2014

Discontinuous Galerkin Spectral Element Approximations for the Reflection and Transmission of Waves from Moving Material Interfaces

Andrew R. Winters
DISCONTINUOUS GALERKIN SPECTRAL ELEMENT APPROXIMATIONS FOR THE
REFLECTION AND TRANSMISSION OF WAVES FROM MOVING MATERIAL
INTERFACES

By
ANDREW R. WINTERS

A Dissertation submitted to the
Department of Mathematics
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Degree Awarded:
Spring Semester, 2014
Andrew R. Winters defended this dissertation on February 20, 2014.
The members of the supervisory committee were:

David Kopriva
Professor Directing Dissertation

Jorge Piekarewicz
University Representative

M. Yousuff Hussaini
Committee Member

Kyle Gallivan
Committee Member

Nick Cogan
Committee Member

Bettye Anne Case
Committee Member

The Graduate School has verified and approved the above-named committee members, and certifies that the dissertation has been approved in accordance with university requirements.
To my family. Now every gathering can be like that scene from “Spies Like Us.”
ACKNOWLEDGMENTS

I am extremely grateful to Dr. David Kopriva, my advisor. Much of this work would have not been possible without his ideas, encouragement, thorough review, and dedication. I also thank Dr. M. Yousuff Hussaini, Dr. Kyle Gallivan, Dr. Nick Cogan, Dr. Jorge Piekarewicz and Dr. Bettye Anne Case for their comments and support. Last, but not least, I owe my deepest gratitude to my family, for their love, support and patience.
# TABLE OF CONTENTS

List of Tables .................................................. vii
List of Figures .................................................. viii
Abstract .......................................................... xii

1 **Introduction** ............................................. 1
   1.1 Goals of the Dissertation ............................... 3
   1.2 Contributions of the Dissertation ....................... 4
   1.3 Organization of the Dissertation ......................... 4

2 **The Physics: Modeling Wave Reflection and Transmission from a Moving Interface** .............................. 6
   2.1 Physical Tools for Moving Material Problems ........... 6
   2.2 Governing Equations for Wave Propagation Problems .... 10
      2.2.1 Maxwell’s Equations ................................. 11
      2.2.2 Classical Wave Equation .............................. 15

3 **The Physics I: Analytical Solutions** ........................ 18
   3.1 Plane Wave Solutions ................................... 18
   3.2 Phase Angles for a Vertical Moving Interface .......... 19
      3.2.1 Oblique Phase Angles ................................. 21
   3.3 Reflection and Transmission Coefficients for a Moving Plane Interface .......................... 23
      3.3.1 Reflection and Transmission Coefficients for a Transverse Electric Plane Wave ................. 23
      3.3.2 Reflection and Transmission Coefficients for a Transverse Magnetic Plane Wave ................. 24
      3.3.3 Reflection and Transmission Coefficients for the Classical Wave Equation ................... 25
   3.4 Recovering Electric and Magnetic Fields in a Moving Dielectric .................................... 27
   3.5 Special Cases of Electromagnetic Wave Reflection and Transmission from a Static Dielectric Interface ........................................... 28
      3.5.1 Brewster’s Angle ....................................... 29
      3.5.2 Total Internal Reflection ............................... 30
   3.6 Physical Properties of Acoustic Wave Scattering from a Moving Interface ......................... 32
      3.6.1 Limiting Cases ......................................... 32
      3.6.2 Amplitudes as Functions of Object Speed $\beta$ .................................................... 33
      3.6.3 Spatial Dispersion ..................................... 33

4 **Spatial Approximation: The Discontinuous Galerkin Spectral Element Method on Moving Meshes** ............ 35
   4.1 Discretization in Space of an ALE Conservation Law .......... 35
   4.2 Computation of the Mesh Velocity .......................... 41
   4.3 Three Dimensional Discontinuous Galerkin Method on a Moving Hexahedral Element ............... 42
# The Physics II: Riemann Solvers for Wave Propagation in Moving Media

5.1 Exact, Upwind Numerical Fluxes
5.1.1 Maxwell’s Equations
5.1.2 The Classical Wave Equation
5.2 Consistency Checks for the Numerical Fluxes
5.2.1 Maxwell’s Equations
5.2.2 The Classical Wave Equation

# Temporal Approximation: High-Order Explicit Local Time Stepping

6.1 Derivation of a Local Time Stepping Procedure for the DGSEM
6.2 Variable Coefficient Adams-Bashforth Time Integrator
6.3 Outline of the LTS Algorithm
6.4 Storage and Computational Cost of the LTS Method
6.4.1 Storage Requirements of the LTS Method Compared to LSRK3
6.4.2 Reducing Mesh Induced Stiffness: Theoretical Predictions
6.4.3 For What Kinds of Meshes is LTS Appropriate?

# Efficient Implementation: Parallelization of DGSEM and LTS

7.1 Background for Parallel DGSEM
7.2 Parallel DGSEM with a Standard Time Integrator
7.3 Local Time Stepping in Parallel

# Numerical Results

8.1 Reflection and Transmission of a Plane Wave from a Moving Material Interface
8.1.1 Transverse Electric Waves
8.1.2 Transverse Magnetic Waves
8.1.3 Acoustic Waves
8.2 Local Time Stepping on Static and Moving Quadrilateral and Hexahedral Meshes
8.2.1 Convergence Study of the ALE-DGSEM with AB3LTS
8.2.2 Reducing Non-Physical Stiffness on Quadrilateral Meshes: Numerical Verification
8.2.3 Steady-State Calculation with NACA0012 Airfoil
8.3 Speedup of Parallel DGSEM in Two and Three Spatial Dimensions
8.3.1 Quadrilateral Results
8.3.2 Hexahedral Results

# Summary and Conclusions

9.1 Conclusions

References
Biographical Sketch
LIST OF TABLES

6.1 Estimated factor of increase in storage complexity for AB3LTS versus global time stepping LSRK3 for various $N$ ................................................................. 74

6.2 Estimations of work reduction for AB3LTS versus LSRK3 on four meshes ........................................ 78

8.1 Computed error of a constant solution to the wave equation on a moving mesh with local time stepping ................................................................................. 113

8.2 Maximum error for a constant solution to the wave equation on a three dimensional mesh with local time stepping ................................................................. 115

8.3 Error for a constant solution to the wave equation on a three dimensional moving mesh with local time stepping ................................................................. 115

8.4 Comparison of the predicted and measured speedup of AB3LTS versus the global time stepping LSRK3 on several static meshes ................................................. 118

8.5 Comparison of the predicted and measured speedup of AB3LTS versus the global time stepping LSRK3 on several moving meshes .............................................. 119

8.6 LSRK3 versus AB3LTS to progress a solution of the Euler equations (8.2.4) to steady state ................. 120
### LIST OF FIGURES

3.1 Oblique reflection from a moving boundary normal to the $x'$--axis. ........................................ 19

3.2 The reflection coefficient for TE and TM polarization as a function of the angle of incidence. The special cases of Brewster’s angle and TIR are labeled. .......................... 29

3.3 Wave propagation through a thin film of low refractive index, $n_2$, between two high-index semi-infinite media, $n_1$ and $n_3$. ................................................................. 31

3.4 The reflection and transmission coefficients for a moving vertical interface as functions of $\beta$. We take $c_L = 1$, $c_R = 0.8$, and $\theta_i = \frac{\pi}{4}$. .......................................................... 33

3.5 The dispersion relation in moving object as a function of $\beta$ with $c_L = 1$ and $c_R = 0.8$ (left) Fixed propagation angle $\theta = \pi/4$. (right) Fixed wavenumber $k = 1$. ....................... 34

4.1 A three-dimensional example of a map from computational coordinates to physical coordinates................................................................. 43

5.1 Left and right states viewed at a moving boundary. The boundary moves with velocity $v_o$. ................................................................. 47

6.1 (a) Illustration of synchronization levels using three elements. (b) Each element evolves one local time step from a synchronization level. Note that the largest element is immediately at the next synchronization level. ........................ 66

6.2 (a)–(c) The progression of repeated application of the evolve condition to a three element mesh. Elements ready to evolve are shaded. (d) Each element has reached the next synchronization level. .................................................. 67

6.3 (a) Solution history for $e_1$ on the boundary and projection of the solution onto the boundary at $t^{n+1}$. (b) Three nodes used to create the polynomial interpolant in time. (c) Fully constructed polynomial interpolant in time. ......................... 68

6.4 Several two dimensional spectral element meshes. ................................................................. 76

6.5 Histograms depict the distribution of element sizes for four test meshes. ..................... 77

7.1 A mesh where each partition has the same approximate number of elements. ........... 83

7.2 Partition of a mesh where elements are weighted according to size. ......................... 84

8.1 Reflection and transmission of a TE Gaussian plane wave from a moving dielectric interface. Dashed lines are negative contours, the moving dielectric is shaded gray, and the overlay of squares shows the locations of the element boundaries. .......... 90
8.2 Semi-log plot shows the spectral convergence of the approximation for reflection and transmission from a normal moving dielectric interface. ......................... 91

8.3 Reflection and transmission of a TE Gaussian plane wave from an oblique moving dielectric interface at two times. Dashed lines are negative contours. The moving dielectric is shaded gray. .............................................. 92

8.4 Semi-log plot shows the spectral convergence of the approximation for the oblique moving interface problem. .................................................. 92

8.5 Reflection of a TE Gaussian plane wave from an oblique moving mirror. Dashed lines are negative contours. The moving mirror is shaded gray. ..................... 93

8.6 Semi-log plot shows the spectral convergence of the approximation for the oblique moving mirror problem. .................................................. 93

8.7 (a) Error in the transmitted $D_z$ electric field for three values of $N$. (b) Spectral convergence for incident angles $\theta_i = \pi/10$ and $\theta_i = 2\pi/5$. ......................... 94

8.8 Reflection and transmission of a TE plane wave from a moving dielectric wave scatterer at four times. Dashed lines are negative contours, the moving dielectric object is shaded gray, and the overlay of squares shows the locations of the element boundaries. 96

8.9 Semi-log plot shows the spectral convergence of the approximation for the reflection and transmission from an oscillating cylinder. .......................... 97

8.10 Total internal reflection of a TE Gaussian plane wave. We see the finite penetration depth of the evanescent wave into the lighter material. The dielectric material is shaded gray. .............................................. 97

8.11 Semi-log plot shows the spectral convergence of the approximation for total internal reflection. .................................................. 98

8.12 Frustrated total internal reflection of a TE Gaussian plane wave. Note the evanescent wave allows the plane wave to jump a small gap (in this case of air). The dielectric materials are shaded gray. .............................................. 99

8.13 Reflection and transmission of a TM Gaussian plane wave from a moving dielectric interface initially located at $x = 0$. The moving dielectric is shaded gray. ................. 100

8.14 Semi-log plot shows the spectral convergence and design time accuracy of the approximation. .................................................. 100

8.15 Reflection and transmission of a TM Gaussian plane wave from an oblique moving dielectric interface. The moving dielectric is shaded gray. ..................... 101

8.16 Semi-log plot shows the spectral convergence of the approximation for the oblique moving interface problem. .................................................. 101
8.17 Reflection of a TM Gaussian plane wave from an oblique moving mirror. The moving mirror is shaded gray. ................. 102

8.18 Semi-log plot shows the spectral convergence of the approximation for the oblique moving mirror problem. .................. 103

8.19 Pure transmission of a TM Gaussian plane wave traveling at Brewster’s angle. The dielectric material is shaded gray. ............. 103

8.20 Semi-log plot shows the spectral convergence of the approximation for Brewster’s angle problem. ................................. 104

8.21 Reflection and transmission of the pressure plane wave from a moving material interface initially located at $x = 0$. The moving medium is shaded gray. ......................... 105

8.22 Semi-log plot showing exponential convergence of the error for the numerical solution of the constantly moving vertical material interface test problem. .................. 105

8.23 Reflection and transmission of the pressure from a moving oblique material interface with a normal angle $\theta = \pi/16$. The moving medium is shaded gray. (a) The pressure, $p$, of the initial wave pulse. (b) The reflection and transmission of the incident wave at a later time. ................................. 106

8.24 Semi-log plot showing exponential convergence of the error for the numerical solution of the constantly moving oblique material interface. ................................. 106

8.25 Pure reflection of the pressure from an oblique moving wall, which is shaded gray. .. 107

8.26 Semi-log plot shows the spectral convergence of the approximation for the oblique moving wall problem. .......................... 108

8.27 (a) Error in the transmitted pressure $p$ for three values of $N$. (b) Spectral convergence for incident angles $\theta_i = \pi/10$ and $\theta_i = 2\pi/5$. ................................. 108

8.28 Contour plots of the plane wave propagation across a refined mesh. The overlay of squares shows the element boundaries. ......................... 110

8.29 (left) Semi-log plot of the spectral convergence for the DGSEM with AB3LTS on a refined mesh. (right) A log-log plot of the third order temporal accuracy of AB3LTS in each group of elements. ................................. 111

8.30 Contour plots of the plane wave propagation across a refined mesh with a moving circular boundary. The overlay of quadrilaterals shows the element boundaries. ........ 112

8.31 (left) Semi-log plot shows spectral convergence for the ALE-DGSEM with AB3LTS. (right) A log-log plot showing third order temporal accuracy of AB3LTS in each group of elements. ................................. 112
8.32 Pseudocolor plots of plane wave propagation across a refined mesh. The overlay of hexagons show element boundaries. ................................................................. 114

8.33 (left) Semi-log plot of the spectral convergence for the DGSEM with AB3LTS on a 3D refined mesh. (right) A log-log plot of the third order temporal accuracy of AB3LTS in each group of elements. ................................................................. 114

8.34 Pseudocolor plots of plane wave propagation across a refined mesh. The overlay of hexagons show element boundaries. ................................................................. 116

8.35 (left) Semi-log plot of the spectral convergence for the DGSEM with AB3LTS on a moving 3D refined mesh. (right) A log-log plot of the third order temporal accuracy of AB3LTS in each group of elements. ................................................................. 116

8.36 Comparison of the convergence of AB3LTS and LSRK3 for steady flow over the NACA0012 airfoil. ................................................................. 121

8.37 Quadrilateral mesh used in two dimensional parallel speedup tests. ................................................................. 122

8.38 Speedup of two dimensional parallel DGSEM approximation. ................................................................. 122

8.39 Speedup of two dimensional parallel DGSEM approximation with local time stepping. 123

8.40 Comparison of speedup for parallel global and local time stepping relative to their own serial implementation. ................................................................. 124

8.41 A shootout between the parallel implementations of a standard time integrator against the local time stepping integrator. ................................................................. 125

8.42 Hexahedral mesh used in three dimensional parallel speedup tests. ................................................................. 126

8.43 Speedup of three dimensional parallel DGSEM approximation. ................................................................. 127

8.44 Speedup of three dimensional parallel DGSEM approximation with local time stepping. 127
ABSTRACT

This dissertation develops and evaluates a computationally efficient and high-order numerical method to compute wave reflection and transmission from moving material boundaries. We use a discontinuous Galerkin spectral element approximation with an arbitrary Lagrangian-Eulerian mapping and derive the exact upwind numerical fluxes to model the physics of wave reflection and transmission at jumps in material properties. Spectral accuracy is obtained by placing moving material interfaces at element boundaries and solving the appropriate Riemann problem. We also derive and evaluate an explicit local time stepping (LTS) integration for the DGSEM on moving meshes. The LTS procedure is derived from Adams-Bashforth multirate time integration methods. We present speedup and memory estimates, which show that the explicit LTS integration scales well with problem size. The LTS time integrator is also highly parallelizable. The manuscript also gathers, derives and analyzes several analytical solutions for the problem of wave reflection and transmission from a plane moving material interface. We present time-step refinement studies and numerical examples to show the approximations for wave reflection and transmission at dielectric and acoustic interfaces are spectrally accurate in space and have design temporal accuracy. The numerical tests also validate theoretical estimates that the LTS procedure can reduce computational cost by as much as an order of magnitude for time accurate problems. Finally, we investigate the parallel speedup of the LTS integrator and compare it to a standard, low-storage Runge-Kutta method.
CHAPTER 1

INTRODUCTION

Modeling the reflection and transmission of a wave from a moving material interface is important for a variety of applications in electromagnetism and acoustics. Examples include radio communication with moving spaceships [70], and ultrasound scanners, e.g., imaging the human lung [67]. Further examples include wave scattering from vibrating surfaces in electromagnetism [7, 14, 24, 42, 44, 60], acoustics [52, 71, 73], and biology [4, 5, 25, 50, 54, 66].

