distributed grids in \( \mathbb{R}^d \). Consider \( I_\alpha \) be a hypercube centered at \( x_\alpha \) with vertices at \( x_{\alpha \pm 1} \) where the number of vertices is \( 2^d \).

Centered at each grid \( \{x_\alpha\} \), the numerical approximation is a polynomial \( \Phi|_{I_\alpha} = \Phi_\alpha(x) \in P_r \), where \( P_r \) denotes a linear space of all polynomials of degree at most \( r \) in all \( x_i \):

\[ P_r := \left\{ p \mid p(x)|_{I_\alpha} = \sum_{0 \leq \beta \leq r} a_\beta (x - x_\alpha)^\beta, \quad 1 \leq i \leq d, \quad a_\beta \in \mathbb{R} \right\}. \]

Note \( \dim(P_r) = (r + 1)^d \). Sampling the AE system \((4.1)\) in \( I_\alpha \), we obtain the semi-discrete AEDG scheme

\[
\int_{I_\alpha} \left( \frac{\partial_t \Phi_\alpha}{\epsilon} + H(\Phi_\alpha, \partial_x \Phi_\alpha) \right) \eta \, dx + \frac{1}{\epsilon} \int_{I_\alpha} \Phi_\alpha \eta \, dx = \frac{1}{\epsilon} \int_{I_\alpha} \Phi^{SN}_\alpha \eta \, dx - B, \quad (4.2)
\]

\[
B = \sum_{j=1}^d \int_{I_\alpha} H_j(\Phi_\alpha, \nabla \Phi_\alpha)[\Phi] \eta(x) \delta(x^j - x^j_{\alpha_j}) \, dx, \quad (4.3)
\]

where \( \Phi^{SN}_\alpha \) are sampled from neighboring polynomials, and \( H_j = \partial_{p_j} H(x, z, p) \). The choice of \( \Phi^{SN}_\alpha \) is not unique, and we shall take

\[ \Phi^{SN}_\alpha \in \text{span}\{\Phi_{\alpha \pm \epsilon_j}^d\}_{j=1}^d. \]

In the two-dimensional case with \( \alpha = (i, j) \), the terms involving neighboring polynomials are as follows:

\[
\int_{x_{i-1}}^{x_{i+1}} \int_{y_{j-1}}^{y_{j+1}} H(x, y, \Phi, \partial_x \Phi^{SN}, \partial_y \Phi^{SN}) \eta \, dy \, dx = \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} H(x, y, \Phi_{i,j}, \partial_x \Phi_{i+1,j}, \partial_y \Phi_{i,j+1}) \eta(x, y) \, dy \, dx
\]

\[ + \int_{x_i}^{x_{i+1}} \int_{y_{j-1}}^{y_{j+1}} H(x, y, \Phi_{i,j}, \partial_x \Phi_{i+1,j}, \partial_y \Phi_{i,j-1}) \eta(x, y) \, dy \, dx
\]

\[ + \int_{x_{i-1}}^{x_i} \int_{y_j}^{y_{j+1}} H(x, y, \Phi_{i,j}, \partial_x \Phi_{i-1,j}, \partial_y \Phi_{i,j+1}) \eta(x, y) \, dy \, dx
\]

\[ + \int_{x_{i-1}}^{x_i} \int_{y_{j-1}}^{y_j} H(x, y, \Phi_{i,j}, \partial_x \Phi_{i-1,j}, \partial_y \Phi_{i,j-1}) \eta(x, y) \, dy \, dx. \]

