HERMITE METHODS FOR THE SCALAR WAVE EQUATION

DANIEL APPELÖ[*], THOMAS HAGSTROM[†], AND ARTURO VARGAS[‡]

Abstract. Arbitrary order dissipative and conservative Hermite methods for the scalar wave equation are presented. Both methods use $(m + 1)^d$ degrees of freedom per node for the displacement in d-dimensions; the dissipative and conservative methods achieve orders of accuracy $(2m - 1)$ and $2m$, respectively. Stability and error analyses as well as implementation strategies for accelerators are also given.

Key words. Wave equation, Hermite methods

AMS subject classifications. 65M70, 65M60

1. Introduction. We construct, analyze, and test arbitrary order dissipative and conservative Hermite methods for the scalar wave equation in a medium with constant speed of sound c. The degrees-of-freedom for Hermite methods are tensor-product Taylor polynomials of degree m in each coordinate centered at the nodes of Cartesian grids, staggered in time. The dissipative method achieves space-time accuracy of order $2m - 1$, while the conservative method has space-time order $2m$. Besides their high order of accuracy in both space and time combined, they have the special feature that they are stable for $c\Delta t \leq h$, for *all orders of accuracy*. This is significantly better than standard high-order element methods. Moreover, the large time steps are purely local to each cell, minimizing communication and storage requirements.

Our primary interest in these schemes are as highly efficient building blocks in hybrid methods where most of the mesh can be taken to be rectilinear and where geometry is handled by more flexible (but less efficient) methods close to physical boundaries. In this work we restrict our consideration to square geometries with boundary conditions of Dirichlet, Neumann or periodic type, where boundary conditions are simple to apply. In previous work [7] we considered this type of hybridization of the Hermite methods for first order system proposed in [11] with nodal discontinuous Galerkin (dG) methods. For wave equations in second order form we envision a similar hybridization where the geometry is handled by, for example, our recently developed discontinuous Galerkin methods for wave equations in second order form [1]. Our dG method has the property that, based on the choice of numerical flux, it is either dissipative or conservative.

We provide optimal stability and convergence results for both the conservative and dissipative method for one dimensional periodic domains. The analysis for the dissipative method follows the analysis for first order systems [11] but here it is based on the energy of the wave equation $\int v^2 + |\nabla u|^2 \, dx$. A difference compared to [11] is that we require that the (polynomial) approximation spaces of the velocity, v, and displacement, u to differ by one degree. Its extension to higher space dimensions, however, does not follow in a straightforward way from the Hermite method for first-order systems, requiring a specialized interpolation scheme to achieve order-independent stability at CFL one.

The analysis of the conservative method is new and quite different from that of [11]. Additionally, the analysis is done by introducing what we denote conserved variables, this simplifies the analysis considerably compared to the classical, [14], way to analyze conservative methods for wave equations. To the best of our knowledge this is an original contribution.

We also note that this paper presents the first application of Hermite methods applied to wave equations in second order form.

Many attractive high-order accurate methods have been proposed for wave equations in second order form. For example there are finite difference methods based on the summation-by-parts framework [16] and upwinding [4, 3], finite element methods which use mass lumping to achieve efficiency [14], discontinuous

Galerkin methods [12, 18, 1, 9], as well as more exotic methods such as Fourier-Continuation [6, 15] and Galerkin differences [2]. However, we believe that the unique properties of Hermite methods make them an interesting alternative to existing approaches.

The rest of the paper is organized as follows. In Section 2 we recall the basic properties of Hermite interpolation which will be used later on. In Section 3 we describe and analyze the dissipative discretization, doing the same in Section 4 for the conservative scheme. In Section 5 we verify the convergence properties in one and two space dimensions and finally in Section 6 demonstrate highly efficient performance of our GPU implementation.

2. Preliminaries. In this section we introduce the grids we will use for the discretizations, discuss the representation of the approximations as Hermite interpolating polynomials and state some of the basic properties of Hermite interpolation. For simplicity we often restrict attention to one or two space dimensions as extensions to higher space dimensions are straightforward.

2.1. Grids. Let the domain $x \in [X_L, X_R]$ be discretized by the grid

$$(2.1) \qquad x_i = X_L + ih_x, \quad h_x = (X_R - X_L)/n_x,$$

and let the two dimensional domain $(x, y) \in [X_L, X_R] \times [Y_B, Y_T]$ be discretized by the grid

$$(2.2) \qquad (x_i, y_j) = (X_L + ih_x, Y_B + jh_y), \quad h_x = \frac{X_R - X_L}{n_x}, \quad h_y = \frac{Y_T - Y_B}{n_y}.$$

We let the indices i, j be integers or half-integers. The primal grid corresponds to the indices $i = 0, \ldots, n_x$, $j = 0, \ldots, n_y$ and the dual grid to $i = 1/2, \ldots, n_x - 1/2$, $j = 1/2, \ldots, n_y - 1/2$.

In time we suppose data is defined on the primal grid when $t_n = n\Delta t$, $n = 0, 1, 2, \ldots$, and on the dual grid when $n = 1/2, 3/2, 5/2, \ldots$. That is, we evolve the solution between grids one half time step $\Delta t/2$ at a time.

2.2. Approximation by Hermite Interpolation. The degrees-of-freedom for our methods are approximations to the solution and its scaled derivatives at the nodes. For example, in one space dimension the data at node x_i is given by

$$c_{l,i}^n \approx \frac{h_x^l}{l!} \frac{\partial^l u}{\partial x^l}(x_i, t_n), \quad l = 0, \ldots, m.$$

Our global approximation to $u(x, t_n)$ is then the piecewise degree $2m + 1$ Hermite interpolant, $p^n(x)$, to the nodal data. Precisely,

$$(2.3) \qquad p^n(x) = p_{i+1/2}^n(x) \equiv \sum_{l=0}^{2m+1} c_{l,0,i+1/2}^n \left(\frac{x - x_{i+1/2}}{h_x} \right)^l, \quad x \in [x_i, x_{i+1}],$$

where $p_{i+1/2}^n(x)$ is the unique polynomial of degree $2m + 1$ satisfying for $l = 0, \ldots, m$

$$(2.4) \qquad \frac{h_x^l}{l!} \frac{\partial^l p_{i+1/2}^n}{\partial x^l}(x_i) = c_{l,i}^n, \quad \frac{h_x^l}{l!} \frac{\partial^l p_{i+1/2}^n}{\partial x^l}(x_{i+1}) = c_{l,i+1}^n.$$

In d spatial dimensions we approximate the solution by tensor product polynomials. For example if $d = 2$ the data at node (x_i, y_j) is

$$c_{k,l,i,j}^n \approx \frac{h_x^k h_y^l}{k!l!} \frac{\partial^{k+l} u}{\partial x^k \partial y^l}(x_i, y_j, t_n), \quad k, l = 0, \ldots m.$$

The global approximation p^n is then given by

$$(2.5) \qquad p^n(x, y) = p_{i+1/2,j+1/2}^n(x, y) \equiv \sum_{k=0}^{2m+1} \sum_{l=0}^{2m+1} c_{k,l,0,i+1/2,j+1/2}^n \left(\frac{x - x_{i+1/2}}{h_x} \right)^k \left(\frac{y - y_{j+1/2}}{h_y} \right)^l,$$

for $(x, y) \in [x_i, x_{i+1}] \times [y_j, y_{j+1}]$, where $p^n_{i+1/2,j+1/2}(x)$ is the unique tensor-product polynomial of degree $(2m + 1, 2m + 1)$ satisfying for $k, l = 0, \ldots, m$, $i_x = i, i + 1$, $j_y = j, j + 1$

$$(2.6) \qquad \frac{h_x^k h_y^l}{k! l!} \frac{\partial^{k+l} u}{\partial x^k \partial y^l}(x_{i_x}, y_{j_y}, t_n) = c^n_{k,l,i_x j_y}.$$

We note that here we are using the same degrees in x and y to approximate u, but we also make use of interpolants with differing degree, $m_x \neq m_y$, in different variables. These will be of degree $(2m_x + 1, 2m_y + 1)$ and are defined by the obvious modifications to (2.5)-(2.6). Such interpolants are also useful for adaptive computations [8].

In what follows we will use the notation $\mathcal{I}_{m_1,\ldots,m_d} f$ to denote the global piecewise polynomial Hermite interpolants defined above of some function f assuming data of the degree m_j in the jth variable. Although these operators depend on whether data is given on the primal or dual grid we suppress this in our notation.

Also, we note that the scaling used here is not strictly necessary but it makes the interpolation and the evaluation of the interpolants (e.g. used for dense output) slightly better conditioned as the matrix entries and coefficients do not vary too much in size.

2.2.1. Properties of Hermite Interpolation. We next recall the basic properties of Hermite interpolation when applied to smooth 2π-periodic functions in d space dimensions. Much of the analysis of the method makes use of the seminorm $|\cdot|_{m_1+1,\ldots,m_d+1}$ defined below. Here we introduce the notation $\mathcal{T}^d = [-\pi, \pi]^d$ and the usual multiindex notation D^α, $\alpha = (\alpha_1, \ldots, \alpha_d)$ for mixed partial derivatives of order α_j in the jth coordinate. In d dimensions this semi-norm is induced by the semi-inner-product:

$$\langle f, g \rangle_{m_1+1,\ldots,m_d+1} = \int_{\mathcal{T}^d} D^\alpha f \cdot D^\alpha g \, dV, \quad \alpha = (m_1 + 1, \ldots, m_d + 1).$$

Note that when $d = 1$ this is simply the order $m + 1$ Sobolev semi-inner-product.

A central observation is that Hermite interpolation is in fact a projection operator in this seminorm. Precisely, as shown in [11]:

$$(2.7) \qquad \langle \mathcal{I}_{m_1,\ldots,m_d} u, v - \mathcal{I}_{m_1,\ldots,m_d} v \rangle_{m_1+1,\ldots,m_d+1} = 0.$$

From (2.7) we deduce the Pythagorean Theorem:

$$(2.8) \qquad |u|^2_{m_1+1,\ldots,m_d+1} = |\mathcal{I}_{m_1,\ldots,m_d} u|^2_{m_1+1,\ldots,m_d+1} + |u - \mathcal{I}_{m_1,\ldots,m_d} u|^2_{m_1+1,\ldots,m_d+1},$$

as well as the fact that the interpolation operators are self-adjoint:

$$\begin{aligned}
\langle \mathcal{I}_{m_1,\ldots,m_d} u, v \rangle_{m_1+1,\ldots,m_d+1} &= \langle \mathcal{I}_{m_1,\ldots,m_d} u, \mathcal{I}_{m_1,\ldots,m_d} v \rangle_{m_1+1,\ldots,m_d+1} + \langle \mathcal{I}_{m_1,\ldots,m_d} u, v - \mathcal{I}_{m_1,\ldots,m_d} v \rangle_{m_1+1,\ldots,m_d+1} \\
&= \langle u, \mathcal{I}_{m_1,\ldots,m_d} v \rangle_{m_1+1,\ldots,m_d+1} + \langle \mathcal{I}_{m_1,\ldots,m_d} u - u, \mathcal{I}_{m_1,\ldots,m_d} v \rangle_{m_1+1,\ldots,m_d+1} \\
&= \langle u, \mathcal{I}_{m_1,\ldots,m_d} v \rangle_{m_1+1,\ldots,m_d+1}.
\end{aligned}$$

Denote the L_2-inner product and norm by

$$(f, g) = \int_{\mathcal{T}^d} f(x)^T g(x) \, dV, \quad \|g\|^2 = (g, g),$$

and the Sobolev norms on H^q_{per} by

$$\|f\|^2_{H^q} = \sum_{|\alpha| \leq q} \|D^\alpha\|^2.$$

Then, for $d = 1$, the following lemma (Lemma 3.1 and Corollary 3.1 in [11]) details the basic approximation properties of Hermite interpolation. We note that these results are derived locally on each subinterval and thus generally extend to nonperiodic functions.