The reflection and transmission of waves from moving material interfaces differ from reflection and transmission at stationary interfaces. The problem of wave reflection and transmission from a moving dielectric medium has been examined theoretically by many authors [46, 69, 75, 79, 83, 84, 87, 98], whereas the classical wave equation in a moving medium has received limited attention [52, 62, 80]. The physics of the problem changes as the motion introduces Doppler shifts in the frequencies, shifts in the amplitudes, and the possible compression of the medium [14, 24].

Computationally, electromagnetic scattering from a uniformly moving body has been treated by Harfoush and Taflove [42] with the finite-difference time-domain (FDTD) method applied directly to Maxwell’s equations. Recently, Ho [44] reported a computational solution of the scattering from moving boundaries and in a moving dielectric medium [45] in one dimension. Both approaches used rigid meshes.

A common way to approximate solutions to problems with moving boundaries is to use an arbitrary Lagrangian-Eulerian (ALE) formulation [1]. In the ALE formulation, one maps a time dependent domain Ωt that has moving boundaries onto a fixed reference domain Ω. In the process, conservation law equations in the original domain are transformed to conservation law equations in the reference domain [29, 64, 68]. In the numerical approximation on the reference domain, the new set of equations depends on the mesh velocity.

To design methods to solve wave propagation problems on a moving domain we will subdivide Ωt into K elements, use an ALE formulation on each element, and discretize on the reference domain with a high-order discontinuous Galerkin (DG) spectral method [15]. The DG method approxi-
mates the solution for reflection and transmission of a wave from a material interface with spectral accuracy, if one simply ensures that an element boundary is placed on the material discontinuity [56]. The development and application of DG spectral methods for static domains are reviewed in [12, 15, 56].

From the perspective of computational efficiency, a problem with a high-order discontinuous Galerkin spectral element method (DGSEM) is that the systems are stiff. The maximum stable time step allowed by the Courant-Friedrichs-Lewy (CFL) condition is dictated by the smallest elements in the mesh and the polynomial order of the approximation [15, 16]. If refinement in the mesh is restricted to a small region, then a very small time step in the entire computational domain has a high computational cost [22]. Also, global time stepping methods are computationally inefficient on moving meshes. They must recompute the geometry of large moving elements on the time scale of the small elements. For the ALE-DGSEM the computation of an element’s geometric properties is expensive [13, 56]. If the time step needed for stability, $\Delta t_{stab}$ is less than the time scale required for accuracy, $\Delta t_{acc}$, the problem is stiff for an unphysical reason.

The unphysical stiffness is increased since the selection of stable time steps for the DGSEM on quadrilateral meshes with local refinement is ad hoc. Toulorge and Desmet [91] studied the effect that a triangular element’s shape has on the time step of the DGSEM. They found a precise relationship between the shortest edge of a triangle and the maximum Courant number for $N \leq 10$. Similar studies are not yet available for quadrilaterals.

A natural response to the restrictive time steps required for stability by explicit global time stepping methods is to switch to implicit methods. Unfortunately, the matrices in spectral methods are dense, so that the matrix solvers for implicit schemes are expensive [57]. To be effective, the increase in the size of the time step must be enough to offset the increased work per time step. For steady-state problems, when time accuracy is not an issue, implicit methods perform well [11, 57, 93]. However, for three dimensional problems or as the polynomial order $N$ increases, the memory requirements to store arrays for linear solvers and preconditioners may become prohibitive [57, 93]. For time accurate computations, where time steps are limited also by accuracy, much of the advantage of implicit methods can be lost [57].

Alternatively, explicit local time stepping (LTS) schemes have been developed to allow each element, or groups of elements, to evolve at their largest, stable time step [18, 27, 33, 37, 40]
while retaining time accuracy. For instance, Constantinescu and Sandu [17, 18] devised second order multirate explicit methods for hyperbolic conservation laws, using Runge-Kutta and Adams-Bashforth schemes, respectively.

Discontinuous Galerkin methods are well-suited for the development of explicit LTS integrators because the approximation is local to each element [56]. For example, Lörcher et al. [65] derive consistent, high order LTS for discontinuous Galerkin methods that use Cauchy-Kovalevskaya time integration. The Cauchy-Kovalevskaya procedure also motivated the development of an arbitrary high-order derivatives (ADER) discontinuous Galerkin approximation for elastic wave equations [27] and the 3D Maxwell’s equations [88]. Combining the idea of a multirate linear multistep method of Gear and Wells [34], Stock [85] derived many two-rate Adams-Bashforth LTS schemes. For the damped wave equation, Grote and Mitkova [40] derived continuous and discontinuous Galerkin approximations with Adams-Bashforth schemes up to order four. A second order LTS integrator that conserves the discrete energy in each element was proposed by Ezziani and Joly [31]. Krivodonova [59] developed a conservative, second order LTS strategy using Heun’s method for use with non-linear conservation laws. Finally, Gassner et al. [33] created a high-order accurate predictor-corrector method for local time stepping using continuous extension Runge-Kutta (CERK).

1.1 Goals of the Dissertation

The main goal of this dissertation is to propose, derive, and evaluate methods to approximate wave reflection and transmission from a moving material interface efficiently and accurately. To do so we provide a detailed description of the relevant physics as well as the mathematics to derive exact, upwind numerical fluxes for a DG approximation. The computation of the numerical flux at a moving material interface or within a moving object must incorporate the altered physics of the moving material. As was noted previously, this physics differs from classical physics [72, 84].

The DGSEM with an ALE mapping approximation is a feasible approach to compute scattering waves from moving boundaries [1]. Generally, DG spectral element methods are well-suited to model the solution of wave propagation problems because of exponentially low dissipation and dispersion errors [3]. A discontinuous Galerkin approximation can resolve a wave using three to four points per wavelength [3]. In contrast, a high-order finite difference scheme typically requires six to eight
points per wavelength to fully resolve a wave [23]. The discontinuous Galerkin approximation is easily extended to solve problems on complex geometries, for details see [13, 55].

For computational efficiency we design and evaluate an explicit LTS integrator for use with the DGSEM on moving or static meshes with local refinement. We also derive estimates of memory requirements and CPU time savings. We follow an approach similar to that of Stock [85], except that the LTS procedure is simplified by the weak coupling in the approximation. Also, the DG approximation is almost entirely local to an element which makes the DGSEM an excellent candidate for parallel computations [8]. We demonstrate that the new time integration technique remains highly parallelizable.

1.2 Contributions of the Dissertation

The original contributions of this dissertation are the following:

- We derive exact, upwind numerical fluxes at moving material interfaces for both electromagnetic and acoustic problems.
- We develop and analyze a local time stepping integrator to reduce stiffness due to local mesh refinement. The method is valid on moving or static meshes.
- We demonstrate that the approximations including the newly developed local time stepping strategy can be parallelized.
- We compute analytical solutions for reflection and transmission problems with a Gaussian incident wave.

Parts of this thesis have been presented and published in the Journal of Scientific Computing [96] and the Journal of Computational Physics [95]. Other parts are under review for publication [97].

1.3 Organization of the Dissertation

In Chap. 2 we present the physical model for wave reflection and transmission at moving material interfaces. In Chap. 3 we derive and analyze the analytical solutions to be used as comparisons to numerical solutions. In Chap. 4 we present the discontinuous Galerkin spectral element method (DGSEM) where the Arbitrary Lagrangian Eulerian mapping (ALE) is used to move the mesh. In Chap. 5 we derive exact, upwind numerical fluxes through the solution of an
appropriate Riemann problem at the moving material interface. In Chap. 6 we derive a high-order
time integration method that uses local time stepping (LTS). In Chap. 7 we introduce basic aspects
of high performance computing and outline the parallelization of the DGSEM and LTS. In Chap.
8 we present numerical results. Chap. 9 presents a summary of the results obtained so far and
conclusions.
CHAPTER 2

THE PHYSICS: MODELING WAVE REFLECTION AND TRANSMISSION FROM A MOVING INTERFACE

To accurately describe the moving interface, wave reflection and transmission problem we review the relevant physics. To discuss wave propagation in moving materials we introduce several physical results in Sec. 2.1 including polarization of plane waves and the Lorentz transformation. The phase angles of plane electromagnetic and acoustic waves obey the same transformations [72]. Thus, the discussion in Sec. 2.1 is formulated using electromagnetic waves but is applicable to acoustic waves. Using these physical tools we derive, in Sec. 2.2, the governing equations and constitutive relations for Maxwell’s equations and the classical wave equation in constantly moving frames of reference.

2.1 Physical Tools for Moving Material Problems

The governing equations of electrodynamics in a moving medium are the source-free Maxwell’s equations

\[
\begin{align*}
\frac{\partial \mathbf{B}}{\partial t} &= \nabla \times \mathbf{E}, \\
\frac{\partial \mathbf{D}}{\partial t} &= \nabla \times \mathbf{H}, \\
\nabla \cdot \mathbf{B} &= 0, \\
\nabla \cdot \mathbf{D} &= 0,
\end{align*}
\]  

(2.1.1)

where \( \mathbf{E} \), \( \mathbf{D} \) are electric field vectors and \( \mathbf{B} \), \( \mathbf{H} \) are magnetic field vectors. Until constitutive relations between the four field vectors are specified, Maxwell’s equations (2.1.1) are insufficient to produce a unique solution [87].

The behavior of electromagnetic plane waves at a dielectric interface depends upon their polarization. For this purpose the plane of incidence is defined as the plane of projection of an incident wave vector \( \mathbf{k}_i \) upon the interface. One cartesian axis is traditionally defined as being normal to this plane of incidence [49]. Any pair of orthogonally polarized uniform plane waves can be superimposed to achieve an arbitrary wave polarization. As is customary, we recognize two simple types
of incident electromagnetic waves that can be superimposed: Transverse electric (TE) waves where the electric field has zero component along the $z$-axis of propagation and non-zero components along the transverse $x - y$ direction. Transverse magnetic (TM) waves are defined in a similar fashion where the magnetic field is zero in the $z$ direction and non-zero in the $x - y$ plane.

The principle of duality, used to switch between the TE and TM polarizations in the current and charge-free Maxwell’s equations, allows one to transform results for the TE polarization to the TM polarization. The TE and TM polarizations are dual problems [72] under the transformations,

$$
\begin{align*}
E & \rightarrow H, \quad D \rightarrow B, \\
H & \rightarrow -E, \quad B \rightarrow -D, \\
\varepsilon & \leftrightarrow \mu.
\end{align*}
$$

(2.1.2)

If we apply the duality transformations (2.1.2) to the current and charge-free Maxwell’s equations (2.1.1) we find

$$
\begin{align*}
-\frac{\partial B}{\partial t} &= \nabla \times E \rightarrow \frac{\partial D}{\partial t} = \nabla \times H, \\
\frac{\partial D}{\partial t} &= \nabla \times H \rightarrow \frac{\partial B}{\partial t} = -\nabla \times E, \\
\nabla \cdot B &= 0 \rightarrow \nabla \cdot D = 0, \\
\nabla \cdot D &= 0 \rightarrow \nabla \cdot B = 0,
\end{align*}
$$

(2.1.3)

where the transformed set of equations on the right-hand side of (2.1.3) is the same as the original Maxwell’s equations, on the left, although sequenced differently. Thus, any solution of Maxwell’s equations also solves the dual problem.

When a wave is incident upon a dielectric interface between two materials of differing properties, a portion of the wave is reflected and a portion is transmitted. To ensure that the incident, reflected, and transmitted waves have the same phase, parallel components of all three waves must satisfy the boundary conditions

$$
\begin{align*}
E_{i,\parallel} + E_{r,\parallel} &= E_{T,\parallel} \\
H_{i,\parallel} + H_{r,\parallel} &= H_{T,\parallel},
\end{align*}
$$

(2.1.4)

everywhere along a non-conducting surface [49]. The boundary conditions (2.1.4) for the TE or TM polarization remain the same, a fact easily verified by duality (2.1.2). Therefore, TE and TM waves travel with the same phase angle but typically are reflected and transmitted with different amplitudes [72]. The fact that all TE and TM waves have the same phase simplifies the outline of analytical solutions in Chap. 3.
To relate the electric and magnetic fields in a moving dielectric material, we introduce two frames of reference that move relative to one another. We use the Lorentz transformation to switch between moving frames of reference [28, 72]. We adopt the notation of the laboratory frame, $\mathcal{S}$, where the material interface moves, and the reference frame, $\mathcal{S}'$, where the material interface appears at rest. The convention we use here is that primed coordinates represent those in $\mathcal{S}'$ and unprimed coordinates are those in $\mathcal{S}$. Formally, the Lorentz transformation describes a linear space-time transformation between two coordinate systems moving relative to each other with a constant velocity [72]. Maxwell’s equations and the d’Alembert operator

$$\Box = \Delta - \frac{1}{c^2} \frac{\partial}{\partial t},$$

both remain invariant under Lorentz transformations [72], which can be verified by the chain rule or through tensor algebra.