An average of two neighboring polynomials \( \Phi_{i\pm 1,j} \) and \( \Phi_{i,j\pm 1} \) will be used to evaluate \( \int_{I_\alpha} \Phi^{SN}_\alpha \eta \, dx \), that is

\[ \int_{I_\alpha} \Phi^{SN}_\alpha \eta \, dx = \frac{1}{2} \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} (\Phi_{i+1,j} + \Phi_{i,j+1}) \eta(x, y) \, dy \, dx
\]

\[ + \frac{1}{2} \int_{x_i}^{x_{i+1}} \int_{y_{j-1}}^{y_j} (\Phi_{i+1,j} + \Phi_{i,j-1}) \eta(x, y) \, dy \, dx
\]

\[ + \frac{1}{2} \int_{x_{i-1}}^{x_i} \int_{y_j}^{y_{j+1}} (\Phi_{i-1,j} + \Phi_{i,j+1}) \eta(x, y) \, dy \, dx
\]

\[ + \frac{1}{2} \int_{x_{i-1}}^{x_i} \int_{y_{j-1}}^{y_j} (\Phi_{i-1,j} + \Phi_{i,j-1}) \eta(x, y) \, dy \, dx. \]
The B term in (4.3) can then be computed as

\[
B = \int_{y_j}^{y_{j+1}} H_1(x, y, \Phi_{i,j}, \Phi_{i+1,j}) \eta(x, y) dy \\
+ \int_{y_{j-1}}^{y_j} H_2(x, y, \Phi_{i,j}, \Phi_{i+1,j}) \eta(x, y) dy,
\]

For the boundaries, we first consider the side boundary along \(x = x_1\). Then

\[
\int_{x_{j-1}}^{x_{j+1}} \int_{y_{j-1}}^{y_j} H(x, y, \Phi, \partial_x \Phi^{SN}, \partial_y \Phi^{SN}) \eta \, dy \, dx = \int_{x_{j-1}}^{x_{j+1}} \int_{y_{j-1}}^{y_j} H(x, y, \Phi_{i,j}, \partial_x \Phi_{i+1,j}, \partial_y \Phi_{i+1,j}) \eta(x, y) \, dy \, dx \\
+ \int_{x_{j-1}}^{x_{j+1}} \int_{y_{j-1}}^{y_j} H(x, y, \Phi_{i,j}, \partial_x \Phi_{i+1,j}, \partial_y \Phi_{i+1,j}) \eta(x, y) \, dy \, dx,
\]

and

\[
\int_{x_{j-1}}^{x_{j+1}} \int_{y_{j-1}}^{y_j} \Phi^{SN} \eta \, dy \, dx = \frac{1}{2} \int_{y_{j-1}}^{y_j} \int_{x_{j-1}}^{x_{j+1}} (\Phi_{i,j} + \Phi_{i+1,j}) \eta(x, y) \, dy \, dx \\
+ \frac{1}{2} \int_{y_{j-1}}^{y_j} \int_{x_{j-1}}^{x_{j+1}} (\Phi_{i,j} + \Phi_{i+1,j}) \eta(x, y) \, dy \, dx,
\]

and

\[
B = \frac{1}{2} \int_{y_{j-1}}^{y_j} \int_{x_{j-1}}^{x_{j+1}} H_1(x, y, \Phi_{i,j}, \partial_x \Phi_{i,j}, \partial_y \Phi_{i,j}) \eta(x, y) \, dy \, dx \\
+ \int_{y_{j-1}}^{y_j} \int_{x_{j-1}}^{x_{j+1}} H_2(x, y, \Phi_{i,j}, \partial_x \Phi_{i,j}, \partial_y \Phi_{i,j}) \eta(x, y) \, dy \, dx.
\]

In the above, \(\Phi_{0,j}(x_1, y)\) may be taken different ways: the given boundary data at \(x_1\) for inflow boundary; \(\Phi_{2,j}(x_1, y)\) for outgoing flow; and \(\Phi_{N_1,1}(x_0, y)\) for periodic boundary conditions. Similar computations are made along the other side boundaries \(x = x_N\), \(y = y_1, y = y_N\).