3

LEMMA 2.1. *The Hermite interpolation operator $\mathcal{I}_m$ satisfies:*

$$(2.9) \qquad \|g - \mathcal{I}_m g\| \le C h_x^{2m+2} \left\| \frac{d^{2m+2}g}{dx^{2m+2}} \right\| \quad for \;\; g \in H_{\mathrm{per}}^{2m+2},$$

$$(2.10) \qquad \| \frac{d^s}{dx^s}(g - \mathcal{I}_m g)\| \le C h_x^{2m+2-s} \left\| \frac{d^{2m+2}g}{dx^{2m+2}} \right\| \quad for \;\; g \in H_{\mathrm{per}}^{2m+2}, \;\; s \le 2m+2,$$

$$(2.11) \qquad \|g - \mathcal{I}_m g\| \le C h_x^{m+1} \left\| \frac{d^{m+1}g}{dx^{m+1}} \right\| \quad for \;\; g \in H_{\mathrm{per}}^{m+1},$$

$$(2.12) \qquad \|(f - \mathcal{I}_m f)\| \le C h_x^{m+1} \left\| \frac{d^{m+1}}{dx^{m+1}}(f - \mathcal{I}_m f) \right\| \quad for \;\; f \in H_{\mathrm{per}}^{m+1}.$$

The estimates (2.9) - (2.11) follow directly from the Peano kernel representation, see [5], and (2.12) follows from (2.11) with $g = f - \mathcal{I}_m f$, as for that g it holds that $\mathcal{I}_m g = 0$. We note that for $d > 1$ the analogous results require the use of more complicated Sobolev seminorms (Lemma 6.1 in [11]):

$$|f|_{[r_1,\dots r_d]}^2 \equiv \sum_{\alpha \in \mathcal{Q}_r} \|D^\alpha f\|^2,$$

$$\mathcal{Q}_r = \{\alpha | \alpha_j \in \{0, r_j\}, \; |\alpha| > 0\}.$$

Lastly we note that the leading order error in Hermite interpolation for sufficiently smooth functions has a particularly simple form which is the same in all dimensions. Here for ease of notation we suppose the cell center is the coordinate origin so that the nodes are $(\pm h/2, \dots, \pm h/2)$. Then, using $(x_1, \dots, x_d)$ to denote the coordinates and assuming equal orders in all variables, a multi-point Taylor series expansion yields

$$(2.13) \qquad \mathcal{I}_{m,\dots,m} f = f - \frac{1}{(2m+2)!} \sum_{j=1}^d \left(x_j^2 - \frac{h^2}{4} \right)^{m+1} \frac{\partial^{2m+2} f}{\partial x_j^{2m+2}}(0, \dots, 0) + O(h^{2m+3}).$$

3. Dissipative Hermite Method for the Scalar Wave Equation. We now describe our dissipative discretization for the scalar wave equation with constant speed, c. Here we approximate the wave equation, in $d+1$ dimensions, as a first order system in time for the displacement u and the velocity v

$$(3.1) \qquad \frac{\partial u}{\partial t} = v,$$

$$(3.2) \qquad \frac{\partial v}{\partial t} = c^2 \Delta u + f,$$

with initial data

$$(3.3) \qquad u(x, 0) = g_0(x), \;\; v(x, 0) = g_1(x).$$

In this work we will consider periodic, homogenous Dirichlet, or homogeneous Neumann boundary conditions. In one space dimension the method is essentially equivalent to the original method in [11] applied to the 2×2 system for $(\frac{\partial u}{\partial x}, v)$ followed by a reconstruction of u, but in multiple space dimensions additional modifications are proposed to maintain the method's time step stability properties.

3.1. One Space Dimension. This section outlines the dissipative method in one dimension. A schematic picture of a full time step can be found in Figure 3.1.

3.1.1. Initialization. We start by approximating u and v on the primal grid at the initial time, t_0. As discussed above, this amounts to setting approximate values of derivatives of the functions g_0 and g_1 at the nodes:

$$c_{l,i}^0 \approx \frac{h_x^l}{l!} \frac{d^l g_0}{dx^l}(x_i), \;\; l = 0, \dots m, \;\; i = 0, \dots, n_x,$$
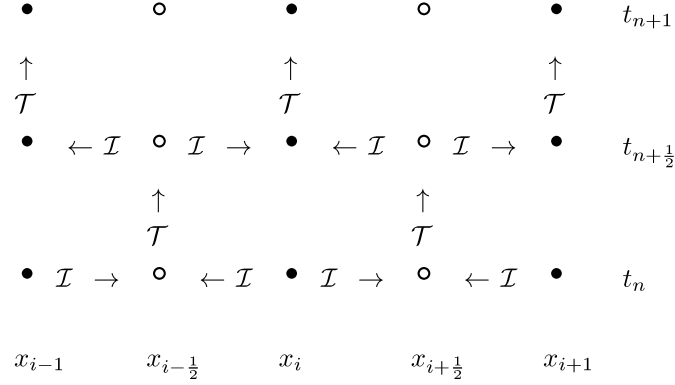
Fig. 3.1: A schematic picture of the steps in the dissipative method. Solid circles represent the primal grid and open circles represent the dual grid. $\mathcal{I}$ is the Hermite interpolation operator and $\mathcal{T}$ is the time evolution operator.

$$d_{l,i}^0 \approx \frac{h_x^l}{l!} \frac{d^l g_1}{dx^l}(x_i), \quad l = 0, \ldots m-1, \quad i = 0, \ldots, n_x.$$

Here the data $c_{l,i}^n$ will define the approximations, p^n, to u and $d_{l,i}^n$ the approximations, q^n, to v. Ideally we would use exact values of the derivatives but alternatively they can be obtained by interpolation or projection of the initial data. Note that derivatives of order m are used in the approximation of u, higher than for the approximation of v. The reason for this choice will be clear from the analysis presented in Section 3.2.

3.1.2. Evolution. The first step in evolving the approximate solution to the time $t = t_0 + \Delta t/2$ is to construct the piecewise Hermite interpolants of the data (2.3):

$$p_{i+\frac{1}{2}}^0(x) = \sum_{l=0}^{2m+1} c_{l,0,i+\frac{1}{2}}^0 \left(\frac{x - x_{i+\frac{1}{2}}}{h_x} \right)^l, \quad \forall x \in [x_i, x_{i+1}],$$

$$q_{i+\frac{1}{2}}^0(x) = \sum_{l=0}^{2m-1} d_{l,0,i+\frac{1}{2}}^0 \left(\frac{x - x_{i+\frac{1}{2}}}{h_x} \right)^l, \quad \forall x \in [x_i, x_{i+1}],$$

where the coefficients $c_{l,0,i+\frac{1}{2}}$ and $d_{l,0,i+\frac{1}{2}}$ are uniquely determined by the interpolation conditions (2.4).

The second step is to expand the new polynomials in time

$$p_{i+\frac{1}{2}}^0(x,t) = \sum_{l=0}^{2m+1} \sum_{s=0}^{\kappa_u(l,m)} c_{l,s,i+\frac{1}{2}}^0 \left(\frac{x - x_{i+\frac{1}{2}}}{h_x} \right)^l \left(\frac{t - t_0}{\Delta t} \right)^s,$$

$$q_{i+\frac{1}{2}}^0(x,t) = \sum_{l=0}^{2m-1} \sum_{s=0}^{\kappa_v(l,m)} d_{l,s,i+\frac{1}{2}}^0 \left(\frac{x - x_{i+\frac{1}{2}}}{h_x} \right)^l \left(\frac{t - t_0}{\Delta t} \right)^s.$$

The coefficients with $s > 0$ are obtained by insisting that the approximation satisfy equations (3.1) and (3.2), and equations obtained by repeated differentiation in space and time of (3.1) and (3.2). Precisely we

5

ask that at $(x_{i+\frac{1}{2}}, t_0)$ the approximations satisfy

$$\frac{\partial^{l+s} p^0_{i+\frac{1}{2}}}{\partial x^l \partial t^s} = \frac{\partial^{l+s-1} q^0_{i+\frac{1}{2}}}{\partial x^l \partial t^{s-1}},$$

$$\frac{\partial^{l+s} q^0_{i+\frac{1}{2}}}{\partial x^l \partial t^s} = c^2 \frac{\partial^{l+s+1} p^0_{i+\frac{1}{2}}}{\partial x^{l+2} \partial t^{s-1}} + \frac{\partial^{l+s-1} f}{\partial x^l \partial t^{s-1}}.$$

Expressing these equations as truncating recursions for the coefficients we obtain

$$(3.4) \qquad c^0_{l,s,i+\frac{1}{2}} = \frac{\Delta t}{s} d^0_{l,s-1,i+\frac{1}{2}},$$

$$(3.5) \qquad d^0_{l,s,i+\frac{1}{2}} = c^2 \frac{(l+2)(l+1)}{s} \frac{\Delta t}{h_x^2} c^0_{l+2,s-1,i+\frac{1}{2}} + \frac{h_x^l}{l!} \frac{\Delta t^s}{s!} \frac{\partial^{l+s-1} f}{\partial x^l \partial t^{s-1}}.$$

The upper limits of the temporal Taylor series, κ_u and κ_v are chosen so that, for $f = 0$, the Taylor series truncate which means that $\kappa_v(l,m) = 2m - 1 - 2\lfloor l/2 \rfloor$, and $\kappa_u(l,m) = \kappa_v(l,m) + 1$. Here the notation $\lfloor x \rfloor$ is rounding to the nearest smaller integer ($\lfloor x \rfloor$ = "floor of x"). The fact that the Taylor series truncates means that for $f = 0$ the polynomial data is *evolved exactly*. Moreover, as long as $c\Delta t \leq h_x$, the evolution at the cell center is exact for the piecewise polynomial data, a fact which is exploited in our analysis.

Once the coefficients above have been computed we find the approximate solution at the dual nodes at the half time step by simply evaluating the polynomials and their derivatives. The initial data for the next half time step are thus

$$(3.6) \qquad c^{1/2}_{l,i+1/2} = \frac{h_x^l}{l!} \frac{\partial^l}{\partial x^l} p_{i+\frac{1}{2}}(x_{i+\frac{1}{2}}, t_{\frac{1}{2}}), \quad l = 0, \ldots, m,$$

$$(3.7) \qquad d^{1/2}_{l,i+1/2} = \frac{h_x^l}{l!} \frac{\partial^l}{\partial x^l} q_{i+\frac{1}{2}}(x_{i+\frac{1}{2}}, t_{\frac{1}{2}}), \quad l = 0, \ldots, m-1.$$

3.1.3. Boundary Conditions for the Second Half-step. To complete a full time step the procedure is repeated, starting with the data (3.6) and (3.7). At the interior nodes $x_i, i = 1, \ldots, n_x - 1$ the procedure is the same as above but at the nodes x_0 and x_{n_x} we must use the boundary conditions to fill in ghost-polynomials at $x_{-1/2}$ and $x_{n_x+1/2}$. For periodic problems this is straightforward, but for other boundary conditions this requires additional calculations.

Fundamentally, we must determine $2(m+1)$ unknowns that specify the polynomial at a boundary, say, at $x = x_0$. Requiring that the degree $(2m+1)$ polynomial centered at x_0 interpolates the $(m+1)$ data at $x_{\frac{1}{2}}$ yields $(m+1)$ independent linear equations. The first of the remaining $(m+1)$ independent linear equations can be obtained by requiring that the polynomial coincides with the boundary condition, say $u(0,t) = g(t)$. Additional independent equations on the even derivatives of u and v can be obtained by differentiating the boundary condition in time together with the PDE. For example $g_{tt}(t) = u_{tt}(0,t) = u_{xx}(0,t)$ and $g_{ttv}(t) = u_{ttv}(0,t) = v_{xx}(0,t)$ and so forth.

Practically, we "solve" these equations by simply extending the polynomial from the interior. For example, if the boundary condition is of Dirichlet type and constant in time we first specify the ghost polynomial so that the coefficients associated with even powers of x of the interpolating polynomial (that is centered at the boundary) are all zero. If the boundary condition is non-zero we then adjust the constant term in the interpolating polynomial to agree with the value of the boundary condition. Neumann conditions are handled similarly but with an even extension of the interior polynomial to the ghost polynomial.

We note that this approach to enforcing boundary conditions is often referred to as compatibility boundary conditions and has been used, for example, by Henshaw [13] in the context of solving Maxwell's equations in second order form.