The first form of the Lorentz transformation that we consider is where a moving frame, $\mathcal{S}'$, moves relative to a fixed frame, $\mathcal{S}$, along the $x-$axis at a constant velocity $v_x = (\nu_x, 0, 0)$. If we write the space-time coordinates in $\mathcal{S}$ as $\mathbf{x} = \{t \ x \ y \ z\}$, and $\mathbf{x}' = \{t' \ x' \ y' \ z'\}$ for the space-time coordinates of $\mathcal{S}'$, then $\mathbf{x}$ and $\mathbf{x}'$ are related by the Lorentz transformation

$$t' = \gamma (t - \frac{\nu_x}{c^2} x),$$

$$x' = \gamma (x - \nu_x t),$$

$$y' = y,$n

$$z' = z,$n

where

$$\gamma = \frac{1}{\sqrt{1 - \beta^2}}, \quad \beta = \frac{\nu_x}{c}.$$  

Sometimes the transformation (2.1.6) is called the $x$-directed Lorentz boost [72]. If we use $\beta$ and the scaled quantity $\tau = ct$, then the Lorentz transformation (2.1.6) may be written compactly as

$$\tau' = \gamma (\tau - \beta x),$$

$$x' = \gamma (x - \beta \tau),$$

$$y' = y,$n

$$z' = z.$$  

8
The inverse Lorentz transformation is found if we exchange the primed and unprimed coordinates and negate the sign of $\beta$.

The transformations (2.1.8) indicate that one frame is boosted to move relative to the other. Interchanging the roles of $x$ and $y$, or $x$ and $z$, one obtains the Lorentz transformations for motion along the $y$ or $z$ directions, respectively. One expresses the transformations (2.1.8) in matrix form as

\[
\begin{bmatrix}
\tau' \\
x' \\
y' \\
z'
\end{bmatrix} =
\begin{bmatrix}
\gamma & -\gamma \beta & 0 & 0 \\
-\gamma \beta & \gamma & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\tau \\
x \\
y \\
z
\end{bmatrix}.
\]

(2.1.9)

The most general Lorentz boost corresponding to an arbitrary velocity $v_o = (\nu_x, \nu_y, \nu_z)$ is given in matrix form by [72]

\[
L = \begin{bmatrix}
\gamma & -\gamma \beta T \\
-\gamma \beta & I + \frac{\gamma^2}{\gamma^2 + 1} \beta \beta T
\end{bmatrix}, \quad \text{where} \quad \beta = \frac{v_o}{c}, \quad \gamma = \frac{1}{\sqrt{1 - \beta \cdot \beta}}.
\]

(2.1.10)

We generalize the Lorentz transformation for a frame $\mathcal{S}'$ moving in the $x$–direction rotated at angle $\theta$ with normal vector $\hat{n} = (\cos \theta, \sin \theta)$. We write the $x$–directed Lorentz transformation as the linear operator $L(N)$ [48]:

\[
L(N) = \begin{bmatrix}
\gamma & -\frac{N}{c \ell} \gamma & 0 & 0 \\
-\frac{N}{c \ell} \gamma & \gamma & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix},
\]

(2.1.11)

where $N = \hat{n} \cdot v_o$. Note that $L(N)L(-N) = L(-N)L(N) = I$, i.e., $L^{-1}(N) = L(-N)$.

We use four-vector properties to describe the phase angles of the reflected and transmitted waves in the laboratory frame $\mathcal{S}$. A four-vector is a four-dimensional vector that transforms like the space-time vector under a Lorentz transformation [72]. The frequency-wavenumber vector is a four-vector. Under the $x$–directed transformation, the frequency-wavenumber vector in $\mathcal{S}'$ will be

\[
\begin{align*}
\omega' &= \gamma (\omega - \beta c k_x), \\
k'_x &= \gamma \left( k_x - \frac{\beta}{c} \omega \right), \\
k'_y &= k_y, \\
k'_z &= k_z,
\end{align*}
\]

(2.1.12)
where the inverse four-vector transformation is found by exchanging the primed and unprimed coordinates and negating the sign of $\beta$. Notice that the frequency-wavenumber transformation (2.1.12) exhibits the relativistic Doppler effect [49].

An immediate consequence of the four-vector transformation is that a plane wave remains a plane wave in all constant moving frames of reference [72], i.e.,

$$
\phi = \omega t - k_x x - k_y y - k_z z = \omega^t - k_x' x - k_y' y - k_z' z = \phi'.
$$

Result (2.1.13) follows from a property of a four-vector couple: For any two four-vectors $R$ and $S$, the quadratic form $R^T G S$ remains invariant under a Lorentz transformation. That is, $R^T G S' = R^T G S$, where $G = \text{diag}(1, -1, -1, -1)$ is a diagonal metric matrix. For the purposes of our later examples, we consider waves propagating in the $x - y$ plane only. Therefore in (2.1.13), the $z$-component $k_z = 0$.

The electromagnetic fields are related through a six-vector transformation [72]. The six-vector pairs $(E, cB)$ and $(H, cD)$ have the Lorentz transformation properties

$$
E'_\perp = \gamma (E_\perp + c\beta \times B),
$$

$$
B'_\perp = \gamma \left( B_\perp - \frac{1}{c} \beta \times E \right),
$$

$$
E'_\parallel = E_\parallel,
$$

$$
B'_\parallel = B_\parallel,
$$

and

$$
H'_\perp = \gamma (H_\perp - c\beta \times D),
$$

$$
D'_\perp = \gamma \left( D_\perp + \frac{1}{c} \beta \times H \right),
$$

$$
H'_\parallel = H_\parallel,
$$

$$
D'_\parallel = D_\parallel,
$$

where $\parallel$ and $\perp$ denote parallel and perpendicular vector components respectively. We use the six-vector transformation to derive the constitutive relations in a moving dielectric material as well as the reflection and transmission coefficients for analytical solutions of Maxwell’s equations.

## 2.2 Governing Equations for Wave Propagation Problems

We now derive, for moving media, the governing equations and constitutive relations for Maxwell’s equations and the classical wave equation. The details of the physical analysis for the two wave
propagation models differ [62, 98]. Most prominently the form of Maxwell’s equations is invariant in all constantly moving frames of reference [28, 87] but the constitutive relations change [69, 84], whereas the form of the classical wave equation will change in different reference frames [62] but the constitutive relations remain the same [78].

2.2.1 Maxwell’s Equations

As the form of Maxwell’s equations is invariant in constantly moving frames of reference [28, 87], the motion of media affects the constitutive relations that relate electric and magnetic fields. We follow a formulation due to Minkowski [69] to describe a covariant theory for electrodynamics in moving media. We change between the two frames of reference with the general Lorentz transformation (2.1.10) where the material interface may move uniformly with an arbitrary velocity

$$\mathbf{v}_o = \nu_x \hat{x} + \nu_y \hat{y} + \nu_z \hat{z}.$$  

We determine the constitutive relations for the problem where free space is to the left of a plane moving homogeneous, isotropic dielectric material. In the reference frame $S'$ we know that to the left of the material interface, which we locate at $x' = x'_0$, waves travel at the speed of light, i.e., $c = c_L = 1/\sqrt{\varepsilon_L \mu_L}$. To the right of the dielectric interface, waves travel with wavespeed $c_R = 1/\sqrt{\varepsilon_R \mu_R}$, where $\varepsilon$ and $\mu$ and the permittivity and permeability of the dielectric medium respectively. On either side of the dielectric interface in the reference frame $S'$ we assume that the permittivity and permeability are piecewise constant

$$\varepsilon = \begin{cases} \varepsilon_L & \text{if } x' \leq x'_0, \\ \varepsilon_R & \text{if } x' > x'_0, \end{cases} \quad \text{and} \quad \mu = \begin{cases} \mu_L & \text{if } x' \leq x'_0, \\ \mu_R & \text{if } x' > x'_0. \end{cases} \tag{2.2.1}$$

The moving dielectric will affect the constitutive relations [69, 87, 79]. We see that the piecewise constant permittivity and permeability do not appear explicitly in Maxwell’s equations (2.1.1). The electric and magnetic fields depend implicitly on the discontinuous dielectric properties and the velocity of the dielectric medium, $\mathbf{v}_o$.

The constitutive relations that relate $\mathbf{E}$, $\mathbf{D}$ and $\mathbf{H}$, $\mathbf{B}$ change depending on the frame of reference. When the medium is at rest, i.e., in $S'$, the constitutive relations between two homogeneous, isotropic, static dielectrics are

$$\mathbf{D}' = \varepsilon \mathbf{E}' \quad \text{and} \quad \mathbf{B}' = \mu \mathbf{H}'.$$  

(2.2.2)
To transform the electric and magnetic fields between constant frames of reference we use the six-vector transformations (2.1.14) and (2.1.15) to see that

\[
\begin{align*}
\mathbf{D} + \frac{1}{c^2} \mathbf{v}_o \times \mathbf{H} &= \varepsilon (\mathbf{E} + \mathbf{v}_o \times \mathbf{H}), \\
\mathbf{B} - \frac{1}{c^2} \mathbf{v}_o \times \mathbf{E} &= \mu (\mathbf{H} - \mathbf{v}_o \times \mathbf{D}).
\end{align*}
\] (2.2.3)

We can express the quantities \( \mathbf{D} \) and \( \mathbf{H} \) solely in terms of the quantities \( \mathbf{E} \) and \( \mathbf{B} \) if we use the double vectorial products

\[
\begin{align*}
\mathbf{v}_o \times (\mathbf{v}_o \times \mathbf{D}) &= \mathbf{v}_o (\mathbf{v}_o \cdot \mathbf{D}) - v_o^2 \mathbf{D}, \\
\mathbf{v}_o \times (\mathbf{v}_o \times \mathbf{E}) &= \mathbf{v}_o (\mathbf{v}_o \cdot \mathbf{E}) - v_o^2 \mathbf{E}.
\end{align*}
\] (2.2.4)

We then find the relativistic expression for the excitations as a function of the electromagnetic fields in the laboratory frame [79]

\[
\begin{align*}
\mathbf{D} &= \varepsilon \mathbf{E} + \gamma^2 \left( \varepsilon - \frac{1}{\mu c^2} \right) \mathbf{v}_o \times \left( \mathbf{B} - \frac{\mathbf{v}_o \times \mathbf{E}}{c^2} \right), \\
\mathbf{H} &= \frac{\mathbf{B}}{\mu} + \gamma^2 \left( \varepsilon - \frac{1}{\mu c^2} \right) \mathbf{v}_o \times (\mathbf{E} + \mathbf{v}_o \times \mathbf{B}).
\end{align*}
\] (2.2.5)

The constitutive relations (2.2.3) simplify if we separate them in terms of the parallel, \( \parallel \), and perpendicular, \( \perp \), components [84]. In terms of these components (2.2.3) becomes

\[
\begin{align*}
\mathbf{D}_\parallel &= \varepsilon \mathbf{E}_\parallel, \quad \mathbf{B}_\parallel = \mu \mathbf{H}_\parallel,
\end{align*}
\] (2.2.6)

and

\[
\begin{align*}
\begin{cases}
\mathbf{D}_\perp \\
\mathbf{B}_\perp
\end{cases} = \begin{cases}
\varepsilon (1 - \beta^2) \mathbf{E}_\perp + (\varepsilon \mu - \varepsilon_L \mu_L) \mathbf{v}_o \times \mathbf{H} \\
\mu (1 - \beta^2) \mathbf{H}_\perp + (\varepsilon_L \mu - \varepsilon \mu_L) \mathbf{v}_o \times \mathbf{E}
\end{cases},
\end{align*}
\] (2.2.7)

where \( \beta = \| \beta \| = \sqrt{\nu_\perp^2 + \nu_\parallel^2 + \nu_\perp^2 / c}, \gamma = 1/\sqrt{1 - \beta \cdot \beta}, \) and the permittivity \( \varepsilon \) and permeability \( \mu \) depend on the spatial variable \( x \).

Note that in free space, when \( \varepsilon = \varepsilon_L \) and \( \mu = \mu_L \), the constitutive relations (2.2.6) and (2.2.7) become those for the stationary, homogeneous, isotropic medium \( \mathbf{D} = \varepsilon \mathbf{E} \) and \( \mathbf{B} = \mu \mathbf{H} \), which is the well-known result that Maxwell’s equations in free space are invariant in any constantly moving frame of reference.

We first derive the Maxwell’s equations in a moving medium for TE polarized waves. We apply the TE results and the principle of duality (2.1.2) to simplify the discussion of the governing equations for the TM polarization.
For the TE problem, we take the electric field to lie only in the $z$–direction, $\mathbf{E} = (0, 0, E_z)$, and the magnetic field to lie in the $x$–$y$ plane, $\mathbf{H} = (H_x, H_y, 0)$. The unknowns for the TE system are $D_z$, $B_x$, and $B_y$ and Maxwell’s equations take the form of a conservation law

$$ q + \nabla \cdot \mathbf{F}_{TE} = 0, \quad (2.2.8) $$

where

$$ q = \begin{bmatrix} D_z \\ B_x \\ B_y \end{bmatrix}, \quad \mathbf{F}_{TE}^{x} = \begin{bmatrix} -e_i \times \mathbf{H} \\ 0 \end{bmatrix}, \quad \mathbf{F}_{TE}^{y} = \begin{bmatrix} H_z \\ 0 \end{bmatrix}. \quad (2.2.9) $$

Explicitly,

$$ q = \begin{bmatrix} D_z \\ B_x \\ B_y \end{bmatrix}, \quad \mathbf{F}_{TE}^{x} = \begin{bmatrix} -H_y \\ 0 \end{bmatrix}, \quad \mathbf{F}_{TE}^{y} = \begin{bmatrix} H_z \\ 0 \end{bmatrix}. \quad (2.2.10) $$

Next we rewrite $H_x$, $H_y$, and $E_z$ in terms of the unknown quantities $D_z$, $B_x$, and $B_y$. To do so we consider the general velocity in the $x$–$y$ plane $\mathbf{v}_o = (\nu_x, \nu_y, 0)$. In terms of the normal velocity, the parallel and perpendicular components of the electric field vector are

$$ \mathbf{E}_\parallel = \mathbf{E} \cdot \hat{\beta} = 0, \quad \text{and} \quad \mathbf{E}_\perp = \mathbf{E} - \hat{\beta} \mathbf{E}_\parallel = \mathbf{E}, \quad (2.2.11) $$

where

$$ \hat{\beta} = \frac{\beta}{||\beta||} = \frac{\mathbf{v}_o}{\sqrt{\nu_x^2 + \nu_y^2}}. \quad (2.2.12) $$

The parallel and perpendicular magnetic field components are

$$ \mathbf{H}_\parallel = \mathbf{H} \cdot \hat{\beta} = \frac{\nu_x H_x + \nu_y H_y}{\sqrt{\nu_x^2 + \nu_y^2}} \quad (2.2.13) $$

and

$$ \mathbf{H}_\perp = \mathbf{H} - \hat{\beta} \mathbf{H}_\parallel = \mathbf{H} - \frac{\nu_x H_x - \nu_y H_y}{\nu_x^2 + \nu_y^2} = \left( \frac{\nu_y^2 H_x - \nu_x \nu_y H_y}{\nu_x^2 + \nu_y^2}, \frac{\nu_x^2 H_y - \nu_x \nu_y H_x}{\nu_x^2 + \nu_y^2}, 0 \right). \quad (2.2.14) $$