At the corners, we first consider the southwest corner \((x_1, y_1)\). Then

\[
\int_{x_1}^{x_2} \int_{y_1}^{y_2} H(x, y, \Phi, \partial_x \Phi^{SN}, \partial_y \Phi^{SN}) \eta \, dy \, dx = \int_{x_1}^{x_2} \int_{y_1}^{y_2} H(x, y, \Phi_{1,1}, \partial_x \Phi_{1,2}, \partial_y \Phi_{1,2}) \eta(x, y) \, dy \, dx,
\]

and

\[
\int_{x_1}^{x_2} \int_{y_1}^{y_2} \Phi^{SN} \eta \, dy \, dx = \frac{1}{2} \int_{x_1}^{x_2} \int_{y_1}^{y_2} (\Phi_{1,1} + \Phi_{1,2}) \eta(x, y) \, dy \, dx.
\]
\[
B = \frac{1}{2} \int_{y_1}^{y_2} H_1(x, y, \Phi_{1,1}, \partial_x \Phi_{1,1}, \partial_y \Phi_{1,1}) (\Phi_{2,1}(x^1, y) - \Phi_{0,1}(x^1, y)) \eta(x, y) \, dy \\
+ \frac{1}{2} \int_{x_1}^{x_2} H_2(x, y, \Phi_{1,1}, \partial_x \Phi_{1,1}, \partial_y \Phi_{1,1}) (\Phi_{1,2}(x, y^1) - \Phi_{1,0}(x, y^1)) \eta(x, y_1) \, dx.
\]

Similar computations can be made for the other corners \((x_1, y_{Ny}), (x_{Nx}, y_1), (x_{Nx}, y_{Ny})\).

Boundary conditions are incorporated in the following ways:

\[
\Phi_{0,1}(x^1, y) = \begin{cases} \text{boundary data} & \text{inflow}, \\ \Phi_{2,1}(x^1, y) & \text{outflow}, \\ \Phi_{Nx-1,1}(x_{Nx}, y) & \text{periodic}, \end{cases}
\]
\[
\Phi_{1,0}(x, y^1) = \begin{cases} \text{boundary data} & \text{inflow}, \\ \Phi_{1,2}(x, y^1) & \text{outflow}, \\ \Phi_{1,Ny-1}(x_1, y_{Ny}) & \text{periodic}. \end{cases}
\]

The SSP Runge–Kutta method [14] can be used to achieve a time discretization with matching accuracy.

5. Numerical tests

In this section, we use some model problems to numerically test the performance of the proposed AEDG scheme in both one dimension and two dimensions. In what follows, we use \(N\) to denote the number of grid points the domain is divided into, i.e., \(\Delta x = (b-a)/(N-1)\) for domain \([a, b]\). The same notation will be also used in two-dimensional case as long as the same partition number is used in both \(x\) and \(y\) direction. Letting the time step at \(t_n\) be \(\Delta t_n\) and \(Q\) to be the CFL number, we then take \(\epsilon\) as

\[
\Delta t_n < \epsilon = \frac{Q \Delta x}{\max |H_p|}.
\]

In the two-dimensional case,

\[
\Delta t_n < \epsilon = Q \left( \frac{\max |H_1|}{\Delta x} + \frac{\max |H_2|}{\Delta y} \right)^{-1}.
\]

The practical choice for \(Q\) is about 0.5 or smaller as \(k\) increases.

We compute the errors as

\[
\|e\|_1 = \sum_i \left[ \sum_j \left| \phi_i(x) - \phi_i^{ex}(x) \right| \right],
\]

\[
\|e\|_\infty = \max_i \max_j |\phi_i(x_j) - \phi_j^{ex}(x_j)|,
\]

where the indice \(i\) corresponds to the computed solution for \(x \in K_i = [x_{i-1/2}, x_{i+1/2}]\) and \(j\) corresponds to the exact solution, or the reference solution with a finer mesh \(K_i'\) such that \(\bigcup_{j=1}^{N} K_i' = K_i\).

Example 5.1. We first test the numerical accuracy of the designed schemes by using the Hamilton–Jacobi equation with convex Hamiltonian and with smooth initial data. The equation is

\[
\phi_t + \frac{(\phi_x + 1)^2}{2} = 0, \quad -1 \leq x \leq 1,
\]

with initial data

\[
\phi(x, 0) = -\cos(\pi x).
\]

In Tables 1–2, the numerical accuracy results using \(P^1\) and \(P^2\) polynomials are presented when the solution is smooth (time \(T = \frac{0.5}{\pi^2}\)). All the proposed methods give the desired order of accuracy.