3.2. Convergence in One Space Dimension. We now analyze the convergence properties of the method described above. We consider 2π-periodic solutions to the wave equation (3.1)-(3.2) with $f = 0$, for

which the natural energy estimate is

$$(3.8) \qquad \frac{1}{2}\frac{d}{dt}\int_{-\pi}^{\pi}c^2\left(\frac{\partial u}{\partial x}\right)^2+v^2 dx=0.$$

To follow the analysis in [11] we will consider the above energy in the $|\cdot|_m$ semi-norm where the interpolation is stable and dissipative in the sense

$$(3.9) \qquad c^2\left|\frac{\partial \mathcal{I}_m u}{\partial x}\right|_m^2+|\mathcal{I}_{m-1}v|_m^2\le c^2\left|\frac{\partial u}{\partial x}\right|_m^2+|v|_m^2.$$

This inequality is a direct consequence of (2.8) as $\left|\frac{\partial w}{\partial x}\right|_m^2=|w|_{m+1}^2$ for any $w(x)$. We note that this inequality is the motivation for approximating u and v by polynomials of differing degree. We also note that the analysis can be applied to problems with homogeneous Dirichlet and Neumann conditions with compatible initial data by extending them smoothly to periodic problems on larger domains.

Now, assuming that the time step satisfies $c\Delta t\le h_x$ we can write the method outlined above as an evolution of the interpolants of the basic degrees of freedom. Precisely:

i. Interpolate the initial data $p^0=\mathcal{I}_m g_0$, $q^0=\mathcal{I}_{m-1}g_1$.

ii. For $n=0,\frac{1}{2},1,\dots$, evolve by half steps

$$(3.10) \qquad \left(\begin{array}{c} p^{n+\frac{1}{2}} \\ q^{n+\frac{1}{2}} \end{array}\right)=\left(\begin{array}{cc} \mathcal{I}_m & 0 \\ 0 & \mathcal{I}_{m-1} \end{array}\right)\mathcal{S}\left(\begin{array}{c} p^n \\ q^n \end{array}\right).$$

Here $\mathcal{S}$ refers to the exact evolution operators over a half time step. As noted above, for polynomial data, these coincide with the recursion relations (3.4) and (3.5) followed by the evaluation of the Taylor series (3.6) and (3.7) as long as the CFL condition $c\Delta t\le h_x$ is satisfied.

We now state and prove our main convergence result for the dissipative method.

THEOREM 1. *Let $1>c\Delta t/h_x>0$ and let $T>0$ be fixed. Suppose $g_0\in H_{\mathrm{per}}^{2m+1}$, $g_1\in H_{\mathrm{per}}^{2m}$ then there is a constant C, independent of h_x, so that*

$$(3.11) \qquad \left\|\frac{d}{dx}(u^n-p^n)\right\|+\|v^n-q^n\|\le Ch_x^{2m-1}\left(\|\frac{d^{2m+1}g_0}{dx^{2m+1}}\|+\|\frac{d^{2m}g_1}{dx^{2m}}\|\right).$$

Proof. We start by defining the local truncation errors η and ξ

$$\left(\begin{array}{c} u^{n+\frac{1}{2}} \\ v^{n+\frac{1}{2}} \end{array}\right)=\left(\begin{array}{cc} \mathcal{I}_m & 0 \\ 0 & \mathcal{I}_{m-1} \end{array}\right)\mathcal{S}\left(\begin{array}{c} u^n \\ v^n \end{array}\right)+\left(\begin{array}{c} \eta^n \\ \xi^n, \end{array}\right).$$

Note that, as the evolution operator $\mathcal{S}$ is exact, the truncation errors are simply the interpolation errors

$$\eta^n=u^{n+\frac{1}{2}}-\mathcal{I}_m u^{n+\frac{1}{2}},\quad \xi^n=v^{n+\frac{1}{2}}-\mathcal{I}_{m-1}v^{n+\frac{1}{2}}.$$

We next define the errors

$$e_u^n=u^n-p^n,\quad e_v^n=v^n-q^n,$$

which are governed by the error equations

$$(3.12) \qquad \left(\begin{array}{c} e_u^{n+\frac{1}{2}} \\ e_v^{n+\frac{1}{2}} \end{array}\right)=\left(\begin{array}{cc} \mathcal{I}_m & 0 \\ 0 & \mathcal{I}_{m-1} \end{array}\right)\mathcal{S}\left(\begin{array}{c} e_u^n \\ e_v^n \end{array}\right)+\left(\begin{array}{c} \eta^n \\ \xi^n, \end{array}\right),$$

with initial data

$$e_u^0=g_0-\mathcal{I}_m g_0,\quad e_v^0=g_1-\mathcal{I}_{m-1}g_1.$$

Now, to obtain the estimate in the theorem we will first estimate the energy error in the semi-norm where we can make use of the orthogonality (2.8). We start by differentiating $e_u^{n+\frac{1}{2}}$ $m+1$ times and $e_v^{n+\frac{1}{2}}$ m times

$$(3.13) \qquad \left(\begin{array}{c} \frac{d^{m+1}e_u^{n+\frac{1}{2}}}{dx^{m+1}} \\ \frac{d^m e_v^{n+\frac{1}{2}}}{dx^m} \end{array}\right)=\left(\begin{array}{cc} \frac{d^{m+1}}{dx^{m+1}}\mathcal{I}_m & 0 \\ 0 & \frac{d^m}{dx^m}\mathcal{I}_{m-1} \end{array}\right)\mathcal{S}\left(\begin{array}{c} e_u^n \\ e_v^n \end{array}\right)+\left(\begin{array}{c} \frac{d^{m+1}}{dx^{m+1}}\eta^n \\ \frac{d^m}{dx^m}\xi^n, \end{array}\right).$$

7

Recalling that the local truncation errors are interpolation errors we can proceed by taking the L^2 norms of (3.13). Then, using the orthogonality (2.8) we rewrite

$$\mathcal{S}\left(\begin{array}{c} e_u^n \\ e_v^n \end{array}\right) = \left(\begin{array}{c} \mathcal{S}_u E^n \\ \mathcal{S}_v E^n \end{array}\right), \quad E^n = \left(\begin{array}{c} e_u^n \\ e_v^n \end{array}\right),$$

and arrive at

$$(3.14) \qquad c^2 \left| \frac{de_u^{n+\frac{1}{2}}}{dx} \right|_m^2 + \left| e_v^{n+\frac{1}{2}} \right|_m^2 = c^2 \left| \frac{d\mathcal{I}_m \mathcal{S}_u E^n}{dx} \right|_m^2 + |\mathcal{I}_{m-1}\mathcal{S}_v E^n|_m^2 + c^2 \left| \frac{d\eta^n}{dx} \right|_m^2 + |\xi^n|_m^2 .$$

To proceed we use the identities

$$\frac{d^{m+1}}{dx^{m+1}}\mathcal{S}_u E^n = \frac{d^{m+1}}{dx^{m+1}}\mathcal{I}_m \mathcal{S}_u E^n - \frac{d^{m+1}}{dx^{m+1}}(\mathcal{I}_m \mathcal{S}_u E^n - \mathcal{S}_u E^n),$$

$$\frac{d^m}{dx^m}\mathcal{S}_v E^n = \frac{d^m}{dx^m}\mathcal{I}_{m-1}\mathcal{S}_v E^n - \frac{d^m}{dx^m}(\mathcal{I}_{m-1}\mathcal{S}_u E^n - \mathcal{S}_v E^n),$$

together with (2.8) to obtain:

$$(3.15) \; c^2 \left| \frac{d\mathcal{I}_m \mathcal{S}_u E^n}{dx} \right|_m^2 + |\mathcal{I}_{m-1}\mathcal{S}_v E^n|_m^2 = c^2 \left| \frac{d\mathcal{S}_u E^n}{dx} \right|_m^2 + |\mathcal{S}_v E^n|_m^2$$

$$-c^2 \left| \frac{d(\mathcal{S}_u E^n - \mathcal{I}_m \mathcal{S}_u E^n)}{dx} \right|_m^2 - |\mathcal{S}_v E^n - \mathcal{I}_{m-1}\mathcal{S}_v E^n|_m^2 .$$

Next we introduce the semi-norm energy error

$$(3.16) \qquad \mathcal{E}_m^n \equiv c^2 \left| \frac{de_u^n}{dx} \right|_m^2 + |e_v^n|_m^2 ,$$

and note the fundamental fact that the energy measured in any Sobolev semi-norm is preserved by the exact solution operator $\mathcal{S}$:

$$(3.17) \qquad c^2 \left| \frac{d\mathcal{S}_u E^n}{dx} \right|_m^2 + |\mathcal{S}_v E^n|_m^2 = \mathcal{E}_m^n.$$

Now, a direct estimate in terms of $\mathcal{E}_m^n$ does not give a sharp error estimate unless we carefully track the cancellation of errors between time-levels. To do so we first introduce δ^n by rewriting (3.14) using (2.10)

$$\delta^n = \left(c^2 \left| \frac{d(\mathcal{S}_u E^n - \mathcal{I}_m \mathcal{S}_u E^n)}{dx} \right|_m^2 + |\mathcal{S}_v E^n - \mathcal{I}_{m-1}\mathcal{S}_v E^n|_m^2 \right)^{1/2} .$$

We then use (3.15)-(3.17) and find:

$$\mathcal{E}_m^{n+1/2} = \mathcal{E}_m^n - (\delta^n)^2 + O(h_x^{2m}).$$

To expose telescoping we separate $(\delta^n)^2$ on the left

$$(3.18) \qquad (\delta^n)^2 = \mathcal{E}_m^n - \mathcal{E}_m^{n+1/2} + O(h_x^{2m}).$$

Now, summing (3.18) to the final time $J = T/\Delta t$ we conclude

$$\sum_{n=0}^{J-1/2} (\delta^n)^2 = \mathcal{E}_m^0 - \mathcal{E}_m^J + O(h_x^{2m-1})$$

$$(3.19) \qquad = O(h_x^{2m-1}),$$

so long as the initial conditions are approximated with sufficient accuracy.

To prove the theorem we must connect the estimates in the semi-norm to L_2-estimates. Taking one derivative of $e_u^{n+\frac{1}{2}}$ we have

(3.20)
$$\begin{pmatrix} \frac{d}{dx} e_u^{n+\frac{1}{2}} \\ e_v^{n+\frac{1}{2}} \end{pmatrix} = \begin{pmatrix} \frac{d}{dx} \mathcal{I}_m \mathcal{S}_u E^n \\ \mathcal{I}_{m-1} \mathcal{S}_v E^n \end{pmatrix} + \begin{pmatrix} \frac{d\eta^n}{dx} \\ \xi^n, \end{pmatrix}.$$

Invoking the triangle inequality and (2.9) and defining

$$\mathcal{E}^n = c^2 \| \frac{d}{dx} e_u^n \|^2 + \| e_v^n \|^2$$

we have

$$\left(\mathcal{E}^{n+\frac{1}{2}} \right)^{1/2} \leq \left(c^2 \| \frac{d}{dx} \mathcal{S}_u E^n \|^2 + \| \mathcal{S}_v E^n \|^2 \right)^{1/2} + \left(c^2 \| \frac{d}{dx} \left((\mathcal{I}_m \mathcal{S}_u - \mathcal{S}_u) E^n \right) \|^2 + \| (\mathcal{I}_{m-1} \mathcal{S}_v - \mathcal{S}_v) E^n \|^2 \right)^{1/2}$$
$$+ O(h_x^{2m}).$$

Now using the fact that $\mathcal{S}$ preserves the energy and invoking (2.12)

$$\left(\mathcal{E}^{n+\frac{1}{2}} \right)^{1/2} \leq (\mathcal{E}^n)^{1/2} + C h^m \delta^n + O(h_x^{2m}).$$

We sum and make use of (3.19)

$$\left(\mathcal{E}^{n+\frac{1}{2}} \right)^{1/2} \leq C h^m \sum_{j=0}^{n} \delta^j + O(h_x^{2m-1})$$

$$\leq C h^m \cdot \sqrt{n} \left(\sum_{j=0}^{n} \left(\delta^j \right)^2 \right)^{1/2} + O(h_x^{2m-1})$$

$$= O(h_x^{2m-1}).$$

Finally noting that the truncation error terms driving the estimate are proportional to

$$c^2 \| \frac{d^{2m+1} u}{dx^{2m+1}} \| + \| \frac{d^{2m} v}{dx^{2m}} \|$$

and by the energy equality these can be bounded by their initial values the estimate (3.11) follows. □

3.3. Two Space Dimensions. The straightforward extension of the method described above to two (or higher) space dimensions would be to use the extension of the one dimensional recursion relation on tensor product interpolants based on derivative data of order m in each variable to approximate u and of order $m-1$ in each variable to approximate v. Unfortunately such a method does not achieve stability at the full geometric CFL condition, $c\Delta t / h_{x,y} < 1$. Fortunately, we have found that a slight modification of the computation of the first time derivatives results in a method which does achieve stability at the full CFL and displays optimal convergence in the energy norm.