Now we can use the relativistic constitutive relations (2.2.6) and (2.2.7) to define $\mathbf{D}$ and $\mathbf{B}$ in terms of $\mathbf{E}$ and $\mathbf{H}$. First, for the electric field $\mathbf{D}$

$$ \left( 1 - \frac{\varepsilon \mu}{\varepsilon L \mu L} \beta^2 \right) \begin{bmatrix} D_x \\ D_y \\ D_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \varepsilon (1 - \beta^2) E_z - \nu_y (\varepsilon \mu - \varepsilon L \mu L) H_x + \nu_x (\varepsilon \mu - \varepsilon L \mu L) H_y \end{bmatrix}. \quad (2.2.15) $$
We determine the magnetic field vector $B$ in a similar fashion through

$$
\left(1 - \frac{\varepsilon \mu}{\varepsilon_L \mu L} \beta^2\right) \begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix} = \begin{bmatrix} -\nu_y(\varepsilon \mu - \varepsilon_L \mu L)E_z + \mu(1 - \varepsilon \mu \nu_x^2 - \varepsilon_L \mu L \nu_y^2)H_x - \mu \nu_x \nu_y(\varepsilon \mu - \varepsilon_L \mu L)H_y \\ \nu_x(\varepsilon \mu - \varepsilon_L \mu L)E_z - \mu \nu_x \nu_y(\varepsilon \mu - \varepsilon_L \mu L)H_x + \mu(1 - \varepsilon_L \mu L \nu_x^2 - \varepsilon \mu \nu_y^2)H_y \end{bmatrix}.
$$

Equations (2.2.15) and (2.2.16) yield the system

$$
\left(1 - \frac{\varepsilon \mu}{\varepsilon_L \mu L} \beta^2\right) \begin{bmatrix} D_z \\ B_x \\ B_y \end{bmatrix} = \begin{bmatrix} \varepsilon(1 - \beta^2) & -\nu_y(\varepsilon \mu - \varepsilon_L \mu L) & \nu_x(\varepsilon \mu - \varepsilon_L \mu L) \\ -\nu_y(\varepsilon \mu - \varepsilon_L \mu L) & \mu(1 - \varepsilon \mu \nu_x^2 - \varepsilon_L \mu L \nu_y^2) & -\mu \nu_x \nu_y(\varepsilon \mu - \varepsilon_L \mu L) \\ \nu_x(\varepsilon \mu - \varepsilon_L \mu L) & -\mu \nu_x \nu_y(\varepsilon \mu - \varepsilon_L \mu L) & \mu(1 - \varepsilon_L \mu L \nu_x^2 - \varepsilon \mu \nu_y^2) \end{bmatrix} \begin{bmatrix} E_z \\ H_x \\ H_y \end{bmatrix}.
$$

We solve (2.2.17) and find that

$$
\begin{bmatrix} E_z \\ H_x \\ H_y \end{bmatrix} = \frac{\mu(1 - \beta^2)/\alpha}{-\nu_y \overline{\omega}} \begin{bmatrix} -\nu_y \overline{\omega} \\ \nu_x \overline{\omega} \\ -\varepsilon_L \mu_L \nu_x \nu_y \overline{\omega} / \mu \end{bmatrix} \begin{bmatrix} D_z \\ B_x \\ B_y \end{bmatrix},
$$

where $\alpha = \varepsilon \mu - \varepsilon_L \mu L \beta^2$, and $\overline{\omega} = (\varepsilon \mu - \varepsilon_L \mu) / \alpha$.

We use the results for $E_z$, $H_x$, and $H_y$ (2.2.18) to rewrite the fluxes $F_{x}^{TE}$ and $F_{y}^{TE}$ in (2.2.10)

$$
F_{x}^{TE} = \begin{bmatrix} -H_y \\ 0 \\ -E_z \end{bmatrix} = \mathcal{B}^{TE} q = \begin{bmatrix} -\nu_y \overline{\omega} \\ 0 \\ -\mu(1 - \beta^2)/\alpha \end{bmatrix} \begin{bmatrix} \varepsilon_L \mu_L \nu_x \nu_y \overline{\omega} / \mu \\ 0 \\ \nu_y \overline{\omega} \end{bmatrix} \begin{bmatrix} D_z \\ B_x \\ B_y \end{bmatrix},
$$

and

$$
F_{y}^{TE} = \begin{bmatrix} H_x \\ E_z \\ 0 \end{bmatrix} = \mathcal{C}^{TE} q = \begin{bmatrix} -\nu_y \overline{\omega} \\ \mu(1 - \beta^2)/\alpha \\ 0 \end{bmatrix} \begin{bmatrix} 1 + \varepsilon_L \mu_L \nu_y \overline{\omega} / \mu \\ -\varepsilon_L \mu_L \nu_x \nu_y \overline{\omega} / \mu \\ 0 \end{bmatrix} \begin{bmatrix} D_z \\ B_x \\ B_y \end{bmatrix},
$$

where the coefficient matrices $\mathcal{B}^{TE}$ and $\mathcal{C}^{TE}$ depend on space. We write the conservation law (2.2.8) for Maxwell’s equations with TE polarization compactly in matrix form as

$$
q + \mathcal{B}^{TE} q_x + \mathcal{C}^{TE} q_y = 0.
$$

To formulate the system of conservation laws that govern the transverse magnetic (TM) polarization we take advantage of the duality present in the current and charge free Maxwell’s equations [72]. For the TM problem, we consider $\mathbf{H} = (0, 0, H_z)$ and $\mathbf{E} = (E_x, E_y, 0)$. Thus, the unknowns are $B_z$, $D_x$, and $D_y$ in the conservation law

$$
q + \nabla \cdot \mathbf{F}^{TM} = 0,
$$

(2.2.22)
where
\[
q = \begin{bmatrix} B_z \\ D_x \\ D_y \end{bmatrix}, \quad \mathbf{F}_{i}^{TM} = \begin{bmatrix} e_i \times \mathbf{E} \\ -e_i \times \mathbf{H} \end{bmatrix}.
\] 

(2.2.23)

Explicitly, we denote the vector of unknowns \( q \) and flux in the \( x \) and \( y \) direction to be
\[
q = \begin{bmatrix} B_z \\ D_x \\ D_y \end{bmatrix}, \quad \mathbf{F}_{x}^{TM} = \begin{bmatrix} E_y \\ 0 \\ H_z \end{bmatrix}, \quad \mathbf{F}_{y}^{TM} = \begin{bmatrix} -E_x \\ -H_z \\ 0 \end{bmatrix}.
\] 

(2.2.24)

We next determine how to rewrite \( E_x, E_y, \) and \( H_z \) in (2.2.24) in terms of the unknown quantities \( B_z, D_x, \) and \( D_y \). We apply the duality transformations (2.1.2) to the constitutive relation from the TE problem (2.2.18), to find
\[
\begin{bmatrix} H_z \\ E_x \\ E_y \end{bmatrix} = \begin{bmatrix} \varepsilon(1 - \beta^2)/\alpha \\ \nu_y \varpi \\ -\nu_x \varpi \end{bmatrix} \begin{bmatrix} \nu_y \varpi \(1 + \varepsilon\mu L \varpi \nu_y \varpi)/\varepsilon \end{bmatrix} \begin{bmatrix} -\nu_x \varpi \(1 + \varepsilon\mu L \varpi \nu_x \varpi)/\varepsilon \end{bmatrix} \begin{bmatrix} B_z \\ D_x \\ D_y \end{bmatrix}.
\] 

(2.2.25)

We use (2.2.25) to write the TM fluxes (2.2.24) in terms of the unknown quantities \( B_z, D_x, \) and \( D_y \)
\[
\mathbf{F}_{x}^{TM} = \begin{bmatrix} E_y \\ 0 \\ H_z \end{bmatrix} = \mathbf{B}^{TM} q = \begin{bmatrix} -\nu_x \varpi \(\varepsilon L\mu L \nu_x \nu_y \varpi)/\varepsilon \\ \varepsilon(1 - \beta^2)/\alpha \nu_y \varpi \\ -\nu_x \varpi \(1 + \varepsilon L\mu L \nu^2 \nu_y \varpi)/\varepsilon \end{bmatrix} \begin{bmatrix} B_z \\ D_x \\ D_y \end{bmatrix},
\] 

and
\[
\mathbf{F}_{y}^{TM} = \begin{bmatrix} -H_x \\ -E_z \\ 0 \end{bmatrix} = \mathbf{C}^{TM} q = \begin{bmatrix} -\nu_y \varpi \(\varepsilon L\mu L \nu_y \nu_y \varpi)/\varepsilon \\ -\nu_y \varpi \(1 + \varepsilon L\mu L \nu^2 \nu_y \varpi)/\varepsilon \\ \nu_x \varpi \(\varepsilon L\mu L \nu_x \nu_y \varpi)/\varepsilon \end{bmatrix} \begin{bmatrix} B_z \\ D_x \\ D_y \end{bmatrix},
\] 

(2.2.27)

where the coefficient matrices \( \mathbf{B}^{TM} \) and \( \mathbf{C}^{TM} \) depend on space. We write the conservation law (2.2.22) for Maxwell’s equations with TM polarization compactly in matrix form as
\[
q_t + \mathbf{B}^{TM} q_x + \mathbf{C}^{TM} q_y = 0.
\] 

(2.2.28)

### 2.2.2 Classical Wave Equation

We next derive the classical wave equation in the laboratory frame \( S \). As noted previously, the form of the wave equation changes with the frame of reference, unlike the form of Maxwell’s equations. We will see that the wave equation in the laboratory frame \( S \) includes a local drag component, which explicitly shows how the motion of the medium alters the wave speed.
We begin with the classical wave equation in the reference frame $S'$

$$\Delta' p - \frac{1}{c^2} \frac{\partial^2 p}{\partial t'^2} = 0,$$

(2.2.29)

where $c$ is the piecewise constant wave speed

$$c = \begin{cases} 
  c_L, & \text{if } x' \leq x'_0 \\
  c_R, & \text{if } x' > x'_0
\end{cases}.$$  

(2.2.30)

We pull the wave equation (2.2.29) back into the laboratory frame $S$ with the general Lorentz transformation (2.1.10). We assume that free space is to the left of the material interface, so that the wave speed $c_L$ is like the speed of light in a vacuum. Since $c = c_L$, the wave equation (2.2.29) is a Lorentz invariant [49] to the left of the material interface and

$$\Delta p - \frac{1}{c_L^2} \frac{\partial^2 p}{\partial t^2} = 0.$$  

(2.2.31)

The wave equation is not invariant with respect to the Lorentz transformation (2.1.10) on the right side of the material interface in $S'$. We add and subtract the term $c_L^{-2} \partial^2 p / \partial t'^2$ to separate the wave equation (2.2.29) into a Lorentz invariant piece and a non-Lorentz invariant piece,

$$\Delta' p - \frac{1}{c_L^2} \frac{\partial^2 p}{\partial t'^2} - \frac{1 - \bar{n}^2}{c_R^2} \frac{\partial^2 p}{\partial t'^2} = 0,$$

(2.2.32)

where $\bar{n} = c_R/c_L$. We apply the chain rule to the extra time derivatives in (2.2.32) and transform the wave equation (2.2.32) to the laboratory frame $S$ on the right of the material interface

$$c_R^2 \Delta p - \bar{n}^2 \frac{\partial^2 p}{\partial t^2} - (1 - \bar{n}^2) \gamma^2 \left( \frac{\partial}{\partial t} + \mathbf{v}_o \cdot \nabla \right)^2 p = 0, \quad x > x_0 + \mathbf{v}_o t.$$  

(2.2.33)

The wave equation in the laboratory frame $S$ within the moving medium (2.2.33) incorporates terms from the classical wave equation plus a local drag component. A direct consequence of the wave equation (2.2.33) is the formula for the Fresnel drag in a moving medium [26, 62].

We obtain the governing equation on the entire domain in $S$ from (2.2.31) and (2.2.33)

$$c^2 \Delta p - \bar{n}^2 \frac{\partial^2 p}{\partial t^2} - (1 - \bar{n}^2) \gamma^2 \left( \frac{\partial}{\partial t} + \mathbf{v}_o \cdot \nabla \right)^2 p = 0,$$

(2.2.34)

where $\bar{n}$ is defined as the piecewise constant

$$\bar{n} = \frac{c_R}{c_L}, \quad c = \begin{cases} 
  c_L, & \text{if } x \leq x_0 + \mathbf{v}_o t \\
  c_R, & \text{if } x > x_0 + \mathbf{v}_o t
\end{cases}.$$  

(2.2.35)
We define the piecewise constant $n = 1/n$ as the reciprocal of the index of refraction. This definition of $n$ allows us to take $c_R = 0$ without issues of dividing by zero.

We use the constitutive relations [56, 78]

$$u_t = -p_x,$$
$$v_t = -p_y,$$  \hfill (2.2.36)

to convert the wave equation in the laboratory frame (2.2.34) as a first order system of conservation laws. If we assume that the order of mixed partial derivatives does not matter, then (2.2.36) lets us convert the wave equation (2.2.34) into the conservation law

$$\begin{bmatrix}
    p \\
    u \\
    v
\end{bmatrix}_t + \begin{bmatrix}
    2ab\nu_x & b(c^2 - a\nu_x^2) & -ab\nu_x\nu_y \\
    1 & 0 & 0 \\
    0 & 0 & 0
\end{bmatrix}\begin{bmatrix}
    p \\
    u \\
    v
\end{bmatrix}_x + \begin{bmatrix}
    2ab\nu_y & -ab\nu_x\nu_y & b(c^2 - a\nu_y^2) \\
    0 & 0 & 0 \\
    1 & 0 & 0
\end{bmatrix}\begin{bmatrix}
    p \\
    u \\
    v
\end{bmatrix}_y = 0, \hfill (2.2.37)
$$

where $v_o = (\nu_x, \nu_y)$ is the velocity of the moving medium, $a = \gamma^2(1 - n^2)$, and $b = (n^2 + a)^{-1}$. We write the conservation law form of the wave equation (2.2.37) in matrix form

$$\mathbf{q}_t + \nabla \cdot \mathbf{F}^W = \mathbf{q}_t + \mathbf{B}^W \mathbf{q}_x + \mathbf{C}^W \mathbf{q}_y = 0,$$  \hfill (2.2.38)

where $\mathbf{B}^W$ is the coefficient matrix in the $x$ direction and $\mathbf{C}^W$ is the coefficient matrix in the $y$ direction.
CHAPTER 3

THE PHYSICS I: ANALYTICAL SOLUTIONS

This chapter describes, in detail, several analytical solutions of the electromagnetic and acoustic wave reflection and transmission problems from a moving material interface.

An analytical solution is indispensable to verify the convergence rate and accuracy of a numerical approximation, as we will see in Chap. 8. There are known analytical solutions for the plane wave reflection and transmission for a static interface for electromagnetics [72] and acoustics [56] as well as from a moving plane dielectric interface available, but they are spread across several sources [21, 46, 48, 72, 84, 98]. To collect and describe the reference solutions for scattering from a moving interface we:

1. Derive the phase matching condition for the TE and TM polarization and the wave equation in Sec. 3.2, since they satisfy the same boundary conditions at a non-conducting interface [49, 72].