The solution becomes discontinuous at time \(T = \frac{1}{\pi^2}\), and in Fig. 1, we plot the solutions at time \(T = \frac{1.5}{\pi^2}\). For the numerical experiments the CFL number is taken to be \(0.5\) and \(\Delta t = 0.5\epsilon\).

Example 5.2. The following example is with a non-convex Hamiltonian,

\[
\phi_t + \frac{1}{4} (\phi_x^2 - 1) (\phi_x^2 - 4) = 0, \quad -1 \leq x \leq 1,
\]

\[
\phi(x, 0) = -2|x|.
\]
Table 1
The $L^1$ and $L^\infty$ error for convex Hamiltonian, Example 5.1 using $N$ equally spaced grid points for AEDG scheme using $P^1$ polynomials at $T = \frac{5}{7\pi}$, when the solution is continuous.

<table>
<thead>
<tr>
<th>$N - 1$</th>
<th>Scheme</th>
<th>$L^1$ error</th>
<th>$L^1$ order</th>
<th>$L^\infty$ error</th>
<th>$L^\infty$ order</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>AEDG</td>
<td>0.019367230146</td>
<td>1.92876661</td>
<td>0.029367856297</td>
<td>1.28579317</td>
</tr>
<tr>
<td>40</td>
<td>AEDG</td>
<td>0.005068673182</td>
<td>2.09279280</td>
<td>0.012045068333</td>
<td>1.79411412</td>
</tr>
<tr>
<td>80</td>
<td>AEDG</td>
<td>0.0001192497567</td>
<td>2.07549106</td>
<td>0.0008659545520</td>
<td>2.01461282</td>
</tr>
<tr>
<td>160</td>
<td>AEDG</td>
<td>0.0000282925725</td>
<td>2.05519046</td>
<td>0.0002143892772</td>
<td>2.00334169</td>
</tr>
<tr>
<td>320</td>
<td>AEDG</td>
<td>0.0000068076695</td>
<td>2.08514528</td>
<td>0.000053242772</td>
<td>2.00957476</td>
</tr>
<tr>
<td>640</td>
<td>AEDG</td>
<td>0.0000003700941</td>
<td>2.11605179</td>
<td>0.000013150665</td>
<td>2.01744992</td>
</tr>
<tr>
<td>1280</td>
<td>AEDG</td>
<td>0.0000000116043798</td>
<td>2.13474920</td>
<td>0.0000022752772</td>
<td>2.05849071</td>
</tr>
</tbody>
</table>

Table 2
The $L^1$ and $L^\infty$ error for convex Hamiltonian, Example 5.1 using $N$ equally spaced grid points for AEDG scheme using $P^2$ polynomials at $T = \frac{5}{7\pi}$, when the solution is continuous.

<table>
<thead>
<tr>
<th>$N - 1$</th>
<th>Scheme</th>
<th>$L^1$ error</th>
<th>$L^1$ order</th>
<th>$L^\infty$ error</th>
<th>$L^\infty$ order</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>AEDG</td>
<td>0.001261711385</td>
<td>3.13390617</td>
<td>0.000930253332</td>
<td>3.11959208</td>
</tr>
<tr>
<td>40</td>
<td>AEDG</td>
<td>0.000143734248</td>
<td>3.13474920</td>
<td>0.00011661585</td>
<td>3.05849071</td>
</tr>
<tr>
<td>80</td>
<td>AEDG</td>
<td>0.000016364650</td>
<td>2.99621664</td>
<td>0.000022752723</td>
<td>2.29502185</td>
</tr>
<tr>
<td>160</td>
<td>AEDG</td>
<td>0.000002050953</td>
<td>3.12676552</td>
<td>0.000003160315</td>
<td>2.84789905</td>
</tr>
<tr>
<td>320</td>
<td>AEDG</td>
<td>0.000000234804</td>
<td>3.09258614</td>
<td>0.000000362958</td>
<td>3.12219537</td>
</tr>
</tbody>
</table>

Fig. 1. Comparison of plots for H–J equation with convex Hamiltonian, Example 5.1, at discontinuity on $[-1, 1]$, $T = 1.5/\pi^2$, $N = 81$, using $P^1$ and $P^2$ polynomials.