The motivation for the modified method is the control of the multidimensional semi-norm in which Hermite interpolation is a projection. However, the modified method described below does not result in exact evolution of polynomial initial data, which is the second main ingredient in our error analysis. Therefore a more involved analysis would be required, and to date we have not been able to carry it out.

The recursion relations for computing the time derivatives are a straightforward generalization of the one dimensional case and read as follows (for brevity we consider the case $f = 0$ and suppress the indices of the space-time grid)

(3.21)
$$c_{k,l,s} = \frac{\Delta t}{s} d_{k,l,s-1},$$

(3.22)
$$d_{k,l,s} = c^2 \frac{(k+2)(k+1)}{s} \frac{\Delta t}{h_x^2} c_{k+2,l,s-1} + c^2 \frac{(l+2)(l+1)}{s} \frac{\Delta t}{h_y^2} c_{k,l+2,s-1}.$$

Now, the local degrees of freedom representing v and u at each node are m^2 and $(m+1)^2$ coefficients of node centered tensor product polynomials of degree $m-1$ and m. In order for the two dimensional method to be stable with a time step $\Delta t = \mathrm{CFL}\min(h_x, h_y)$ where $\mathrm{CFL} \approx 1$ we have found that it is necessary to start up the recursion with the following first step

$$
\begin{array}{ll}
c_{k,l,1} = \Delta t\, d_{k,l,0}, & k, l = 0, \ldots, 2m-1, \\
d_{k,l,1} = c^2(k+2)(k+1)\frac{\Delta t}{h_x^2} c_{k+2,l}^X + c^2(l+2)(l+1)\frac{\Delta t}{h_y^2} c_{k,l+2}^Y, & k, l = 0, \ldots, 2m-1.
\end{array}
$$

Here $c_{k,l}^X$ are the $(2m+2) \times (2m)$ coefficients of $\mathcal{I}_{m,m-1}u$ and $c_{k,l}^Y$ are the $(2m) \times (2m+2)$ coefficients of $\mathcal{I}_{m-1,m}u$. For the remaining coefficients $s = 2 \ldots, 4m+3$ we use (3.21) and (3.22) with $k, l = 0, \ldots, 2m+1$.

To add up the Taylor series at the cell center we must also compute an approximation to u itself. For this we use $\mathcal{I}_{m,m}u$. As in the one dimensional case the evaluation of the approximations (now truncated in k and l) concludes the first half-step.

As we are considering flat boundaries we can apply the one dimensional odd/even extension of ghost polynomials to enforce Dirichlet and Neumann boundary conditions.

The motivation for the use of the special interpolants in the startup step is again the basic inequality for the conserved energy which follows from (2.8):

$$
c^2 \left| \frac{\partial \mathcal{I}_{m,m-1}u}{\partial x} \right|_{m,m}^2 + c^2 \left| \frac{\partial \mathcal{I}_{m-1,m}v}{\partial y} \right|_{m,m}^2 + |\mathcal{I}_{m-1,m-1}v|_{m,m}^2
$$

(3.23)
$$
\leq c^2 \left| \frac{\partial u}{\partial x} \right|_{m,m}^2 + c^2 \left| \frac{\partial u}{\partial y} \right|_{m,m}^2 + |v|_{m,m}^2 .
$$

To use (3.23) to establish stability we must supplement it with a separate treatment of functions which are independent of one of the coordinates. This is a somewhat complex argument which we will not pursue here. In addition we have not been able to generalize the convergence estimate (3.11). However, in all numerical experiments (see §5) we have carried out we do observe convergence at or above the order predicted by (3.11) as well as the ability to march in time at a CFL number one independent of order.

We note that although the method involves four Hermite interpolants, a proper organization of the calculation based on viewing $\mathcal{I}_{m,m-1}u$, $\mathcal{I}_{m-1,m}u$ and $\mathcal{I}_{m,m}u$ as corrections to $\mathcal{I}_{m-1,m-1}u$ shows that for m not small the cost is not much more than the cost of computing two Hermite interpolants.

3.4. Truncation Error. To better understand the error behavior of the method as well as its dispersion/dissipation properties it is useful to examine the truncation error to leading order. To that end we use (2.13) as well as a straightforward generalization to the mixed order interpolants used when $d > 1$:

$$
\frac{\partial^2 \mathcal{I}_{m,m-1}f(x,y)}{\partial x^2} \approx \frac{\partial^2 f(x,y)}{\partial x^2} - \frac{1}{(2m+2)!} \frac{\partial^2}{\partial x^2} \left(x^2 - \frac{h^2}{4} \right)^{m+1} \frac{\partial^{2m+2}f}{\partial x^{2m+2}}(0,0)
$$

(3.24)
$$
- \frac{1}{(2m)!} \left(y^2 - \frac{h^2}{4} \right)^{m} \frac{\partial^{2m+2}f}{\partial x^2 \partial y^{2m}}(0,0),
$$

$$
\frac{\partial^2 \mathcal{I}_{m-1,m}f(x,y)}{\partial y^2} \approx \frac{\partial^2 f(x,y)}{\partial y^2} - \frac{1}{(2m+2)!} \frac{\partial^2}{\partial y^2} \left(y^2 - \frac{h^2}{4} \right)^{m+1} \frac{\partial^{2m+2}f}{\partial y^{2m+2}}(0,0)
$$

(3.25)
$$
- \frac{1}{(2m)!} \left(x^2 - \frac{h^2}{4} \right)^{m} \frac{\partial^{2m+2}f}{\partial x^{2m} \partial y^2}(0,0).
$$

Again we assume the cell center is the origin. Writing $c\Delta t = \lambda h$ the exact update formulas are

$$
u(0,0,\Delta t/2) = \sum_{j=0}^{m+1} \frac{\left(\lambda \frac{h}{2}\right)^{2j}}{(2j)!} \nabla^{2j} u(0,0,0) + c^{-1} \sum_{j=0}^{m} \frac{\left(\lambda \frac{h}{2}\right)^{2j+1}}{(2j+1)!} \nabla^{2j} v(0,0,0) + O(h^{2m+3}),
$$

$$
v(0,0,\Delta t/2) = c \sum_{j=1}^{m+1} \frac{\left(\lambda \frac{h}{2}\right)^{2j-1}}{(2j-1)!} \nabla^{2j} u(0,0,0) + \sum_{j=0}^{m} \frac{\left(\lambda \frac{h}{2}\right)^{2j}}{(2j)!} \nabla^{2j} v(0,0,0) + O(h^{2m+2}).
$$

Substituting (2.13) as well as (3.24)-(3.25) into the update formulas for the discrete solutions $u^{1/2}$, $v^{1/2}$ we derive an expression for the truncation errors to leading order. To simplify these we introduce:

$$L^k u \equiv \frac{\partial^k u}{\partial x^k}(0,0,0) + \frac{\partial^k u}{\partial y^k}(0,0,0), \quad M^k u \equiv \frac{\partial^k u}{\partial x^{k-2} \partial y^2}(0,0,0) + \frac{\partial^k u}{\partial x^2 \partial y^{k-2}}(0,0,0).$$

Then

$$u(0,0,\Delta t/2) - u^{1/2}(0,0) = \frac{1}{(2m+2)!} L^{2m+2} u \sum_{j=0}^{m+1} \frac{\left(\lambda \frac{h}{2}\right)^{2j}}{(2j)!}(2j)! \left(\begin{array}{c} m+1 \\ j \end{array} \right) \left(-\frac{h^2}{4}\right)^{m+1-j}$$

$$+ \frac{1}{(2m)!} M^{2m+2} u \sum_{j=1}^{m+1} \frac{\left(\lambda \frac{h}{2}\right)^{2j}}{(2j)!}(2j-2)! \left(\begin{array}{c} m \\ j-1 \end{array} \right) \left(-\frac{h^2}{4}\right)^{m+1-j}$$

$$+ \frac{c^{-1}}{(2m)!} L^{2m} v \sum_{j=0}^{m} \frac{\left(\lambda \frac{h}{2}\right)^{2j+1}}{(2j+1)!}(2j)! \left(\begin{array}{c} m \\ j \end{array} \right) \left(-\frac{h^2}{4}\right)^{m-j} + O(h^{2m+3}),$$

$$v(0,0,\Delta t/2) - v^{1/2}(0,0) = \frac{1}{(2m+2)!} L^{2m+2} u \sum_{j=1}^{m+1} \frac{\left(\lambda \frac{h}{2}\right)^{2j-1}}{(2j-1)!}(2j)! \left(\begin{array}{c} m+1 \\ j \end{array} \right) \left(-\frac{h^2}{4}\right)^{m+1-j}$$

$$+ \frac{1}{(2m)!} M^{2m+2} u \sum_{j=1}^{m+1} \frac{\left(\lambda \frac{h}{2}\right)^{2j-1}}{(2j-1)!}(2j-2)! \left(\begin{array}{c} m \\ j-1 \end{array} \right) \left(-\frac{h^2}{4}\right)^{m+1-j}$$

$$+ \frac{1}{(2m)!} L^{2m} v \sum_{j=0}^{m} \frac{\left(\lambda \frac{h}{2}\right)^{2j}}{(2j)!}(2j)! \left(\begin{array}{c} m \\ j \end{array} \right) \left(-\frac{h^2}{4}\right)^{m-j} + O(h^{2m+2}).$$

To simplify these expressions introduce some notation and apply the binomial formula to find

(3.26) $u(0,0,\Delta t/2) - u^{1/2}(0,0)$

$$= \lambda \frac{h}{2} \left(\alpha h^{2m+1} L^{2m+2} u + \beta h^{2m+1} M^{2m+2} u + c^{-1} \gamma h^{2m} L^{2m} v + O(h^{2m+2}) \right),$$

(3.27) $v(0,0,\Delta t/2) - v^{1/2}(0,0)$

$$= \lambda \frac{h}{2} \left(c \eta h^{2m} L^{2m+2} u + c \gamma h^{2m} M^{2m+2} u + \nu h^{2m-1} L^{2m} v + O(h^{2m+1}) \right),$$

where

$$\nu = \frac{2^{-(2m-1)}}{(2m)!} \frac{\left(\lambda^2 - 1\right)^m}{\lambda}, \quad \alpha = \frac{\lambda^2 - 1}{4(2m+2)(2m+1)}\nu, \quad \eta = \frac{\lambda}{4m+2}\nu,$$

$$\beta = \frac{2^{-(2m+2)}}{(2m)!} \sum_{j=0}^{m}(j+1)^{-1}(2j+1)^{-1}\lambda^{2j+1}(-1)^{m-j} \left(\begin{array}{c} m \\ j \end{array} \right), \quad \gamma = \frac{2^{-2m}}{(2m)!} \sum_{j=0}^{m}(2j+1)^{-1}\lambda^{2j}(-1)^{m-j} \left(\begin{array}{c} m \\ j \end{array} \right).$$

From (3.26)-(3.27) we make the following predictions which will be verified in the numerical experiments:

i. For $\lambda < 1$ the third term in (3.27) dominates the error and we expect convergence at order $2m-1$, exactly as proved for the energy when $d=1$.

ii. For $\lambda \to 1$ the second term in (3.27) dominates the error and the convergence rate will approach $2m$.