2. Rotate the phase angles to a problem where the moving interface is oblique to the plane of incidence in Sec. 3.2.1.

3. Derive the reflection and transmission coefficients for the TE problem in Sec. 3.3.1, the TM problem in Sec. 3.3.2, and the wave equation in Sec. 3.3.3.

4. Recover the other TE electromagnetic field vectors from the electric field $E_z$ in Sec. 3.4.

3.1 Plane Wave Solutions

A plane TE wave analytical solution has the form

$$ q = \begin{bmatrix} E_z \\ B_x \\ B_y \end{bmatrix} = a \psi(\mathbf{k} \cdot \mathbf{x} - \omega t) \begin{bmatrix} 1 \\ \frac{k_y}{\omega} \\ \frac{-k_x}{\omega} \end{bmatrix}, \quad (3.1.1) $$

for wave reflection and transmission from a constantly moving plane material interface with velocity purely in the $x$ direction, $\mathbf{v}_o = (v_x, 0)$. In (3.1.1), $\psi(\xi)$ is a smooth waveform with maximum value one, $a$ is an amplitude, $\mathbf{k}$ is the wavevector, and $\omega$ is the frequency. We obtain the TM plane wave
solution from the principle of duality. The plane wave solution for the classical wave equation takes a similar form,

\[ \mathbf{q} = \begin{bmatrix} p \\ u \\ v \end{bmatrix} = \alpha \psi (k \cdot x - \omega t) \begin{bmatrix} 1 \\ \frac{k_x}{\omega} \\ \frac{k_y}{\omega} \end{bmatrix}. \] 

(3.1.2)

### 3.2 Phase Angles for a Vertical Moving Interface

To analyze the phase matching at a vertical moving material interface we use a standard procedure from electromagnetic theory, first proposed by Einstein [28]:

1. Start with a propagating wave in the laboratory frame.
2. Transform into a reference frame where the material interface appears at rest.
3. Compute the phase angle for the static interface problem.
4. Transform the results back to the initial laboratory frame.

We illustrate the procedure in Fig. 3.1.

![Figure 3.1: Oblique reflection from a moving boundary normal to the x'-axis.](image)

Figure 3.1: Oblique reflection from a moving boundary normal to the x'-axis.
The propagation phases are a Lorentz invariant in the two frames of reference, a direct consequence of the frequency-wavenumber four-vector (2.1.13). We solve the discontinuous problem in $S'$ in the usual way [49]. To specify any one of the incident, reflected, or transmitted waves, we replace $k'$ with the appropriate wavevector and $a'$ with the appropriate amplitude in (3.1.1) or (3.1.2). Snell’s Law [72], which states that for angle of incidence $\theta'_i$, angle of reflection $\theta'_r$, angle of transmission $\theta'_T$, reflected wavevector $k'_r$, and transmitted wavevector $k'_T$, matches the phases in the reference frame $S'$,

$$|k'_i| \sin \theta'_i = |k'_r| \sin \theta'_r = |k'_T| \sin \theta'_T.$$  \hfill (3.2.1)

We take the incident wavevector to be

$$k'_i = \frac{\omega'}{c_L} (\cos \theta'_i \hat{x} + \sin \theta'_i \hat{y}) = \frac{\omega'}{c_L} \left( k'_{ix} \hat{x} + k'_{iy} \hat{y} \right),$$  \hfill (3.2.2)

where $(k'_{ix})^2 + (k'_{iy})^2 = 1$. From (3.2.1) and the law of reflection, $\theta'_i = \theta'_r$, we find that

$$k'_r = \frac{\omega'}{c_L} (\cos \theta'_r \hat{x} + \sin \theta'_r \hat{y}) = \frac{\omega'}{c_L} \left( -k'_{ix} \hat{x} + k'_{iy} \hat{y} \right),$$

$$k'_T = \frac{\omega'}{c_R} (\cos \theta'_T \hat{x} + \sin \theta'_T \hat{y}) = \frac{\omega'}{c_R} \left( \sqrt{1 - \left( \frac{k'_{iy}}{n} \right)^2} \hat{x} + \frac{k'_{iy}}{n} \hat{y} \right),$$  \hfill (3.2.3)

where $n = c_L/c_R$ is the index of refraction.

We need the angles of reflection and transmission plus Doppler shifted frequencies to specify the phase angle in the laboratory frame. We apply the $x$-directed inverse Lorentz transformation on the frequency-wavenumber four-vector (2.1.12) with $k_z = k'_z = 0$ to pull the wavevectors back to $S$ and obtain six equations for the frequencies and $x$-components of the incident, reflected and transmitted waves, for complete details see [72].

The frequency dependent generalization of Snell’s law states that [72]

$$\omega_i \sin \theta_i = \omega_r \sin \theta_r = \omega_T n_T \sin \theta_T = \omega' \sin \theta'_i = \omega' \sin \theta'_r = n \omega' \sin \theta'_T,$$  \hfill (3.2.4)

where $n_T$ is the effective refractive index in the moving medium. We call the phase velocity in the moving material $\vartheta_T \equiv \omega_T/k_T = c_L/n_T$. From the generalized Snell’s Law (3.2.4), the incident, reflected and transmitted waves have the same phase in either frame of reference.

We collect the components of the reflected and transmitted plane waves written in the laboratory frame. The reflected angle $\theta_r$ and reflected frequency $\omega_r$ are [28, 72]

$$\cos \theta_r = \frac{(1 + \beta^2) \cos \theta_i - 2 \beta}{1 - 2 \beta \cos \theta_i + \beta^2},$$  \hfill (3.2.5)
and
\[ \omega_r = \omega_i \left( \frac{1 - 2\beta \cos \theta_i + \beta^2}{1 - \beta^2} \right). \]  

(3.2.6)

Note that it is no longer true that \( \theta_i = \theta_r \) unless \( \beta = \nu_x/c_L = 0 \). The transmitted angle, \( \theta_T \), and frequency, \( \omega_T \), in \( S \) are
\[ \cos \theta_T = \frac{Q + \beta(1 - \beta \cos \theta_i)}{n_T \{ \beta Q + 1 - \beta \cos \theta_i \}}, \]  

(3.2.7)

and
\[ \omega_T = \omega_i \left( \frac{1 - \beta \cos \theta_i + \beta Q}{1 - \beta^2} \right), \]  

(3.2.8)

where
\[ Q = \sqrt{(1 - n^2)(1 - \beta \cos \theta_i)^2 + n^2(\cos \theta_i - \beta)^2}. \]  

(3.2.9)

We also find the effective refractive index, \( n_T \), in the frame to be
\[ n_T = \frac{\sqrt{\gamma^2 \{ Q + \beta(1 - \beta \cos \theta_i) \}^2 + (1 - \beta^2) \sin^2 \theta_i}}{\gamma \{ 1 - \beta \cos \theta_i + \beta Q \}}. \]  

(3.2.10)

We review how to specify the phase angle for the incident, reflected and transmitted waves in the laboratory frame for the vertical moving interface. The incident wave has the known frequency \( \omega_i \), angle \( \theta_i \), and wavenumber \( |k_i| = \omega_i/c_L \). For the reflected wave one uses the frequency (3.2.6), reflection angle (3.2.5), and wavenumber \( |k_r| = \omega_r/c_L \). The phase of the transmitted wave is defined by the frequency (3.2.8), transmission angle (3.2.7), and the wavenumber \( |k_T| = \omega_T n_T/c_L \) with the effective refractive index (3.2.10).

### 3.2.1 Oblique Phase Angles

Next we find the phase angle for reflection and transmission when the plane material interface moves with velocity \( \mathbf{v}_o \) oblique to the \( x \)-axis with normal \( \hat{n} \) at some angle \( \theta \). In the reference frame \( S' \) the interface appears to be at rest and the interface is still vertical. Therefore, we can once more solve the problem by the standard stationary method. The phases of the incident, reflected and transmitted waves still satisfy the invariance principle at the rotated interface [72].

We apply a rotation after we apply the Lorentz transformation to relate the frequency-wavenumber and space-time quantities from \( S' \) to \( S \) [48]. That is,
\[ b' = L(N)R(-\theta)b \quad \text{and} \quad b = R(\theta)L(-N)b', \]  

(3.2.11)
where \( \mathbf{b} \) is a four-vector in \( S \), \( \mathbf{b}' \) is a four-vector in \( S' \), \( \hat{n} = (n_x, n_y) = (\cos \theta, \sin \theta) \), \( L(\mathbf{N}) \) is given in (2.1.11), and

\[
R(\theta) = \begin{bmatrix}
1 & 0 & 0 \\
0 & n_x & -n_y \\
0 & n_y & n_x
\end{bmatrix}.
\]

(3.2.12)

Note that \( R(\theta)R(-\theta) = R(-\theta)R(\theta) = \mathbf{I} \) and from the odd/even properties of sine/cosine we know that, in two dimensions, \( R^T(\theta) = R(-\theta) \).

We introduce the incident frequency-wavenumber vector \( \mathbf{k}_i = [\omega_i \; k_{ix} \; k_{iy}]^T \), the reflected frequency-wavenumber vector \( \mathbf{k}_r = [\omega_r \; k_{rx} \; k_{ry}]^T \), and the transmitted frequency-wavenumber vector \( \mathbf{k}_T = [\omega_T \; k_{Tx} \; k_{Ty}]^T \). For the reflected wave,

\[
\mathbf{k}_i = R(\theta)L(-\mathbf{N})\mathbf{k}'_i \quad \text{and} \quad \mathbf{k}_r = R(\theta)L(-\mathbf{N})\mathbf{k}'_r.
\]

(3.2.13)

But

\[
\mathbf{k}'_r = G\mathbf{k}'_i,
\]

(3.2.14)

where

\[
G = \begin{bmatrix}
1 & 0 & 0 \\
0 & -1 & 0 \\
0 & 0 & 1
\end{bmatrix}.
\]

(3.2.15)

Thus,

\[
\mathbf{k}_r = R(\theta)L(-\mathbf{N})GL(\mathbf{N})R(-\theta)\mathbf{k}_i.
\]

(3.2.16)

Similarly, for the transmitted wave

\[
\mathbf{k}_i = R(\theta)L(-\mathbf{N})\mathbf{k}'_i \quad \text{and} \quad \mathbf{k}_T = R(\theta)L(-\mathbf{N})\mathbf{k}'_T.
\]

(3.2.17)

But

\[
\mathbf{k}'_T = D\mathbf{k}'_i,
\]

(3.2.18)

where

\[
D\mathbf{k}'_i = \begin{bmatrix}
\omega' \\
\sqrt{\omega'c_R^{-2} - (k_{iy}')^2} \\
k_{iy}'
\end{bmatrix}.
\]

(3.2.19)

Therefore,

\[
\mathbf{k}_T = R(\theta)L(-\mathbf{N})DL(\mathbf{N})R(-\theta)\mathbf{k}_i.
\]

(3.2.20)

We compute the effective refractive index from the wavenumber of the transmitted wave, \( n_T = c_Lk_T/\omega_T \).
3.3 Reflection and Transmission Coefficients for a Moving Plane Interface

To compute the reflection and transmission coefficients for the electromagnetic wave scattering problem we use the Fresnel equations [72] for the TE polarization and find the coefficients for the TM polarization from the duality transformations (2.1.2). We derive the reflection and transmission coefficients for the classical wave equation from the Rankine-Hugoniot jumps conditions [90]. Again, we find the coefficients for the vertical interface first, then rotate the result to the oblique problem.

3.3.1 Reflection and Transmission Coefficients for a Transverse Electric Plane Wave

Just as was done with the phase angles, we find the reflection and transmission coefficients in the reference frame $S'$ and then transform the results to $S$. The Fresnel equations in the reference frame $S'$ for the TE polarization [72] give the reflection and transmission ratios

$$\rho_{TE} = \frac{a'_r}{a'_i} = \frac{\cos \theta'_i - n \cos \theta'_T}{\cos \theta'_i + n \cos \theta'_T}, \quad \tau_{TE} = \frac{a'_T}{a'_i} = 1 + \rho_{TE}. \quad (3.3.1)$$

Eq. (3.3.1) in $S$ becomes [72]

$$\rho_{TE} = \frac{\cos \theta_i - \beta - Q}{\cos \theta_i - \beta + Q}, \quad \tau_{TE} = 1 + \rho_{TE}. \quad (3.3.2)$$

We now derive the reflected and transmitted fields in the frame $S$ from those in $S'$. The simplest way is to apply a six-vector transformation for the electric field (2.1.14) separately to the incident, reflected, and transmitted waves. In the reference frame, a plane wave propagating along the unit vector $k'$ has magnetic field

$$cB' = c\mu L H' = nk' \times E', \quad (3.3.3)$$

where $n = 1$ for the incident and reflected waves. Since the wave is TE, and the motion is along the $z$–direction, the electric field is perpendicular to the velocity, that is, $\beta \cdot E' = 0$. Then

$$E = E_\perp = \gamma (E'_\perp - \beta \times cB'_\perp) = \gamma (E' - \beta \times (nk' \times E'))$$

$$= \gamma (E' - n(\beta \cdot E')k' + n(\beta \cdot k')E')$$

$$= \gamma E'(1 + n\beta \cdot k'). \quad (3.3.4)$$
We apply the result (3.3.4) to the incident, reflected, and transmitted fields, to find the electric fields

\[ E_i = \gamma E'_i (1 + \beta \cos \theta'_i) \]
\[ E_r = \gamma E'_r (1 - \beta \cos \theta'_i) = \gamma \rho^{TE} E'_i (1 - \beta \cos \theta'_i) \]
\[ E_T = \gamma E'_T (1 + n \beta \cos \theta'_T) = \gamma \tau^{TE} E'_i (1 + n \beta \cos \theta'_T) \]

(3.3.5)

From (3.3.5) we find the TE reflection and transmission coefficients in \( S \) to be

\[ \frac{E_r}{E_i} = \rho^{TE} \frac{1 - \beta \cos \theta'_i}{1 + \beta \cos \theta'_i}, \quad \frac{E_T}{E_i} = \tau^{TE} \frac{1 + n \beta \cos \theta'_T}{1 + \beta \cos \theta'_i} = \gamma \tau^{TE} \frac{\omega_T}{\omega_i}. \]

(3.3.6)

We note that a TE perfect mirror corresponds to \( \rho^{TE} = -1 \) and \( \tau^{TE} = 0 \).

We can also generalize the reflected and transmitted fields (3.3.6) to an oblique moving interface. We rotate the interface to have the normal \( \hat{n} = (n_x, n_y) \) if we replace \( \cos \theta_i \) in \( \rho^{TE}, \tau^{TE} \), and \( Q \) by the quantity

\[ \cos \theta_i \rightarrow \cos \hat{\theta}_i = n_x \cos \hat{\theta}_i + n_y \sin \hat{\theta}_i, \]

(3.3.7)

where \( \cos \hat{\theta}_i \) and \( \sin \hat{\theta}_i \) are the normalized wavevector components in the rotated frame in which the moving interface appears vertical. We use the rotated frequencies \( \omega_r \) and \( \omega_T \) in (3.3.6) as well.