This test problem is used to show resolution of discontinuities when the initial data has discontinuous derivatives. In testing our AEDG methods, the CFL number used for all order schemes was 0.5. Linear extension boundary conditions are used for the numerical tests. In Fig. 2, we can see that the AEDG scheme with the application of a limiter provides the desired viscosity solution. In Fig. 3, a comparison of the scheme with the applied limiter is shown for $P^1$ and $P^2$ polynomials.

Example 5.3. We next solve a linear Hamiltonian with kink solutions,

$$\Phi_t + \Phi_x = 0,$$

$$\Phi(x, 0) = \Phi_0(x),$$

with $\Phi_0(x) = f(x - 0.5)$ where

$$f(x) = -\left(\frac{\sqrt{3}}{2} + \frac{9}{2} + \frac{2\pi}{3}\right)(x + 1) + \begin{cases} 
2\cos\left(\frac{3\pi x^2}{2}\right) - \sqrt{3}, & -1 \leq x < -\frac{1}{3}, \\
\frac{3}{2} + 3\cos(2\pi x), & -\frac{1}{3} \leq x < 0, \\
\frac{15}{2} - 3\cos(2\pi x), & 0 \leq x < \frac{1}{3}, \\
\frac{28 + 4\pi + \cos(3\pi x)}{3} + 6\pi(x - 1), & \frac{1}{3} \leq x < 1.
\end{cases}$$
Example 5.2. We solve the H–J equation with Riemann initial data, Example 5.2, at discontinuity on $[-1, 1]$, $T = 1$, $N = 81$, $\Delta t = 0.8\epsilon$, using $P^1$ polynomials.

We incorporate periodic boundary conditions on the domain $[-1, 1]$. This example was tested in [35] and [17]. We run the scheme to times $t = 2$ and $t = 32$ for $P^1$ and $P^2$ polynomials, see Figs. 4 and 5.

Example 5.4. We solve the two-dimensional Burgers’ equation using $P^1$ and $P^2$ polynomials

$$\Phi_t + \frac{(\Phi_x + \Phi_y + 1)^2}{2} = 0,$$

with initial data

$$\Phi(x, y, 0) = -\cos(x + y).$$

The computational domain is $[0, 2\pi]^2$ and we impose periodic boundary conditions. The solution is still smooth at time $t = \frac{0.5}{\epsilon^2}$ and we test the order of accuracy at this time in Tables 3 and 4. For $P^1$ and $P^2$ polynomials, we get the expected order in all norms.

Example 5.5.

$$\Phi_t + \sqrt{\Phi_x^2 + \Phi_y^2} + 1 = 0,$$
Fig. 4. Plots for Example 5.3 at times $t = 2$ and $t = 32$ with $N = 201$ using $P^1$ polynomials.

Fig. 5. Plots for Example 5.3 at times $t = 2$ and $t = 32$ with $N = 201$ using $P^2$ polynomials.

Table 3
$L^1$ and $L^\infty$ comparison for Example 5.4 at time $t = 0.5/\pi^2$ using $P^1$ polynomials.

<table>
<thead>
<tr>
<th>$N_x - 1 = N_y - 1$</th>
<th>Scheme</th>
<th>$L^1$ error</th>
<th>$L^1$ order</th>
<th>$L^\infty$ error</th>
<th>$L^\infty$ order</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>AEDG</td>
<td>0.895488976035</td>
<td>0.217832830215</td>
<td>1.697501279</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>AEDG</td>
<td>0.213937258115</td>
<td>0.06716214900</td>
<td>2.067863303</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>AEDG</td>
<td>0.047592740748</td>
<td>0.01601909188</td>
<td>2.172293751</td>
<td></td>
</tr>
<tr>
<td>160</td>
<td>AEDG</td>
<td>0.010144117991</td>
<td>0.00353539842</td>
<td>2.189927533</td>
<td></td>
</tr>
<tr>
<td>320</td>
<td>AEDG</td>
<td>0.0024440585288</td>
<td>0.000778890123</td>
<td>2.350741203</td>
<td></td>
</tr>
<tr>
<td>640</td>
<td>AEDG</td>
<td>0.000523995236</td>
<td>0.000152697730</td>
<td>2.465294309</td>
<td></td>
</tr>
</tbody>
</table>

Table 4
$L^1$ and $L^\infty$ comparison for Example 5.4 at time $t = 0.5/\pi^2$ using $P^2$ polynomials.