4. Conservative Hermite Methods for the Wave Equation. In contrast with the dissipative method, our conservative method only uses the variable u, directly approximating

(4.1)
$$\frac{\partial^2 u}{\partial t^2} = c^2 \nabla^2 u + f.$$

The starting point for the conservative schemes is the Taylor expansion in time of $u(x,t)$ around $t = t_n + \frac{\Delta t}{2}$ and $t = t_n - \frac{\Delta t}{2}$, i.e.

$$u(x, t \pm \frac{\Delta t}{2}) = \sum_{r=0} \frac{1}{r!} \left(\pm \frac{\Delta t}{2} \right)^r \frac{\partial^r u(x, t_n)}{\partial t^r}.$$

Adding the equations we find

(4.2) $\qquad u(x, t + \frac{\Delta t}{2}) + u(x, t - \frac{\Delta t}{2}) = 2 \sum_{l=0} \frac{1}{2l!} \left(\frac{\Delta t}{2} \right)^{2l} \frac{\partial^{2l} u(x, t_n)}{\partial t^{2l}} = 2 \sum_{l=0} \frac{1}{2l!} \left(\frac{c \Delta t}{2} \right)^{2l} \nabla^{2l} u(x, t_n),$

where we have used equation (4.1) with $f = 0$ to replace time derivatives with derivatives in x. With these formulas we pursue a Hermite-based generalization of high-order conservative time-stepping schemes based on the so-called modified equation approach; see, for example, [13] for an example in the finite difference context.
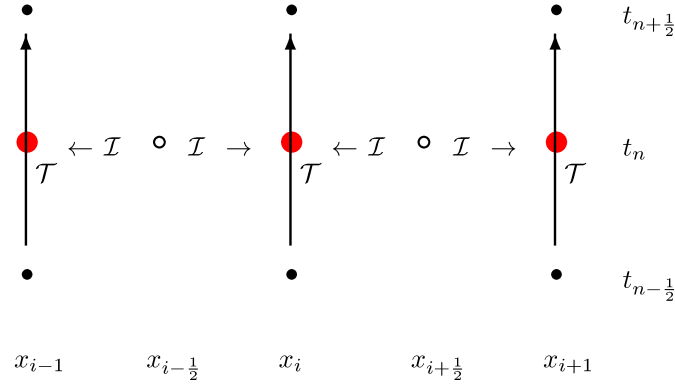

Fig. 4.1: A schematic picture of the steps in the conservative method. Solid black circles represent the primal grid and open circles represent the dual grid. The red circles represent the interpolated solution where the equations are enforced. $\mathcal{I}$ is the Hermite interpolation operator and $\mathcal{T}$ is the time evolution operator.

4.1. The Method in One Space Dimension. Using the same grids and degrees-of-freedom, $c_{k,i}^n$, as in the previous section, as well as the Hermite interpolant at the current time level

$$p_i^n(x) = \sum_{l=0}^{2m+1} c_{l,0,i}^n \left(\frac{x - x_i}{h_x} \right)^l$$

we may employ the update formula (4.2) to find $c_{k,i}^{n+1/2}$

(4.3) $\qquad c_{k,i}^{n+\frac{1}{2}} + c_{k,i}^{n-\frac{1}{2}} = 2 \sum_{l=0}^{m - \lfloor \frac{k}{2} \rfloor} \left(\frac{c \Delta t}{2 h_x} \right)^{2l} c_{2l+k,0,i}^n \frac{(2l+k)!}{(2l)!\, k!}, \quad k = 0, \ldots, m.$

Here the truncation of the sum guarantees that all terms are included and hence the evolution of polynomial data is exact. Thus, under the usual CFL condition $c \Delta t < h_x$, the updated data $c_{k,i}^{n+\frac{1}{2}}$ is also exact for the evolved piecewise polynomial approximations, p_i^n and $p_i^{n-1/2}$. Boundary conditions are enforced in the same way as for the dissipative method. A schematic of the grids used and the process for taking one step can be found in Figure 4.1.

12

4.2. Convergence in One Space Dimension. Assuming periodic boundary conditions we write the method as:

$$(4.4) \qquad p^{n+1/2} = \mathcal{I}\left(\mathcal{S}_+ p^n + \mathcal{S}_- p^n\right) - p^{n-1/2}.$$

The half-step solution operators $\mathcal{S}_\pm$ are defined by:

$$\mathcal{S}_\pm w(x) = w\left(x \pm c\frac{\Delta t}{2}\right),$$

and as above $\mathcal{I}$ denotes both the dual-to-primal and primal-to-dual degree $2m+1$ Hermite interpolation operators with cell widths h (we are suppressing the suffixes as the interpolants are always degree $2m+1$ for the conservative methods).

Concerning the shift operators $\mathcal{S}_\pm$ we note that:

$$(4.5) \qquad |\mathcal{S}_\pm f|_{m+1} = |f|_{m+1},$$
$$(4.6) \qquad \|\mathcal{S}_\pm f\| = \|f\|,$$

The exact solution, u, satisfies:

$$(4.7) \qquad u(x, t + \Delta t/2) = \mathcal{S}_+ u(x,t) + \mathcal{S}_- u(x,t) - u(x, t - \Delta t/2).$$

The error, $e^n(x) = u(x, t_n) - p^n(x)$, then satisfies

$$(4.8) \qquad \begin{aligned} e^{n+1/2} &= \mathcal{I}\left(\mathcal{S}_+ e^n + \mathcal{S}_- e^n\right) \\ &\quad + (1 - \mathcal{I})\left(\mathcal{S}_+ u(\cdot, t_n) + \mathcal{S}_- u(\cdot, t_n)\right) - e^{n-1/2}. \end{aligned}$$

We now introduce conserved variables:

$$U_\pm(x,t) = u(x,t) - \mathcal{S}_\pm u(x, t - \Delta t/2),$$

and note the equation

$$(4.9) \qquad U_\pm(x, t + \Delta t/2) = \mathcal{S}_\mp U_\pm(x,t),$$

holds as a result of (4.7). The equation obviously implies using (4.5),(4.6) that the L^2 norm and Sobolev seminorms of $U_\pm$ are conserved in time. Similarly we define

$$(4.10) \qquad P_\pm^n = p^n - \mathcal{S}_\pm p^{n-1/2},$$

and note that, using the fact that $\mathcal{I}p^n = p^n$, and the definition of the method (4.4),

$$(4.11) \qquad P_\pm^{n+1/2} = \mathcal{I}\mathcal{S}_\mp P_\pm^n + (\mathcal{I} - 1)S_\pm P_\mp^n.$$

Setting

$$(4.12) \qquad E_\pm^n(x) = U_\pm(x, t_n) - P_\pm^n(x),$$

note that

$$(4.13) \qquad E_\pm^n = e^n - \mathcal{S}_\pm e^{n-1/2},$$

and calculate directly using (4.9), (4.11):

$$(4.14) \qquad \begin{aligned} E_\pm^{n+1/2} &= \mathcal{I}\mathcal{S}_\mp E_\pm^n + (\mathcal{I} - 1)\mathcal{S}_\pm E_\mp^n \\ &\quad + (1 - \mathcal{I})\left(\mathcal{S}_+ U_-(\cdot, t_n) + \mathcal{S}_- U_+(\cdot, t_n)\right). \end{aligned}$$

Note that the absence of $\pm$ in the last term due to symmetry.

4.2.1. Energy and error estimates in the Sobolev seminorm. We first note that just as the norms (and seminorms) of $U_\pm$ are conserved by the continuous evolution the sum of the seminorms of $P_\pm^n$ are also conserved. By the orthogonality property of the Hermite interpolation operator (2.7) and the seminorm-preserving property of $\mathcal{S}_\pm$ we have using (4.11)

$$
\begin{aligned}
\left| P_+^{n+1/2} \right|_{m+1}^2 + \left| P_-^{n+1/2} \right|_{m+1}^2 &= \left| \mathcal{I}\mathcal{S}_- P_+^n \right|_{m+1}^2 + \left| (1-\mathcal{I})\mathcal{S}_- P_+^n \right|_{m+1}^2 \\
&\quad + \left| \mathcal{I}\mathcal{S}_+ P_-^n \right|_{m+1}^2 + \left| (1-\mathcal{I})\mathcal{S}_+ P_-^n \right|_{m+1}^2 \\
&= \left| \mathcal{S}_- P_+^n \right|_{m+1}^2 + \left| \mathcal{S}_+ P_-^n \right|_{m+1}^2 \\
&= \left| P_+^n \right|_{m+1}^2 + \left| P_-^n \right|_{m+1}^2 ,
\end{aligned}
$$

(4.15)

with the analogous equality holding at the next half step.

We now consider (4.14). For simplicity we define

$$
\Delta^n = \mathcal{S}_+ U_-(\cdot, t_n) + \mathcal{S}_- U_+(\cdot, t_n),
\tag{4.16}
$$

Note that

$$
\begin{aligned}
U_\pm(x,t) &= u(x,t) - u(x \pm c\frac{\Delta t}{2}, t - \frac{\Delta t}{2}) \\
&= \frac{\Delta t}{2}\left(\frac{\partial u}{\partial t}(x,t) \pm c\frac{\partial u}{\partial x}(x,t) \right) + O(\Delta t^2).
\end{aligned}
\tag{4.17}
$$

Thus for sufficiently smooth solutions u we have

$$
|(1-\mathcal{I})\Delta^n|_{m+1} \leq C\Delta t \cdot h^{m+1}.
\tag{4.18}
$$

We then have:

$$
\begin{aligned}
\left| E_+^{n+1/2} \right|_{m+1}^2 + \left| E_-^{n+1/2} \right|_{m+1}^2 &= \left| \mathcal{I}\mathcal{S}_- E_+^n \right|_{m+1}^2 + \left| (1-\mathcal{I})\mathcal{S}_- E_+^n \right|_{m+1}^2 \\
&\quad + \left| \mathcal{I}\mathcal{S}_+ E_-^n \right|_{m+1}^2 + \left| (1-\mathcal{I})\mathcal{S}_+ E_-^n \right|_{m+1}^2 \\
&\quad - 2\langle (1-\mathcal{I})\left(\mathcal{S}_+ E_-^n + \mathcal{S}_- E_+^n \right), (1-\mathcal{I})\Delta^n \rangle_{m+1} \\
&\quad + 2\left| (1-\mathcal{I})\Delta^n \right|_{m+1}^2 \\
&\leq \left| E_+^n \right|_{m+1}^2 + \left| E_-^n \right|_{m+1}^2 \\
&\quad + C\left(\left| E_+^n \right|_{m+1}^2 + \left| E_-^n \right|_{m+1}^2 \right)^{1/2} \Delta t \cdot h^{m+1} + C\Delta t^2 \cdot h^{2m+2},
\end{aligned}
\tag{4.19}
$$

where we have made use of the orthogonality relation (2.7) and the Cauchy-Schwarz inequality.

Defining

$$
\mathcal{E}_{m+1}^n = \left| E_+^n \right|_{m+1}^2 + \left| E_-^n \right|_{m+1}^2 ,
\tag{4.20}
$$

and summing over all steps we have

$$
\mathcal{E}_{m+1}^n \leq \mathcal{E}_{m+1}^{1/2} + C\Delta t \cdot h^{m+1}\left[\sqrt{\mathcal{E}_{m+1}^{1/2}} + \sum_{j=1}^{n-1}\left(\sqrt{\mathcal{E}_{m+1}^j} + \sqrt{\mathcal{E}_{m+1}^{j+1/2}} \right) \right] + Ct_n\Delta t \cdot h^{2m+2}.
\tag{4.21}
$$

Note that $\mathcal{E}_{m+1}^{1/2}$ is directly determined by the initial data and the first half time step, which we assume has been determined with sufficient accuracy that

$$
\mathcal{E}_{m+1}^{1/2} \leq Ch^{m+1}.
\tag{4.22}
$$

Setting

$$(4.23) \qquad \mathcal{M}_{m+1}^n = \max_{j \leq n} \mathcal{E}_{m+1}^j,$$

we have

$$(4.24) \qquad \mathcal{M}_{m+1}^n \leq Ct_n h^{m+1} \sqrt{\mathcal{M}_{m+1}^n} + Ch^{2m+2}(1 + t_n \Delta t).$$

From this we conclude the fundamental error estimate in the seminorm.