### 3.3.2 Reflection and Transmission Coefficients for a Transverse Magnetic Plane Wave

We can immediately determine the incident, reflected and transmitted fields for a TM wave from the principle of duality (2.1.2). From (3.3.2) the TM Fresnel equations are

\[ \rho^{TM} = \frac{n (\cos \theta_i - \beta) - Q/n}{n (\cos \theta_i - \beta) + Q/n}, \quad \tau^{TM} = 1 + \rho^{TM}, \]

(3.3.8)

and the TM reflection and transmission coefficients in \( S \) are

\[ \frac{H_r}{H_i} = \rho^{TM} \frac{\omega_x}{\omega_i}, \quad \frac{H_T}{H_i} = \tau^{TM} \frac{\omega_T}{\omega_i}. \]

(3.3.9)

We note that a TM perfect mirror corresponds to \( \rho^{TM} = 1 \) and \( \tau^{TM} = 0 \).

The rotation of the plane interface alters the phase angle in the same way regardless of polarization, therefore we rotate the reflected and transmitted fields (3.3.9) in the same way as in Sec. 3.3.1.
3.3.3 Reflection and Transmission Coefficients for the Classical Wave Equation

To find the reflection and transmission coefficients for the classical wave equation, we ensure that the incident, reflected and transmitted waves satisfy the correct jump conditions at the moving interface. The jumps satisfy the Rankine-Hugoniot condition, which states that the normal flux across the moving interface must be continuous [90].

For the wave equation in $S$, (2.2.37), we know that the normal flux is
\[
A = \hat{n} \cdot F^W = B^W n_x + C^W n_y = \begin{bmatrix} 2abN n_x b(c^2 - av_x^2) & -av_x \nu_y n_y b(c^2 - av_y^2) \\ n_x & 0 \\ n_y & 0 \end{bmatrix}.
\] (3.3.10)

The Rankine-Hugoniot condition is
\[
A_L q_L - N q_L = A_R q_R - N q_R,
\] (3.3.11)

where $A$ is the normal flux matrix (3.3.10).

The normal flux matrix $A$ (3.3.10) simplifies considerably for a vertical moving plane material interface that moves only in the $x-$direction. For that situation, we use $\hat{n} = (n_x, n_y) = (1, 0)$, $v_o = (\nu_x, 0)$, and simplify (3.3.11) to relate the left and right states by
\[
\begin{bmatrix} -N & c_L^2 & 0 \\ 1 & -N & 0 \\ 0 & 0 & -N \end{bmatrix} \begin{bmatrix} p_L \\ u_L \\ v_L \end{bmatrix} = \begin{bmatrix} N(2ab - 1) & b(c_R^2 - av_x^2) & 0 \\ 1 & -N & 0 \\ 0 & 0 & -N \end{bmatrix} \begin{bmatrix} p_R \\ u_R \\ v_R \end{bmatrix}.
\] (3.3.12)

The reflection and transmission coefficients from (3.3.12) require relationships between the left and right states, for full details see [97]. Since the incident, reflected, and transmitted waves are in phase, we use the general form of the plane wave (3.1.2) to find that
\[
u_L = \frac{k_x}{c_L} p_L, \quad v_L = \frac{k_y}{c_L} p_L, \quad u_R = \frac{k_x n_T}{c_L} p_R, \quad v_R = \frac{k_y n_T}{c_L} p_R.
\] (3.3.13)

where $k_x$ and $k_y$ are the appropriate components of the wavevector and $k_x^2 + k_y^2 = 1$. We substitute the plane wave relations (3.3.13) into the Rankine-Hugoniot condition (3.3.12) to find
\[
\begin{bmatrix} -N & k_x c_L & 0 \\ 1 & -N k_x/c_L & 0 \\ 0 & 0 & -N k_y/c_L \end{bmatrix} \begin{bmatrix} p_L \\ p_L \\ p_L \end{bmatrix} = \begin{bmatrix} N(2ab - 1) & k_x n_T b(c_R^2 - av_x^2)/c_L & 0 \\ 1 & -N k_x n_T/c_L & 0 \\ 0 & 0 & -N k_y n_T/c_L \end{bmatrix} \begin{bmatrix} p_R \\ p_R \\ p_R \end{bmatrix}.
\] (3.3.14)
The state to the left of the material interface, \( p_L \), is a linear combination of the incident and reflected waves. The state to the right of the material interface, \( p_R \), is the transmitted wave. So,

\[
p_L = p_i + p_r, \quad \text{and} \quad p_R = p_T. \tag{3.3.15}
\]

To obtain a system of equations for the unknown amplitudes we use (3.3.15) and the fact that the incident, reflected and transmitted waves all have the same phase. The third equation in (3.3.14) corresponds to tangential components. Physically, we know that the wave scattering problem is translation invariant in the vertical direction. Thus, the tangential components make no contribution to the amplitudes and we omit the last equation in (3.3.14). We then solve the remaining 2 \( \times \) 2 system to find the amplitudes of the reflected and transmitted waves from a vertical moving plane material interface to be

\[
a_r = \frac{-N^2(dk_{ix} + n_T k_{Tx}) + 3k_{Tx}(Nk_{ix} - c_L) - Nc_L(d + 1) + c_L k_{ix}(1 - Nn_T k_{Tx})}{N^2(dk_{ix} + n_T k_{Tx}) + 3k_{Tx}(Nk_{ix} - c_L) - Nc_L(d + 1) + c_L k_{ix}(1 - Nn_T k_{Tx})}, \tag{3.3.16}
\]

and

\[
a_T = \frac{(N^2 - c_L^2)(k_{ix} - k_{rx})}{N^2(dk_{ix} + n_T k_{Tx}) + 3k_{Tx}(Nk_{ix} - c_L) - Nc_L(d + 1) + c_L k_{ix}(1 - Nn_T k_{Tx})}, \tag{3.3.17}
\]

where \( d = 2ab - 1 \) and \( \zeta = n_T b(c_R^2 - a\nu^2)/c_L \).

We generalize the reflection (3.3.16) and transmission (3.3.17) coefficients from a vertical to an oblique moving interface. To do so, we rotate the problem to an interface with normal \( \hat{n} = (n_x, n_y) \). For example, the incident wave vector rotates to

\[
\vec{k}_i = \frac{\omega_i}{c_L}(\hat{k}_{ix}\hat{x} + \hat{k}_{iy}\hat{y}), \tag{3.3.18}
\]

where

\[
\begin{bmatrix}
\hat{k}_{ix} \\
\hat{k}_{iy}
\end{bmatrix}
= \begin{bmatrix}
   n_x & n_y \\
   -n_y & n_x
\end{bmatrix}
\begin{bmatrix}
\hat{k}_{ix} \\
\hat{k}_{iy}
\end{bmatrix}, \tag{3.3.19}
\]

\( \hat{k}_{ix} \) and \( \hat{k}_{iy} \) are the wavevector components in the rotated frame where the interface appears vertical, and \( \hat{k}_{ix}^2 + \hat{k}_{iy}^2 = 1 \). Thus, the amplitudes for the oblique interface are

\[
a_r = \frac{-N^2(\delta(\vec{k}, \hat{n}) + n_T(\vec{k}_T \cdot \hat{n})) + (\nu x \hat{k}_{ix} + \nu y \hat{k}_{iy})(N(\vec{k}, \hat{n}) - c_L) - Nc_L(d + 1) + c_L(\vec{k}, \hat{n})(1 - Nn_T(\vec{k}_T \cdot \hat{n}))}{N^2(\delta(\vec{k}, \hat{n}) + n_T(\vec{k}_T \cdot \hat{n})) + (\nu x \hat{k}_{ix} + \nu y \hat{k}_{iy})(N(\vec{k}, \hat{n}) - c_L) - Nc_L(d + 1) + c_L(\vec{k}, \hat{n})(1 - Nn_T(\vec{k}_T \cdot \hat{n}))}, \tag{3.3.20}
\]

and

\[
a_T = \frac{(N^2 - c_L^2)(\vec{k}_i \cdot \hat{n} - \vec{k}_T \cdot \hat{n})}{N^2(\delta(\vec{k}, \hat{n}) + n_T(\vec{k}_T \cdot \hat{n})) + (\nu x \hat{k}_{ix} + \nu y \hat{k}_{iy})(N(\vec{k}, \hat{n}) - c_L) - Nc_L(d + 1) + c_L(\vec{k}, \hat{n})(1 - Nn_T(\vec{k}_T \cdot \hat{n}))}, \tag{3.3.21}
\]

where \( \eta = n_T b c_R^2/c_L \).
3.4 Recovering Electric and Magnetic Fields in a Moving Dielectric

The analytical solution for reflection and transmission of a TE plane wave from a constantly moving plane dielectric interface was written in terms of the electric field $E_z$. From that electric field we can recover any of other fields, $B$, $D$, or $H$, from the constitutive relations (2.2.6) and (2.2.7). The reflected wave components are simple to formulate since they propagate in free space, and the constitutive relations simplify to those in a static media. However, in the moving dielectric medium the expressions are more complicated.

The recovered expressions for the other electromagnetic fields in the moving dielectric medium, particularly $D$ and $H$, simplify considerably [21] when the motion is in the $x-$direction only, i.e., $v_o = (\nu_x, 0)$. The plane-wave solution in the moving dielectric is

$$
E_{Tz} = \tau^{TE} \frac{\omega_T}{\omega_i} \psi(k_T \cdot x - \omega_T t),
$$

$$
B_{Tx} = \frac{k_{Ty}}{\omega_T} E_{Tz},
$$

$$
B_{Ty} = -\frac{k_{Tx}}{\omega_T} E_{Tz},
$$

where, again, $\psi$ is a smooth waveform with maximum value 1. The constitutive relations (2.2.5) allow us to write all the remaining field expressions in the moving dielectric medium as

$$
D_{Tz} = \frac{1}{c^2 \mu L} n^2 \frac{(1 - \beta \cos \theta_i) + \beta Q}{1 - N \cos \theta_i + \beta Q} E_{Tz},
$$

$$
H_{Tx} = \frac{1}{c \mu L} \frac{(1 - \beta^2) \sin \theta_i}{1 - \beta \cos \theta_i + \beta Q} E_{Tz},
$$

$$
H_{Ty} = \frac{1}{c \mu L} \frac{\beta n^2 (1 - \beta \cos \theta_i) - Q}{1 - \beta \cos \theta_i + \beta Q} E_{Tz}.
$$

To obtain any of the TM fields, one applies the duality transformations (2.1.2) to (3.4.1) and (3.4.2).

To generalize the transmitted electric and magnetic fields (3.4.1) and (3.4.2) to an oblique moving interface we use the rotated wavevector $\kappa_T$ (3.2.20), $\beta \rightarrow N/c_L$, and substitute (3.3.7) and

$$
\sin \theta_i \rightarrow \sin \tilde{\theta}_i = -n_y \cos \hat{\theta}_i + n_x \sin \hat{\theta}_i,
$$

in $\tau^{TE}$, $Q$, and (3.4.2).
3.5 Special Cases of Electromagnetic Wave Reflection and Transmission from a Static Dielectric Interface

There are three additional interesting phenomena of wave reflection and transmission from a static dielectric interface, even though the focus of this dissertation is predominantly the mathematical description of waves at moving interfaces. These phenomena are Brewster’s angle, where there is no reflected wave, total internal reflection (TIR), where all wave energy is reflected back into a medium with no transmission, and frustrated total internal reflection (fTIR), where some energy in a wave jumps a small material gap. These problems can also be solved effectively with the DGSEM developed here.

To consider these problems, we examine wave propagation from a denser medium, with index of refraction \( n_1 \), to lighter medium, with index of refraction \( n_2 \), where \( n_1 > n_2 \), e.g., a wave propagating from glass to air. In such a situation, TIR can occur [49, 72]. The phenomenon of Brewster’s angle can be observed in dense-to-light or light-to-dense propagation problems [49, 72]. We will derive the angle of incidence \( \theta_i \) for which either phenomenon occur from Snell’s law

\[
n_1 \sin(\theta_i) = n_2 \sin(\theta_T),
\]

where \( \theta_T \) is the propagation angle of the transmitted wave. To simplify the derivations we note the static interface Fresnel coefficients [49] for the TE polarization

\[
\rho_{TE} = \frac{n_1 \cos(\theta_i) - n_2 \cos(\theta_T)}{n_1 \cos(\theta_i) + n_2 \cos(\theta_T)},
\]

\[
\tau_{TE} = \frac{2n_1 \cos(\theta_i)}{n_1 \cos(\theta_i) + n_2 \cos(\theta_T)},
\]

and the TM polarization

\[
\rho_{TM} = \frac{n_1 \cos(\theta_T) - n_2 \cos(\theta_i)}{n_1 \cos(\theta_T) + n_2 \cos(\theta_i)},
\]

\[
\tau_{TM} = \frac{2n_1 \cos(\theta_T)}{n_1 \cos(\theta_T) + n_2 \cos(\theta_i)}.
\]

As a quantitative example we provide a plot of the TE and TM reflection coefficient in Fig. 3.2 for \( n_1 = 1.5 \) and \( n_2 = 1 \) that is typical for a glass to air interface where Brewster’s angle and the region of total internal reflection are immediately seen.
Angle of Incidence $\theta$

Figure 3.2: The reflection coefficient for TE and TM polarization as a function of the angle of incidence. The special cases of Brewster’s angle and TIR are labeled.

### 3.5.1 Brewster’s Angle

Brewster’s angle is the angle of incidence, $\theta_B$, at which the TM Fresnel reflection coefficient (3.5.4) vanishes. The TE coefficient (3.5.2) cannot vanish for any angle $\theta$ for non-magnetic materials [72]. The Brewster angle is sometimes called the polarizing angle because if a mixture of TE and TM waves are incident on a dielectric interface at angle $\theta_B$, only the TE or perpendicularly polarized waves will be reflected.

We calculate Brewster’s angle, when $\theta_i = \theta_B$, from the geometric condition $\theta_i + \theta_T = \frac{\pi}{2}$ [72]. Then in Snell’s law (3.5.1) we have

$$n_1 \sin(\theta_B) = n_2 \sin(\theta_T) = n_2 \sin \left( \frac{\pi}{2} - \theta_B \right) = n_2 \cos(\theta_B). \tag{3.5.6}$$

Solving for $\theta_B$ we find

$$\theta_B = \tan^{-1} \left( \frac{n_2}{n_1} \right). \tag{3.5.7}$$
3.5.2 Total Internal Reflection

If the incident wave is from a denser to a lighter medium it is possible to exhibit the TIR phenomenon. We determine the critical angle from Snell’s law when \( \theta_C = \frac{\pi}{2} \), i.e.,

\[
\sin(\theta_C) = \frac{n_2}{n_1},
\]

(3.5.8)

If \( \theta_i \leq \theta_C \), there is normal refraction into the lighter medium. But, if \( \theta_i > \theta_C \) the incident wave cannot be refracted and gets completely reflected back into the denser medium. Beyond the critical angle Snell’s law (3.5.1) still holds, except that \( \sin(\theta_T) \) becomes greater than one, and \( \cos(\theta_T) \) becomes imaginary, i.e.,

\[
\sin(\theta_T) = \frac{n_1}{n_2} \sin(\theta_i),
\]

\[
\cos(\theta_T) = j \sqrt{\sin^2(\theta_T) - 1},
\]

(3.5.9)

where \( j = \sqrt{-1} \). The Fresnel reflection and transmission formulae (3.5.2)–(3.5.5) become imaginary as well.