<table>
<thead>
<tr>
<th>$N_x - 1 = N_y - 1$</th>
<th>Scheme</th>
<th>$L^1$ error</th>
<th>$L^1$ order</th>
<th>$L^\infty$ error</th>
<th>$L^\infty$ order</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>AEDG</td>
<td>0.115413532776</td>
<td>0.29068625138</td>
<td>3.297975021</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>AEDG</td>
<td>0.014623191059</td>
<td>0.00295732018</td>
<td>3.254344004</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>AEDG</td>
<td>0.002217643937</td>
<td>0.0005611974</td>
<td>2.465294309</td>
<td></td>
</tr>
<tr>
<td>160</td>
<td>AEDG</td>
<td>0.000459523717</td>
<td>0.000824372</td>
<td>2.767140916</td>
<td></td>
</tr>
<tr>
<td>320</td>
<td>AEDG</td>
<td>0.000062313598</td>
<td>0.000152697730</td>
<td>2.350741203</td>
<td></td>
</tr>
</tbody>
</table>
Example 5.5. We now consider an example with smooth initial data

\[ \Phi(x, y, 0) = \frac{1}{4}(\cos(2\pi x) - 1)(\cos(2\pi y) - 1) - 1, \]

and computational domain \([0, 1]^2\) and periodic boundary conditions. The results at time \(t = 0.5\) are shown in Fig. 6. The AEDG scheme provides high resolution in the formation of the singularity.

Example 5.6. We now consider an example with a non-convex Hamiltonian:

\[ \Phi_t - \cos(\Phi_x + \Phi_y + 1) = 0, \]

with initial data

\[ \Phi(x, y, 0) = \cos \left( \frac{\pi (x + y)}{2} \right), \]

on the computation domain \([-2, 2]^2\) with periodic boundary conditions. At time \(t = \frac{0.5}{\pi}\), the solution is still smooth and we test the order of accuracy using \(P^1\) and \(P^2\) polynomials. The results in Tables 5 and 6 show that we get the desired order in all norms. At time \(t = \frac{1}{\pi}\), the solution develops discontinuous derivative and results at time \(t = \frac{1.5}{\pi}\) are shown in Fig. 7.

Example 5.7. We now consider an example relating to controlling optimal cost determination from [25,16] with a nonsmooth Hamiltonian:
Fig. 7. Surface plot for Example 5.6 at time $t = \frac{1.5}{\pi}$ using $P^2$ polynomials, with uniform mesh $\Delta x = \Delta y = \frac{1}{20}$.

\[ \phi_t + \sin(y)\phi_x + (\sin(x) + \text{sign}(\phi_y))\phi_y - \frac{1}{2}\sin^2(y) + \cos(x) - 1 = 0, \]

with initial data
\[ \phi(x, y, 0) = 0, \]

on the computation domain $[-\pi, \pi]^2$ with periodic boundary conditions. The results for the numerical solution and optical control $\text{sign}(\phi_y)$ are shown in Fig. 8 using $P^2$ polynomials and are in agreement with those found in [25,16].

6. Concluding remarks

In this work, we have developed a high-order AEDG scheme to solve the Hamilton–Jacobi equations by following the previous work on high resolution AE finite difference schemes for the Hamilton–Jacobi equation. The AEDG methods have the advantage of easy formulation and implementation, and efficient computation of the solution. High-order accuracy is achieved through the high-order non-oscillatory polynomial approximation and a matching time discretization, numerical experiments in one and two dimensions for $P^1$ and $P^2$ polynomials are presented. The $L^2$ stability of the method was proven for the linear equation.
Acknowledgements

This research was supported by the National Science Foundation under grant DMS13-12636 and the KI-Net research network.

References