LEMMA 4.1. *For u sufficiently smooth there exists C independent of h and Δt such that*

$$(4.25) \qquad \mathcal{E}_{m+1}^n \leq Ch^{2m+2}(1 + t_n^2).$$

4.2.2. Error estimates in L^2. We begin by estimating the L^2 norms of the errors for the conserved variables, $E_{\pm}^n$. Using (4.14) in conjunction with (2.9), (2.12), (4.6) and (4.25) we obtain the inequality

$$\|E_{\pm}^{n+1/2}\| \leq \|S_{\mp} E_{\pm}^n\| + \|(\mathcal{I} - 1)S_{\mp} E_{\pm}^n\| + \|(\mathcal{I} - 1)S_{\pm} E_{\mp}^n\| + \|(\mathcal{I} - 1)\Delta^n\|$$
$$\leq \|E_{\pm}^n\| + Ch^{m+1}\sqrt{\mathcal{E}_{m+1}^n} + C\Delta t \cdot h^{2m+2}$$
$$(4.26) \qquad \leq \|E_{\pm}^n\| + C(1 + t_n)h^{2m+2}.$$

Assuming the initial data is sufficiently accurate

$$(4.27) \qquad \|e^{0,1/2}\| \leq Ch^{2m+1},$$

and that the CFL number is bounded below

$$(4.28) \qquad c\frac{\Delta t}{h} \geq \eta > 0,$$

summing (4.26) yields:

$$(4.29) \qquad \|E_{\pm}^n\| \leq C(1 + t_n^2)h^{2m+1}.$$

Now from (4.13) and (4.29) we obtain

$$(4.30) \qquad \|e^{n+1/2}\| \leq \|S_{\pm} e^n\| + \|E_{\pm}^{n+1/2}\| \leq \|e^n\| + C(1 + t_n^2)h^{2m+1}.$$

Summing (4.30) we obtain our main convergence result.

THEOREM 2. *For u sufficiently smooth there exists C independent of h and $\Delta t \geq c^{-1}\eta h$ such that*

$$(4.31) \qquad \|e^n\| \leq Ch^{2m}(1 + t_n^3).$$

4.3. The Method in Two Space Dimensions. Again using (4.2) and, as for the dissipative method, using the notation in (2.5), (2.6), we derive an explicit and compact update formula for the coefficients (again the spatial indices are suppressed)

$$(4.32) \qquad c_{k,l}^{n+\frac{1}{2}} + c_{k,l}^{n-\frac{1}{2}} = \frac{2}{k! \, l!} \sum_{r=0}^{2m} \sum_{\{i_x,i_y\} \in \Sigma} (k + 2i_x)! \, (l + 2i_y)! \mathcal{P}_{i_x,i_y} c_{k+2i_x,l+2i_y,0}^n, \quad k, l = 0, \ldots, m.$$

Here Σ is the set of integers that satisfy all of the relations

$$i_x + i_y = r, \quad (k + 2i_x) \leq (2m + 1), \quad (l + 2i_y) \leq (2m + 1),$$

or said more plainly, we make sure to not address outside the array holding $c_{(\cdot,\cdot,0)}^n$. The matrix $\mathcal{P}$ is a scaled version of Pascal's triangle (oriented to start in element $\mathcal{P}_{0,0}$). Precisely, let

15

$$(4.33) \qquad P = \begin{pmatrix} 1 & 1 & 1 & 1 & \dots \\ 1 & 2 & 3 & 4 & \dots \\ 1 & 3 & 6 & 10 & \dots \\ 1 & 4 & 10 & \ddots & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix},$$

then

$$\mathcal{P}_{i,j} = \frac{1}{(2i+2j)!} P_{i,j} \left(\frac{c\Delta t}{h_x} \right)^{2i} \left(\frac{c\Delta t}{h_y} \right)^{2j},$$

where the indexation for i and j starts with zero. Obviously, extensions to more space dimensions are straightforward.

4.4. Truncation Error. The truncation error for the conservative method is computed as in the dissipative case. In particular, noting that truncating the sum in (4.2) at $l = m + 1$ produces an error of $O(h^{2m+4})$, setting $c\Delta t = \lambda h$, and invoking (2.13) we find:

$$u(0,0,\Delta t) - p^1(0,0) = \left(\frac{c\Delta t}{2} \right)^2 \left(h^{2m} \frac{2^{-(2m+1)}}{(2m+2)!} \lambda^{-2} L^{2m+2} u \sum_{l=0}^{m+1} \lambda^{2l} (-1)^{m+1-l} \binom{m+1}{l} + O(h^{2m+2}) \right)$$

$$(4.34) \qquad = \left(\frac{c\Delta t}{2} \right)^2 \left(h^{2m} \frac{2^{-(2m+1)}}{(2m+2)!} \frac{(\lambda^2 - 1)^{m+1}}{\lambda^2} L^{2m+2} u + O(h^{2m+2}) \right).$$

From (4.34) we predict, as proven for $d = 1$, that the convergence rate for the conservative method will be $2m$, approaching $2m + 2$ as $\lambda \to 1$. This will be verified in the numerical experiments.

5. Numerical Experiments.

5.1. A Gaussian Pulse in a 1D Box. The aim of this example is to empirically investigate the rates of convergence of the conservative and dissipative methods as a function of m. To do this we solve

$$u_{tt} = u_{xx},$$

on $x \in [-3/2, 3/2]$ with the initial data

$$u(x,0) = e^{-20x^2}, \quad v(x,0) = 0,$$

and with boundary conditions $u(-3/2, t) = 0$, $u(3/2, t) = 0$. The boundary conditions are enforced as described above, that is, if we denote the $m + 1$ coefficients of the polynomial approximating u at the leftmost dual gridpoint $x_{1/2} = -3/2 + h_x/2$ by $c_l[x_{1/2}]$, we set $c_l[x_{-1/2}] = (-1)^l c_l[x_{1/2}]$. Similarly on the right boundary we set the "ghost-polynomial" $c_l[x_{n_x+1/2}] = (-1)^{(l+1)} c_l[x_{n_x-1/2}]$. The boundary conditions for the velocity are handled in the same way.

We solve until time $12 + \tau$, $\tau \approx .25$, when the exact solution is

$$u(x, 12 + \tau) = \frac{1}{2} \left(e^{-20(x+\tau)^2} + e^{-20(x-\tau)^2} \right),$$

and compute the L_2-error for the solution evaluated on a sufficiently fine grid. For all m we choose the time step so that $\Delta t / h_x = \lambda$. We use a sequence of grid refinements starting with $n_x^{(0)} = 10$ and $n_x^{(i)} = \lceil 1.2 n_x^{(i-1)} \rceil$ and fix $\lambda = 0.8$ and 1.0. In order to be able to take an integer number of time steps we adjust τ to be as close to .25 as possible.

The L^2-errors for resulting h_x and for the dissipative method are displayed in Figure 5.1 and the results for the conservative method can be found in Figure 5.2. In both figures we display results for $m = 0, \dots, 5$. As can be seen the rates of convergence are $(2m-1)$ and $(2m)$ for the dissipative method for $\lambda = 0.8$ and 1.0 and $(2m)$ and $(2m+2)$ for the conservative method. These are as predicted by the truncation error analysis given above.
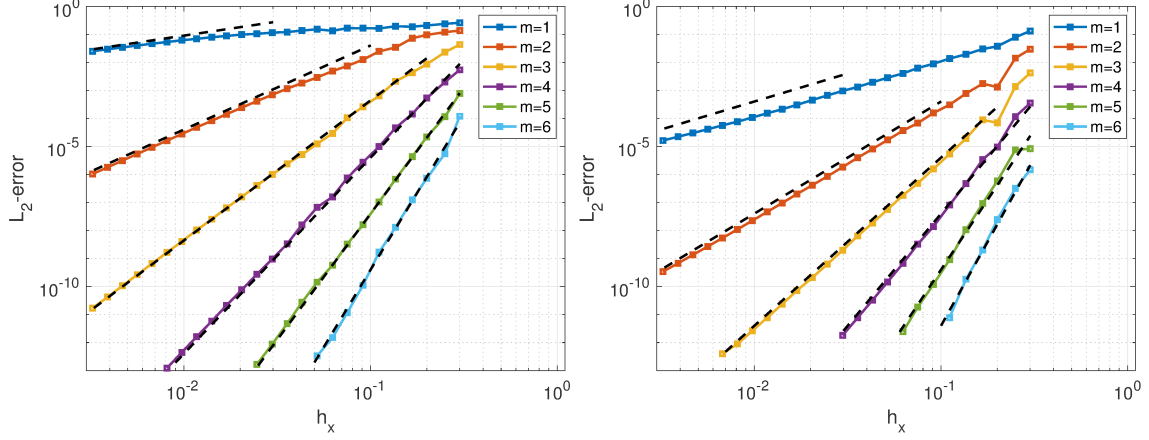
16

Fig. 5.1: The dissipative method, results are for the example with the Gaussian initial data. Displayed are L^2-errors as a function of h_x for $m = 1, \ldots, 6$. To the left is for $\lambda = 0.8$ and to the right $\lambda = 1.0$. The dashed lines are $\sim h_x^{2m-1}$ to the left and $\sim h_x^{2m}$ to the right.
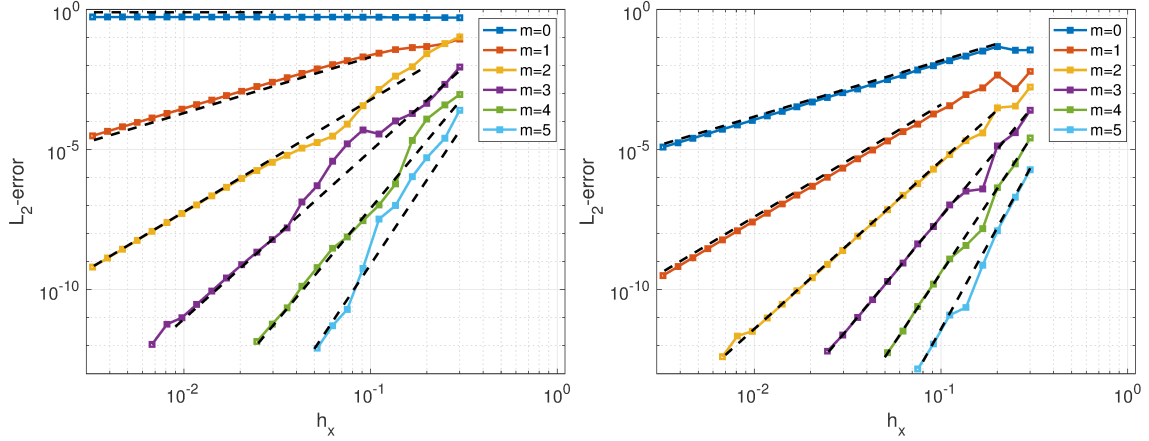

Fig. 5.2: The conservative method, results are for the example with the Gaussian initial data. Displayed are L_2-errors as a function of h_x for $m = 0, \ldots, 5$. To the left is for $\lambda = 0.8$ and to the right $\lambda = 1.0$. The dashed lines are $\sim h_x^{2m}$ to the left and $\sim h_x^{2m+2}$ to the right.

5.2. Numerical Verification of Conservation. The theory for the conservative method above does not account for roundoff effects and in this experiment we investigate how well the energy

$$\mathcal{E}(t_n) \equiv \big| P_+^n \big|_{m+1}^2 + \big| P_-^n \big|_{m+1}^2,$$

is conserved between time steps. In the Figure 5.3 we display the $(m+1)$th derivative of P_+ and P_- for the approximation of the exact solution $u(x,t) = \sin(x)\cos(t)$ at time $t = 0$ and on a grid $x \in [-\pi, \pi]$ with $h_x = \frac{2\pi}{3}$ and with $\Delta t = h_x/2$.

As can be seen in the figure the $(m+1)$th derivatives of $P_\pm$ are discontinuous and displaced relative to the element centers. As the curves are piecewise smooth we (for simplicity) fix $\Delta t = h_x/2$ which allows the sub-division of each element into four parts of size $h_x/4$ where the curves are continuous polynomials that can be integrated exactly (up to machine precision effects).