Next, we examine the components of the transmitted wavevector with \( x \) and \( y \) components

\[
k_{Tx} = \omega n_2 \sin(\theta_T),
\]

\[
k_{Ty} = \omega n_1 \cos(\theta_T) = \pm j \omega n_1 \sqrt{\sin^2(\theta_T) - 1} = \pm j \alpha,
\]

(3.5.10)

where the sign of the square root is chosen for the solution to remain physical. The amplitude of the transmitted wave decays exponentially, for example in positive \( y \) as \( e^{-\alpha y} \). It is important to note there is a finite penetration of the wave into the transmitted medium. The wave does not stop abruptly at the interface [72]. The transmitted wave is no longer a plane wave but an evanescent wave. The penetration depth of the evanescent wave is given by \( 1/\alpha \). Although an evanescent wave penetrates into the transmitted medium, there is no net energy transport across the boundary, which can be shown directly as the Poynting vector of the transmitted wave is zero [72].

The reflected wave has the same amplitude as the incident wave, but there is an angle-dependent phase shift. To fully describe the TIR wave we must discuss the complex valued Fresnel coefficients. We examine the TE reflection coefficient (3.5.2) in depth, but a similar analysis can be performed on the other Fresnel coefficients. This analysis is simplest if we view \( \rho^{TE} \) in complex polar form. It is straightforward to show that the magnitude of the reflection coefficient (3.5.2) is

\[
|\rho^{TE}| = \rho^{TE}(\rho^{TE})^* = 1,
\]

(3.5.11)
and the argument of the complex coefficient $\rho^{TE}$ is

$$\phi_{TE} = -2 \tan^{-1} \left( \frac{n_2 \sqrt{\sin^2(\theta_T) - 1}}{n_1 \cos(\theta_i)} \right).$$  \hspace{1cm} (3.5.12)$$

It is surprising that the argument is non-zero. This shows that the incident and reflected waves are no longer in phase. The small phase shift present in the wave undergoing total internal reflection is known as the Goos-Hänchen effect [38, 72]. The phase shift differs for TE and TM waves [72].

**Frustrated Total Internal Reflection.** The phenomenon of frustrated total internal reflection (fTIR), sometimes called evanescent wave coupling, occurs when an evanescent wave penetrates across a separating medium of low refractive index into a region occupied by a higher refractive index, as sketched in Fig. 3.3 for $n_1, n_3 > n_2$. It has many practical applications like touch screens [10, 41] and fingerprint scanners [61]. Frustrated total internal reflection is similar to quantum tunneling and mathematically analogous if one thinks of the electromagnetic field as the wave function of the photon [10].

![Figure 3.3: Wave propagation through a thin film of low refractive index, $n_2$, between two high-index semi-infinite media, $n_1$ and $n_3$.](image)

The name is a bit of a misnomer as in the case of fTIR because the internal reflection is no longer total as energy may flow across a material gap [72]. The gap between the two high-index materials, $d$, must be smaller than the decay length of the evanescent wave, $1/\alpha$, for a transmitted wave to be generated. Heuristically we know that the transmission coefficient depends on the separation distance between the two media with higher refractive index. As the gap shrinks, $d \to 0$, we recover the typical transmission coefficient between two materials with indexes $n_1$ and $n_3$. However, as the gap widens so that $d > 1/\alpha$ there is no transmission and we return to the TIR regime.
3.6 Physical Properties of Acoustic Wave Scattering from a Moving Interface

We derived the analytical solution of the reflection and transmission of an acoustic plane wave from a moving material interface in Secs. 3.2 and 3.3.3. Next we examine some physical properties of the solution, e.g., we observe that waves that are not dispersive in the reference frame become spatially dispersive in the laboratory frame.

3.6.1 Limiting Cases

The solution to the wave equation in the moving material should become the solution to the static reflection and transmission problem as $\nu_x \to 0$ [56]. We consider the solution for the oblique moving material interface since it is more general. When $\nu_x = 0$, the operator $L(N)$ (2.1.11) becomes the identity matrix, i.e., $L(0) = I$. Thus, all the frequencies are equal, $\omega_i = \omega_r = \omega_T$, i.e., there is no Doppler effect for the static problem. We find the $x-$components of the reflected and transmitted waves to be

$$k_{rx} = -k_{ix} \quad \text{and} \quad k_{Tx} = \omega_i \sqrt{\frac{1}{c_R^2} - \frac{k_{ix}^2}{c_L^2}}. \quad (3.6.1)$$

Note that we recover the law of reflection, $\theta_i = \theta_r$, too. Finally, the reflection and transmission coefficients become

$$\frac{a_r}{a_i} = \frac{c_R(k_T \cdot \hat{n}) - c_L(k_i \cdot \hat{n})}{c_L(k_r \cdot \hat{n}) - c_R(k_T \cdot \hat{n})}, \quad (3.6.2)$$

$$\frac{a_T}{a_i} = \frac{c_L(k_r \cdot \hat{n}) - c_L(k_i \cdot \hat{n})}{c_L(k_r \cdot \hat{n}) - c_R(k_T \cdot \hat{n})}. \quad (3.6.3)$$

Physical intuition also dictates that as the wave speed in the moving material tends to zero, i.e. $c_R \to 0$, the moving material interface should become a moving reflecting mirror. We verify that the moving reflection and transmission solution collapses to the moving reflecting mirror solution [2]. The reflection and transmission coefficients for the moving reflecting mirror are

$$\frac{a_r}{a_i} = -\frac{c_L(k_i \cdot \hat{n}) - N}{c_L(k_r \cdot \hat{n}) - N}, \quad (3.6.4)$$

$$\frac{a_T}{a_i} = 0. \quad (3.6.5)$$
3.6.2 Amplitudes as Functions of Object Speed $\beta$

We plot the reflection and transmission coefficients for the normal moving interface (3.3.16) and (3.3.17) in Fig. 3.4. The oblique interface reflection and transmission coefficients behave similarly. The plot fixes $c_L = 1$, $c_R = 0.8$ and the incident angle $\theta_i = \pi/4$. First, note that when $\beta = 0$ we recover the result from the Fresnel equations that $a_T/a_i = 1 + a_r/a_i$ [72]. It is interesting to note that the curves in Fig. 3.4 are not symmetric with respect to $\beta = 0$. The reflection coefficient increases without bound as $\beta$ decreases and decreases without bound as $\beta$ increases. If the interface moves faster than $\beta \approx 0.25$, the sign of the reflection coefficient changes. In fact it is possible for the interface to move in such a way that the angle of incidence acts as a Brewster angle [72], where there is no reflection and the wave is entirely transmitted into the moving medium. The transmission coefficient $a_T/a_i$ decreases monotonically without bound.

![Figure 3.4: The reflection and transmission coefficients for a moving vertical interface as functions of $\beta$. We take $c_L = 1$, $c_R = 0.8$, and $\theta_i = \pi/4$.](image)

3.6.3 Spatial Dispersion

For a plane wave solution to the wave equation in $\mathcal{S}$ (2.2.34), the moving medium becomes spatially dispersive [35, 80], but remains temporally or frequency non-dispersive. Thus, there is locality in time within the moving material [49].
We derive the dispersion relation for when $S'$ moves relative to the fixed frame $S$ along the $x$–axis, i.e., $v_\alpha = (\nu_\alpha, 0)$. We assume the solution of (2.2.34) is a plane wave of the form

$$p = p_0 e^{i(k \cdot x - \omega t)}. \quad (3.6.6)$$

We substitute the plane waveform (3.6.6) into the wave equation (2.2.34) and simplify to get the dispersion relation

$$k^2 - \frac{\omega^2}{c_L^2} = \gamma^2 (n^2 - 1) \left( \frac{\omega}{c_L} - \beta k \cos \theta \right)^2,$$  

where $n = c_L/c_R$ is the index of refraction. We solve for the frequency in the moving medium as measured in $S$, $\omega$, as a function of $k$ and $\theta$ in the quadratic equation (3.6.7), with the proper choice of sign [80], to find that

$$\omega = \frac{c_L k \left\{ \gamma^2 \beta (n^2 - 1) \cos \theta + \sqrt{1 + \gamma^2 (n^2 - 1)(1 - \beta^2 \cos^2 \theta)} \right\}}{1 + \gamma^2 (n^2 - 1)}. \quad (3.6.8)$$

If we fix the angle $\theta$, the dispersion relation (3.6.8) is a linear function of $k$. At left in Figure 3.5 we plot the dispersion relation for fixed $\theta$ and several values of the material velocity $\beta$. Note that when $\beta = 0$ we recover the dispersion relation for the wave equation, $\omega = c_R k$. Now, if we fix the wavenumber $k$, the dispersion relation (3.6.8) is a non-linear function of the angle $\theta$. The right plot in Figure 3.5 shows the frequency $\omega$ as a function of $\theta$ for fixed $k = 1$. Interestingly, waves with the same wavenumber but different propagation angles travel with different wave speeds.

![Figure 3.5: The dispersion relation in moving object as a function of $\beta$ with $c_L = 1$ and $c_R = 0.8$ (left) Fixed propagation angle $\theta = \pi/4$. (right) Fixed wavenumber $k = 1$.](image-url)
CHAPTER 4

SPATIAL APPROXIMATION: THE DISCONTINUOUS GALERKIN SPECTRAL ELEMENT METHOD ON MOVING MESHES

In this chapter, we review the derivation of a high-order spatial approximation using an arbitrary Lagrangian-Eulerian (ALE) mapping and a discontinuous Galerkin spectral element method (DGSEM) to compute the solution to conservation laws on moving meshes. With the ALE mapping, the equations to be solved will depend on the mesh velocities. In Sec. 4.1, we transform the conservation laws in curvilinear coordinates for moving domains and describe the semi-discrete discontinuous Galerkin spectral element approximation (DGSEM) in two spatial dimensions. In Sec. 4.2, we describe how to compute mesh velocities from known grid information. An extension of the discretization to three spatial dimensions is provided in Sec. 4.3.

4.1 Discretization in Space of an ALE Conservation Law

The moving mesh DGSEM is derived from the weak form of a system of conservation laws

\[
\mathbf{q}_t + \nabla \cdot \mathbf{F} = 0,
\]

(4.1.1)
on the moving domain \(\Omega_t\). To obtain the weak form, we multiply by an \(L^2(\Omega_t)\) test function \(\varphi(x)\), and integrate in space over the moving domain

\[
\int_{\Omega_t} (\mathbf{q}_t + \nabla \cdot \mathbf{F}) \varphi \, dx.
\]

(4.1.2)

For the spectral element method, we subdivide the physical domain \(\Omega_t\) into \(K\) non-overlapping, moving elements, \(e_k\). We break the integrals over the entire domain into the sum of integrals over the moving elements

\[
\sum_{k=1}^{K} \left\{ \int_{e_k} (\mathbf{q}_t + \nabla \cdot \mathbf{F}) \varphi \, dx \right\} = 0.
\]

(4.1.3)

Since the test function \(\varphi\) is arbitrary, each element contributes

\[
\int_{e_k} (\mathbf{q}_t + \nabla \cdot \mathbf{F}) \varphi \, dx = 0
\]

(4.1.4)
to the total integral (4.1.3).

A common approach to handle complex geometries is to create a time dependent transformation \((x, t) = X(\xi, t)\) to map computational coordinates in the reference cube \(\xi = (\xi^1, \xi^2, \xi^3) = (\xi, \eta, \zeta)\) to physical coordinates \(x = (x, y, z) \in \Omega_t\) [56]. The conservation law in the time-dependent domain transforms to conservation law equations in the reference domain [29, 64, 68].

The curvilinear coordinate system for \(\xi\) on the reference cube has three covariant basis vectors, \(a_i\), computed directly from the transformation

\[
a_i = \frac{\partial x}{\partial \xi^i}, \quad i = 1, 2, 3. \tag{4.1.5}
\]

From the covariant basis vectors we derive the contravariant basis vectors \(a^i\), scaled by the Jacobian of the transformation \(J\)

\[
Ja^i = J\nabla \xi^i = a_j \times a_k, \quad (i, j, k) \text{ cyclic.} \tag{4.1.6}
\]

Alternatively, there is an explicitly divergence-free form of the contravariant basis vectors derived in [55]

\[
Ja_n^i = -\hat{x}_i \cdot \nabla \xi \times (x_l \nabla \xi x_m), \quad i = 1, 2, 3; \quad n = 1, 2, 3; \quad (n, m, l) \text{ cyclic.} \tag{4.1.7}
\]

The divergence-free form (4.1.7) of the contravariant basis vectors is particularly important to prevent spurious oscillations in the solution on static, curved sided hexahedral elements [55]. However, for two dimensional problems and straight-sided hexahedral meshes the cross product formulation (4.1.6) is sufficient to prevent the generation of spurious waves by a static mesh [55].

The conservation law in the time-dependent domain transforms to a conservation law equation in the reference domain [29, 64, 68]

\[
\tilde{q}_t + \nabla \xi \cdot \mathcal{F} = 0, \tag{4.1.8}
\]

where

\[
\tilde{q} = Jq, \quad \text{and} \quad \mathcal{F} = Ja^i \cdot (\mathcal{F} - qx_t). \tag{4.1.9}
\]

For complete details on the transformation of the conservation law see Acosta and Kopriva [1]. Notice that in the transformed variables (4.1.9), the solution \(\tilde{q}\) incorporates the time-dependent Jacobian \(J\) and the flux \(\mathcal{F}\) incorporates the mesh velocity \(x_t\).

To ensure that the discrete approximation does not introduce spurious waves on a moving mesh, a constant solution of (4.1.8) should remain constant for all time [1, 55]. In fluid dynamics this is
often referred to as free-stream preservation. The conditions that guarantee free-stream preservation for the transformation \( X(\xi, t) \) are three metric identities and the geometric conservation law [55, 68].