In Figure 5.4 we display the difference in $\mathcal{E}$ at the initial time and subsequent time steps (we display every 100th time step). Two initial data are considered, the exact solution $u(x,t) = \sin(x)\cos(t)$, and random
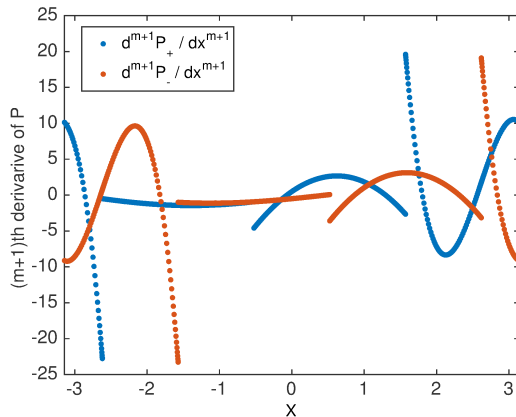
Fig. 5.3: The figure displays the $(m + 1)$th derivative of P_+ and P_- for the approximation of the exact solution $u(x, t) = \sin(x)\cos(t)$ at time $t = 0$.

initial data. For the latter the initial polynomial coefficients are chosen randomly from a uniform distribution between 0 and 1. We use $h_x = 2\pi/30$ and monitor $\mathcal{E}$ for one million time steps for $m = 1, 3, 5$. In the first case the change in $\mathcal{E}$ is relatively small although the size of the difference is increasing with increasing m. For the second set of initial data the difference of $\mathcal{E}$ from its initial value increases more rapidly in t.

5.3. An Example in Two Space Dimensions. To verify the convergence properties of the two methods in two space dimensions we approximate the plane wave $u = \sin(2\pi\kappa(x + y + \sqrt{2}t)$ on the unit square with periodic boundary conditions. As in the one dimensional case we fix $\lambda = 0.8$ and 1.0 and take the final time to be as close to 4.18 as possible using an integer number of time steps. The parameter κ is taken to be $m + 1$.

In Figures 5.5 and 5.6 we display the L_2-errors in u for the conservative and dissipative methods. As in the one dimensional case we see that the methods are of order $2m$ and $2m - 1$ when $\lambda < 1$ and $2m + 2$ $2m$ respectively when $\lambda = 1.0$.

6. Implementations on Accelerators. The Hermite methods of Goodrich and co-authors were first tailored to graphics processing units by Dye in [10] wherein an implementation for the two dimensional advection equation was proposed. Strategies for three-dimensional equations were later proposed by Vargas and co-authors in [21]. Here we focus on extending the techniques introduced in [21] to develop algorithms for the dissipative and conservative Hermite methods in three space dimensions.

In [21] two strategies for implementing Hermite methods on GPUs were explored. The first approach proposed separate interpolation and evolution kernels enabling specialized tuning of each step. A drawback of this approach is the additional memory required to explicitly store the interpolant. To bypass the need of additional storage, a monolithic kernel was proposed which fused the interpolation and evolution procedures
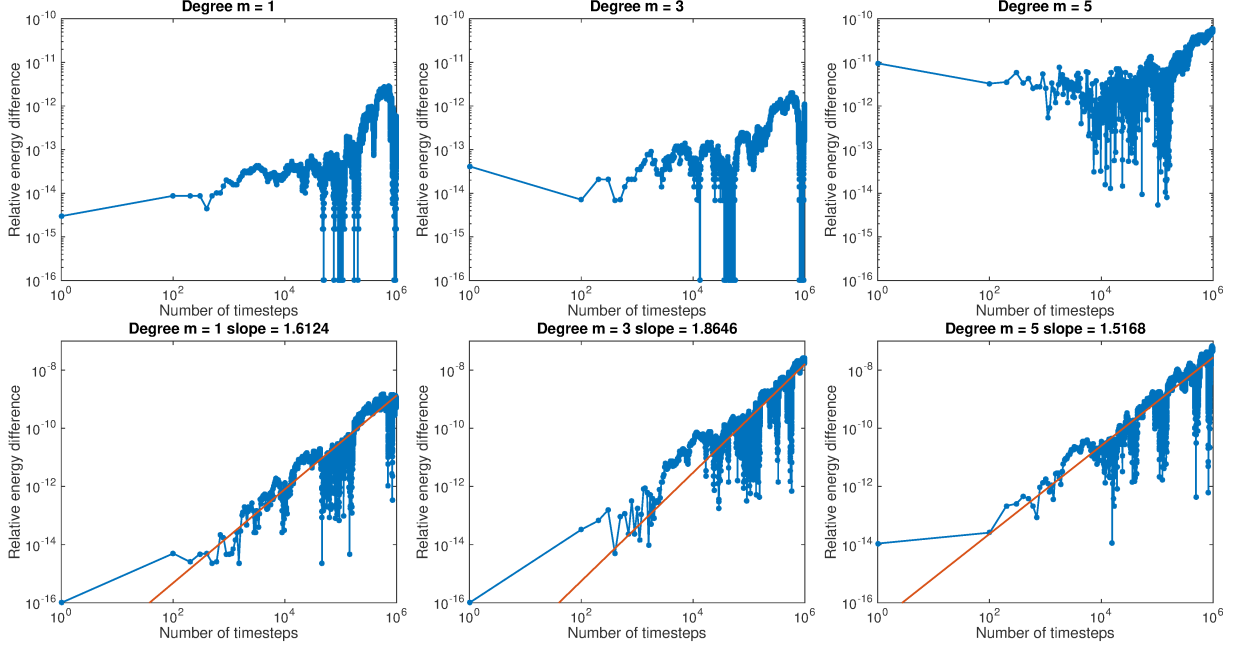
Fig. 5.4: Very long time conservation properties for smooth and rough data under the influence of roundoff errors.
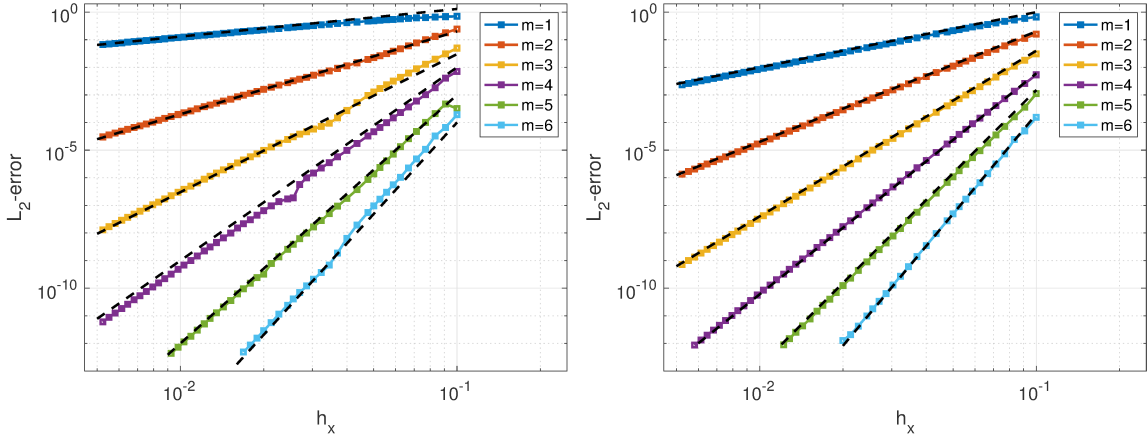

Fig. 5.5: Displayed are the L_2-errors as a function of grid size for $m = 1, \ldots, 6$ for $\lambda = 0.8$ to the left and $\lambda = 1.0$ to the right. These results are for the dissipative method and the dashed slopes are $\sim h_x^{2m-1}$ and $\sim h_x^{2m}$ for $\lambda = 0.8$ and $\lambda = 1.0$.

into a single kernel call. Performance results in [21] demonstrated that both approaches led to a comparable time to solution.

Keeping consistent with the CUDA nomenclature [19], we expose two levels of parallelism. The first layer assigns a block of threads to carryout the interpolation and evolution procedure on a cell. The second layer exposes fine grained parallelism in which block local threads carry out the local operations found in constructing the interpolant and evolution step. Exploiting the tensor product structure of the polynomials, the interpolants are reconstructed in a dimension by dimension manner and expressed as a series of matrix-matrix multiplications. To maximize performance, shared memory is used to hold intermediate computations
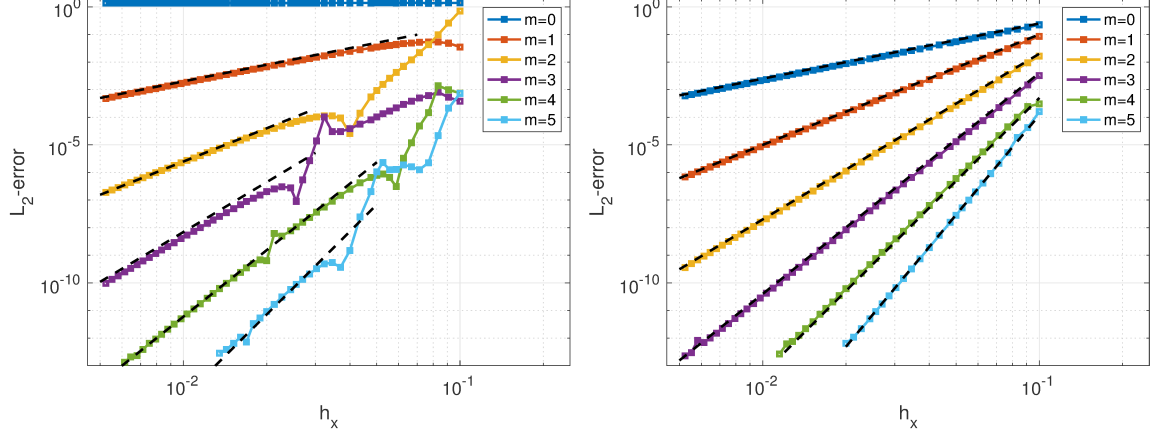
Fig. 5.6: Displayed are the L_2-errors as a function of grid size for $m = 0, \ldots, 5$ for $\lambda = 0.8$ to the left and $\lambda = 1.0$ to the right. These results are for the conservative method and the dashed slopes are $\sim h_x^{2m}$ and $\sim h_x^{2m+2}$ for $\lambda = 0.8$ and $\lambda = 1.0$.

(shared memory is replaced with memory on the stack when considering the CPU). Additionally, we apply the shared memory rolling technique as used in [21, 20].

The experiments presented in this manuscript were carried out on a single node of SMU's ManeFrame II compute cluster using double precision and an NVIDIA Tesla P100 graphics card as the accelerator. The card has a theoretical peak bandwidth of 732 GB/s and can potentially perform 4700 Gigaflops in double precision. To estimate a more "realistic" bandwidth we consider a simple DAXPY (combination of scalar multiplication and vector addition) kernel. For a sufficiently large vector a streaming bandwidth of 552 GB/s was observed on the graphics card; although this is not indicative of the card's capabilities it does serve as a representative of " achievable peak" performance numbers.

In this work we evaluate kernel performance by measuring the effective arithmetic throughput and effective memory bandwidth using the NVIDIA profiler. The effective arithmetic throughput is computed using the number of floating-point operations given by flop_count_dp, and the effective memory bandwidth is obtained by summing *dram_read_throughput* and *dram_write_throughput*. Here bandwidth corresponds to the sum of bytes read and written to global memory by a GPU kernel. We adopt OCCA as our API [17] which allows us to cross-compile code into a variety of API's. For completeness we provide a comparison of time to solution for a GPU and a multi-core CPU. The multi-core CPU is a dual 18-core Intel Xeon CPU E5-2695 with an operating frequency of 2.10 GHz.

We note that these experiments are for problems posed in three space dimensions, as it seems most important to develop efficient implementations of the method with an eye towards challenging large-scale problems.

6.1. Dissipative Method. Following the techniques described above, we begin with tailoring the dissipative Hermite method to the graphics processing unit. The main challenge in tailoring the method stem from the varying number of degrees of freedom used in representing the displacement and velocity variables. This becomes troubling as building interpolants of varying spatial order leads to irregular access patterns. To overcome this challenge local arrays are padded with zeros to ensure data structures of equal dimensions thus simplifying the implementation. The first version of the method, which will be referred to as the **simple-method**, introduced in this article reconstructs a single Hermite interpolant for the displacement and velocity components respectively; while the second variant, which will be referred to as **full-method** introduces an additional interpolant for each spatial dimension (total of five interpolants), the additional interpolants were necessary in order to maintain $\Delta t \approx \min(h_x, h_y, h_z)$. In this section we explore both version and we compare time per iteration.

We begin our performance analysis by considering the two kernel approach for the simple-method and

follow by considering monolithic kernels for both the simple and full methods. As the full method would require substantial global memory for the five interpolants we did not consider the two kernel approach. Figure 6.1 reports the observed bandwidth and GFLOPs for both versions of the method. For these numerical experiments, we chose the number of stages of the Hermite-Taylor Scheme to be $2m + 2$, which is less than what is required for exact evolution of the cell polynomials, but sufficient for convergence at design order. As intermediate computations are stored in shared memory we find that the second variant (with additional interpolants) required more shared memory than supported in the graphics card and thus we cannot go beyond an $m = 2$ method. Here we may conclude that by considering two kernels we can achieve a significantly higher dram throughput but not a significantly better time to solution.
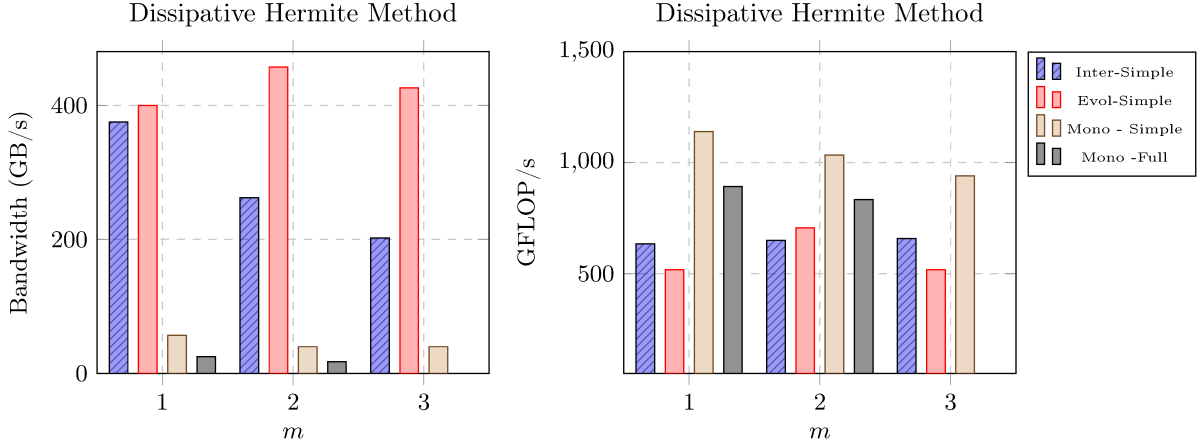

Fig. 6.1: Performance of the Dissipative Hermite Method.

Choice of m	$m = 1$	$m = 2$	$m = 3$
Simple - Interpolation	375 / 634	262 / 650	202 / 658
Simple - Evolution	400 / 517	457 / 706	426 / 518
Simple - Monolithic	57 / 1140	40 / 1034	40 / 940
Full - Monolithic	25 / 893	17.7 / 834	-

Table 6.1: Table of Bandwidth (GB/s) and GFLOP/s.

6.1.1. Time per Iteration. For completeness, a comparison of time per iteration is presented. Here we tailor separate OCCA kernels for the GPU and CPU. Table 6.2 reports the time per iteration when targeting different platforms via the OCCA API. Obviously the GPU delivers the solution in substantially shorter time.

6.2. Conservative Hermite Method. Building on the point-wise formula given in Equation (4.3) we consider an algorithm which first approximates high order derivatives via Hermite interpolation, and propagates the solution by accumulating the necessary spatial derivatives. Figure 6.2 reports the peak achieved bandwidth and GFLOPs for both the separate interpolation/evolution and monolithic kernels. Similar to previous experiments, we find that the performance of separate interpolation and evolution kernels is limited by the device's ability to read and write to global memory. The monolithic kernel does not experience this bottleneck since it does not explicitly write out the interpolant.

6.2.1. Time per Iteration. For completeness, a comparison of time per iteration is presented. As before we tailor separate OCCA kernels for the GPU and CPU. Table 6.4 reports the time per iteration

Choice of m	$m = 1$	$m = 2$	$m = 3$
OCCA::OpenMP - Simple Method	0.88 sec	1.16 sec	1.92 sec
OCCA::OpenMP - Full Method	1.50 sec	2.10 sec	2.57 sec
OCCA::CUDA - Single -Simple method	0.057 sec	0.087 sec	0.10 sec
OCCA::CUDA - Two Kernels - Simple method	0.15 sec	0.15 sec	0.156 sec
OCCA::CUDA - Full method	0.11 sec	0.17 sec	-

Table 6.2: Comparison of time per iteration of the dissipative Hermite kernels executed on the GPU and CPU. For orders $m = 1, 2, 3$ the number of grid points were chosen to be 160, 120, 90 per Cartesian direction.
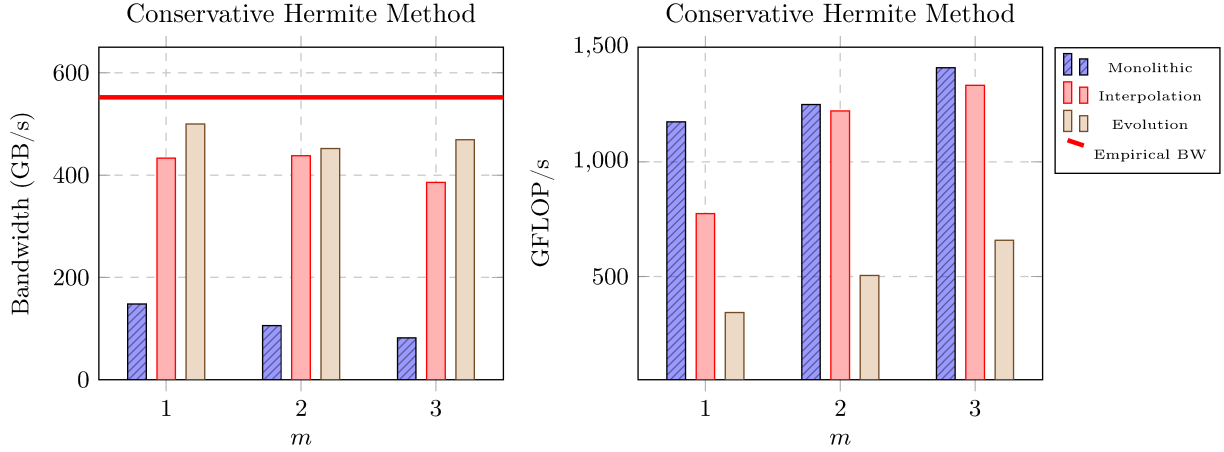

Fig. 6.2: Performance for the conservative Hermite method.

Choice of m	$m = 1$	$m = 2$	$m = 3$
Monolithic Kernel	148/1175	106/1250	82/1410
Interpolation Kernel	433/775	438/1222	386/1334
Evolution Kernel	500/343	452/504	469/658

Table 6.3: Table of Bandwidth (GB/s) and GFLOP/s.

when targeting different platforms via the OCCA API. Again the time to solution is significantly smaller for the GPU.

Choice of m	$m = 1$	$m = 2$	$m = 3$
OCCA::OpenMP	0.55 sec	0.86 sec	1.12 sec
OCCA::CUDA - Single Kernel	0.035 sec	0.052 sec	0.0579 sec
OCCA::CUDA - Two Kernels	0.057sec	0.0628 sec	0.0619 sec

Table 6.4: Comparison of time per iteration of Hermite kernels executed on the GPU and CPU. For orders $m = 1, 2, 3$ the number of grid points were chosen to be 280, 190, 140 points per Cartesian direction. Noticeably a single GPU kernel offers a slightly better time to solution due to the advantage of not having to store the interpolant.

7. Conclusion and Future Work. We have demonstrated two approaches to solving the scalar wave equation using Hermite interpolation. Both are stable for CFL up to 1 independent of order and converge at a rate roughly twice the number of degrees-of-freedom per node. Moreover, we find that the methods do an excellent job of exploiting GPUs, with time-to-solution on a single NVIDIA P100 twenty to thirty times less than on a dual 18-core CPU.

From a practical perspective, future work must include implementations in complex geometry; we plan to use both overset structured grids and hybrid structured-unstructured grids. In addition we will generalize the method to systems, such as the elastic wave equation. Theoretically, although one can prove that the methods are stable at full CFL, we have as yet been unable to extend the convergence analyses to multiple space dimensions.

<div align="center">REFERENCES</div>

[1] D. Appelö and T. Hagstrom. A new discontinuous Galerkin formulation for wave equations in second order form. *SIAM Journal On Numerical Analysis*, 53(6):2705–2726, 2015.
[2] J. Banks and T. Hagstrom. On Galerkin difference methods. *J. Comput. Phys.*, 313:310–327, 2016.
[3] J.W. Banks and W.D. Henshaw . High-order upwind methods for wave equations on curvilinear and overlapping grids. In Robert M. Kirby, Martin Berzins, and Jan S. Hesthaven, editors, *Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2014: Selected papers from the ICOSAHOM conference, June 23-27, 2014, Salt Lake City, Utah, USA*, pages 137–145, 2015.
[4] J.W. Banks and W.D. Henshaw. Upwind schemes for the wave equation in second-order form. *Journal of Computational Physics*, 231(17):5854–5889, 2012.
[5] G. Birkhoff, M. H. Schultz, and R. S. Varga. Piecewise Hermite interpolation in one and two variables with applications to partial differential equations. *Numerische Mathematik*, 11(3):232–256, 1968.
[6] O. Bruno and M. Lyon. High-order unconditionally-stable FC-AD solvers for general smooth domains I. basic elements. *Submitted*, 2009.
[7] X. Chen, D. Appelö, and T. Hagstrom. A hybrid Hermite–discontinuous Galerkin method for hyperbolic systems with application to Maxwell's equations. *Journal of Computational Physics*, 257, Part A:501 – 520, 2014.
[8] X. Chen and T. Hagstrom. P-adaptive Hermite methods for initial value problems. *ESAIM: Mathematical Modelling and Numerical Analysis*, 46:545–557, 2012.
[9] C.-S. Chou, C-W. Shu, and Y. Xing. Optimal energy conserving local discontinuous Galerkin methods for second-order wave equation in heterogeneous media. *Journal of Computational Physics*, 272:88 – 107, 2014.
[10] E. Dye. Performance analysis and optimization of Hermite methods on NVIDIA GPUs using CUDA. Master's thesis, University of New Mexico. Dept. of Mathematics and Statistics, 2014.
[11] J. Goodrich, T. Hagstrom, and J. Lorenz. Hermite methods for hyperbolic initial-boundary value problems. *Math. Comp.*, 75:595–630, 2006.
[12] M. J. Grote, A. Schneebeli, and D. Schötzau. Discontinuous Galerkin finite element method for the wave equation. *SIAM Journal on Numerical Analysis*, 44(6):2408–2431, 2006.
[13] W. D. Henshaw. A high-order accurate parallel solver for Maxwell's equations on overlapping grids. *SIAM Journal on Scientific Computing*, 28(5):1730–1765, 2006.
[14] P. Joly. *Variational methods for time-dependent wave propagation problems.*, volume 31 of *Topics in computational wave propagation*, pages 201–264. Springer, Berlin, 2003.
[15] M. Lyon and O. Bruno. High-order unconditionally-stable FC-AD solvers for general smooth domains II. elliptic, parabolic and hyperbolic pdes: Theoretical considerations. *Submitted*, 2009.
[16] K. Mattsson and J. Nordström. Summation by parts operators for finite difference approximations of second derivatives. *J. Comput. Phys.*, 199:503–540, 2004.
[17] D. Medina. *OKL: A Unified Language for Parallel Architectures*. PhD thesis, Rice University, 2015.
[18] B. Riviere and M.F. Wheeler. Discontinuous finite element methods for acoustic and elastic wave problems. part i: semidiscrete error estimates. *Contemporary Mathematics*, 329:271–282, 2003.
[19] J. Sanders and E. Kandrot. *CUDA by Example: An Introduction to General-Purpose GPU Programming*. Addison-Wesley Professional, 1st edition, 2010.
[20] A. Vargas. *Hermite Methods for the Simulation of Wave Propagation*. PhD thesis, Rice University, 2017.
[21] A. Vargas, J. Chan, T. Hagstrom, and T. Warburton. GPU acceleration of Hermite methods for the simulation of wave propagation. *arXiv preprint arXiv:1609.09841*, 2016.