Because \( q_t = 0 \) for a constant solution we know the flux is also constant \( \mathscr{F} = C \). We substitute \( q = c \) in the conservation law written in computational coordinates to find

\[
\mathcal{J} c + \nabla_\xi \cdot (\mathcal{J} \mathbf{a}^i \cdot (C - cx_t)) = 0. \tag{4.1.10}
\]

Rewriting the spatial derivatives in (4.1.10) we have

\[
\sum_{i=1}^{3} \frac{\partial}{\partial \xi^i} (\mathcal{J} \mathbf{a}^i \cdot C) + c \left( \mathcal{J}_t - \sum_{i=1}^{3} \frac{\partial}{\partial \xi^i} (\mathcal{J} \mathbf{a}^i \cdot x_t) \right) = 0. \tag{4.1.11}
\]

We write the dot product in the first term of (4.1.11) explicitly and rearrange to find

\[
\sum_{n=1}^{3} C_n \left\{ \sum_{i=1}^{3} \frac{\partial (\mathcal{J} \mathbf{a}^i_n)}{\partial \xi^i} \right\} + c \left( \mathcal{J}_t - \sum_{i=1}^{3} \frac{\partial}{\partial \xi^i} (\mathcal{J} \mathbf{a}^i \cdot x_t) \right) = 0. \tag{4.1.12}
\]

Since \( c \) and \( C \) are arbitrary the metric identities

\[
\sum_{i=1}^{3} \frac{\partial (\mathcal{J} \mathbf{a}^i_n)}{\partial \xi^i} = 0, \quad n = 1, 2, 3, \tag{4.1.13}
\]

and the geometric conservation law (GCL)

\[
\mathcal{J}_t - \sum_{i=1}^{3} \frac{\partial}{\partial \xi^i} (\mathcal{J} \mathbf{a}^i \cdot x_t) = 0, \tag{4.1.14}
\]

must hold. For full details on the derivation see [55, 68]. The metric identities (4.1.13) state that the divergence of a constant flux vanishes in the reference cube. The geometric conservation law (4.1.14) states that for a constant solution the control volume is conserved. As we stated earlier, these four identities must be satisfied discretely at each time step to maintain free-stream preservation.

In what follows we will reduce the discussion to two dimensional problems and defer the description of the transfinite map and spatial approximation in three dimensions to Sec. 4.3. The ALE formulation (4.1.8), metric identities (4.1.13), and GCL (4.1.14) have been given in three dimensions since it is simpler to formulate how the equations transform in three dimensions. However, we now consider the approximation of the conservation law (4.1.8) on the reference square. For a two dimensional problem note that the we use the vector flux \( \mathscr{F} = (f, g) \) in the transformed variables (4.1.9).
Next we determine how the reference square is mapped to a moving quadrilateral element. Each element $e_k$ is bounded by four moving curves $\Gamma_j$, $j = 1, 2, 3, 4$. Kopriva [55] showed that the metric identities (4.1.13) are satisfied discretely if the boundaries of a quadrilateral element are approximated by a polynomial with order equal to or less than the order of the polynomial approximation of the solution. Therefore, to satisfy the metric identities (4.1.13) we use an isoparametric approximation in which each curve $\Gamma_j(s, t)$, $s \in [-1, 1]$ is approximated by a polynomial of order $N$. We write each curve in the Lagrange form

$$
\Gamma = \sum_{j=0}^{N} \Gamma(s_j, t) \ell_j(s).
$$

(4.1.15)

A common transformation between the reference square and any curve-sided quadrilateral is transfinite interpolation with linear blending [56]. The mapping between the coordinates of the reference square $E$ and the physical coordinates $X$ is

$$
X(\xi, \eta, t) = \frac{1}{2}[(1 - \xi)\Gamma_4(\eta, t) + (1 + \xi)\Gamma_2(\eta, t) + (1 - \eta)\Gamma_1(\xi, t) + (1 + \eta)\Gamma_3(\xi, t)]
- \frac{1}{4}[(1 - \xi)((1 - \eta)\Gamma_1(-1, t) + (1 + \eta)\Gamma_3(-1, t))]
+ (1 + \xi)((1 - \eta)\Gamma_1(1, t) + (1 + \eta)\Gamma_3(1, t)).
$$

(4.1.16)

Since the boundaries are approximated by polynomials, the transformation (4.1.16) is also a polynomial. Therefore, the calculation of the metric terms $X_\xi$ and $X_\eta$ is straightforward. We use the metric terms to approximate the normal vectors and time dependent Jacobian. See Kopriva [55] for more details on quadrilateral mappings for spectral methods. The mesh velocities $X_t$ can also be calculated given the time derivatives of the element’s boundary curves [1].

From the map $X(\xi, \eta, t)$, (4.1.16), the weak form on each moving element (4.1.4) becomes

$$
\iint_{E} (\bar{a}_t + \nabla_\xi \cdot F) \varphi \, d\xi d\eta = 0.
$$

(4.1.17)

We integrate by parts once in (4.1.17), separating boundary and interior contributions, to find

$$
\iint_{E} \bar{a}_t \varphi \, d\xi d\eta + \int_{\partial E} F \varphi \cdot \hat{n} \, dS - \iint_{E} F \cdot \nabla_\xi \varphi \, d\xi d\eta = 0.
$$

(4.1.18)

Note that $F$ is the vector of contravariant fluxes $F = (\bar{f}, \bar{g})$ given by

$$
\bar{f} = Y_\eta(f - X_tq) - X_\eta(g - Y_tq)
$$
$$
\bar{g} = -Y_\xi(f - X_tq) + X_\xi(g - Y_tq).
$$

(4.1.19)
with the mapping $X = (X, Y)$ and mesh velocity $X_t = (X_t, Y_t)$. The contravariant fluxes $\tilde{f}$ and $\tilde{g}$ come from the definition of the ALE flux (4.1.9).

We select the test function $\varphi$ to be the piecewise polynomial of degree $N$ in each direction on the reference element

$$\varphi^k = \sum_{i=0}^{N} \sum_{j=0}^{N} \varphi_{ij}^k \ell_i(\xi) \ell_j(\eta),$$

(4.1.20)

where $\{\ell_n\}_{n=0}^{N}$ is the Lagrange basis. We do not enforce continuity of the test function $\varphi$ at the element boundaries. We construct a nodal Galerkin approximation [56], so we replace the solution $\tilde{q}$ and contravariant fluxes $\tilde{f}$, $\tilde{g}$ by polynomial interpolants written in Lagrange form

$$\tilde{q} = \mathcal{J}q \approx \mathcal{J}Q = \sum_{n=0}^{N} \sum_{m=0}^{N} \tilde{Q}_{nm} \ell_n(\xi) \ell_m(\eta),$$

$$\tilde{f} \approx \tilde{F} = \sum_{n=0}^{N} \sum_{m=0}^{N} \tilde{F}_{nm} \ell_n(\xi) \ell_m(\eta),$$

(4.1.21)

$$\tilde{g} \approx \tilde{G} = \sum_{n=0}^{N} \sum_{m=0}^{N} \tilde{G}_{nm} \ell_n(\xi) \ell_m(\eta).$$

We substitute the polynomial representation of $\varphi$ (4.1.20) and the polynomial approximations (4.1.21) into the weak form (4.1.18) to obtain

$$\sum_{i=0}^{N} \sum_{j=0}^{N} \left\{ \int_{E} \tilde{Q}_i \ell_i \ell_j d\xi d\eta + \int_{\partial E} \tilde{F}_i \ell_i \ell_j \cdot \hat{n} dS - \int_{E} \tilde{F} \cdot \nabla \mathcal{L}(\ell_i \ell_j) d\xi d\eta \right\} \varphi_{ij} = 0,$$

(4.1.22)

where $F = (\tilde{F}, \tilde{G})$. Since the values of $\varphi_{ij}$ are arbitrary and linearly independent,

$$\int_{E} \tilde{Q}_i \ell_i d\xi d\eta + \int_{\partial E} \tilde{F}_i \ell_i \ell_j \cdot \hat{n} dS - \int_{E} F \cdot \nabla \mathcal{L}(\ell_i \ell_j) d\xi d\eta = 0.$$

(4.1.23)

To complete the spatial discretization, we choose the location of the nodes in the approximation. We select the Lagrange basis that interpolates the Legendre-Gauss nodes, which are entirely interior to the reference element. We approximate the integrals in (4.1.23) with Legendre-Gauss quadrature, replace the flux at element boundaries with a numerical flux, to obtain the semi-discrete approximation of (4.1.8),

$$\frac{d\tilde{Q}_{ij}}{dt} + \left\{ \left[ \tilde{F}^*(1, \eta_j) \frac{\ell_i(1)}{\omega_i^1} - \tilde{F}^*(-1, \eta_j) \frac{\ell_i(-1)}{\omega_i^{-1}} \right] + \sum_{n=0}^{N} \tilde{F}_{nj} \tilde{D}^{(\xi)}_{in} \right\}$$

$$+ \left\{ \left[ \tilde{G}^*(\xi_i, 1) \frac{\ell_j(1)}{\omega_j^1} - \tilde{G}^*(\xi_i, -1) \frac{\ell_j(-1)}{\omega_j^{-1}} \right] + \sum_{n=0}^{N} \tilde{G}_{in} \tilde{D}^{(\eta)}_{jn} \right\}$$

(4.1.24)

$$= \frac{d\tilde{Q}_{ij}}{dt} + L^p_N(Q, X, X_t) = 0,$$

39
for $i, j = 0, \ldots N$, where
\[ \hat{D}_{jk} = -\frac{D_{kj}\omega_k}{\omega_j}. \] (4.1.25)

The matrix $\hat{D}_{jk}$ is the transpose of the standard polynomial derivative matrix $D_{kj} = \ell'_j(\xi_k)$ scaled by the weights from the Legendre-Gauss quadrature. For complete details see [56]. A method to derive exact, upwind numerical fluxes will be discussed in Chap. 5.

The approximation should satisfy the GCL (4.1.14), which we rewrite in the form of a conservation law
\[ \frac{\partial J}{\partial t} + \nabla \xi \cdot \Psi = 0, \] (4.1.26)
where $\Psi = -J a^i \cdot x_i$. We approximate $\Psi$ by a polynomial $\tilde{\Psi} = \tilde{E} \hat{\xi} + \tilde{H} \hat{\eta}$ where
\[ \tilde{E}_{ij} = -(X_i Y_j - Y_i X_j)_{ij}, \]
\[ \tilde{H}_{ij} = -(-X_i Y_j + Y_i X_j)_{ij}. \] (4.1.27)

Following the same steps as above, and integrating with the same spatial approximation as is used for the solution, the semi-discrete approximation of the GCL is
\[ \frac{d\tilde{J}_{ij}}{dt} + \left\{ \left[ \tilde{E}^*(1, \eta_j) \frac{\ell_i(1)}{\omega_i} \ell_j(-1) \frac{\ell_j(-1)}{\omega_j} \right] + \sum_{n=0}^{N} \tilde{E}_{nj} \hat{D}^{(\xi)}_{jn} \right\} \\
+ \left\{ \left[ \tilde{H}^*(\xi_i, 1) \frac{\ell_j(1)}{\omega_j} \ell_i(-1) \frac{\ell_i(-1)}{\omega_i} \right] + \sum_{n=0}^{N} \tilde{H}_{in} \hat{D}^{(\eta)}_{jn} \right\} = \frac{d\tilde{J}_{ij}}{dt} + L_{ij}^{(Q)}(Q, X) = 0. \] (4.1.28)

The quantity $\tilde{J}$ is a polynomial of degree $N$.

The GCL numerical fluxes, $\tilde{E}^*$ and $\tilde{H}^*$, are calculated from the upwind value at the boundary, that is, from the dot product of the outward normal vector $\hat{n}$ and the corresponding boundary mesh velocity calculated from the transfinite interpolation. For instance, $\tilde{H}^*(\xi_i, 1, t)$ is calculated as
\[ \tilde{H}^*(\xi_i, 1, t) = X_i(\xi_i, 1, t) \cdot \hat{n}. \] (4.1.29)

The semi-discrete formulation of the system of conservation laws (4.1.24) and geometric conservation law (4.1.28) use the same spatial discretization. Therefore, we integrate an augmented system of ordinary differential equations
\[ \frac{dU}{dt} \bigg|_{ij} + L(Q, X, X_t) \bigg|_{ij} = 0, \] (4.1.30)
where

\[ U_{ij} = \begin{bmatrix} \tilde{Q} \\ \tilde{J} \end{bmatrix}_{ij} \quad L_{ij} = \begin{bmatrix} L^P_{ij} \\ L^J_{ij} \end{bmatrix}. \] (4.1.31)

Acosta and Kopriva [1] proved that using \( Q_{ij} = \tilde{Q}_{ij} / \tilde{J}_{ij} \) preserves constant solutions and therefore avoids the generation of spurious waves that are an artifact of the mesh motion.

### 4.2 Computation of the Mesh Velocity

The time accuracy of the discrete solution of the augmented system of ODEs (4.1.31) is affected by errors introduced by the temporal discretization as well as errors in the mesh positions and mesh velocities [1].

Acosta and Kopriva [1] cast methods to calculate the mesh velocities into three categories: (i) exact differentiation of the mapping \( X(\xi, t) \), (ii) integration of an acceleration equation, and (iii) numerical differentiation of the mesh position via the time integrator (Inverse operator). Full time accuracy is obtained with any of these three methods.

We select a spring-mass dashpot analogy [68], which falls into category (ii), to compute mesh velocities. The spring-mass dashpot analogy replaces the edges of an element in a computational mesh by springs. The force exerted by the springs on a node by its neighboring nodes is calculated from Hooke’s law. Controlling the motion of the mesh is done by modifying the stiffness, rest length and damping constants. Cross springs, connecting opposing corners, are included to help prevent the mesh from crossing itself. The dynamics of the coupled spring system is governed by Newton’s second law of motion

\[ \mathbf{F} = m \mathbf{A}, \] (4.2.1)

where \( \mathbf{F} \) is the sum of the forces, \( m \) is the mass, and \( \mathbf{A} \) is the resulting acceleration.

We move the mesh in response to some arbitrary motion of a physical boundary with a spring-mass dashpot analogy. As a result, there is no analytical representation for the position or the velocity of the mesh nodes. Instead, the motion is computed as part of the solution process. We write the acceleration equation (4.2.1) as a system of first order ODEs using a change of variables. The motion of the mesh at each node is then described by

\[
\begin{align*}
\frac{d\mathbf{X}}{dt} &= \mathbf{W}(t), \\
\frac{d\mathbf{W}}{dt} &= \mathbf{A}(\mathbf{X}, \mathbf{W}, t) = \frac{1}{m} \mathbf{F}(\mathbf{X}, \mathbf{W}, t),
\end{align*}
\] (4.2.2)
where \( m \) is a mass, \( X \) is the position, \( W \) is the velocity, and \( A \) is the acceleration of the mesh.

We integrate the spring-mass dashpot ODE system (4.2.2) along with the augmented system of ODEs (4.1.30)

\[
\begin{bmatrix}
  U \\
  X_a \\
  W_a
\end{bmatrix}_t + \begin{bmatrix}
  L(U, X_a, W_a) \\
  A(X_a, W_a, t)
\end{bmatrix} = 0,
\]

(4.2.3)

where \( X_a \) is the approximate mesh position at a fixed time \( t \), and \( W_a \) is the approximate mesh velocity. For simplicity we have omitted the \( ij \) subscripts.

Acosta and Kopriva [1] showed that using the same time integrator to approximate the position and mesh velocity in this way is sufficient to preserve the temporal accuracy of an explicit global time stepping approximation of (4.2.3).

### 4.3 Three Dimensional Discontinuous Galerkin Method on a Moving Hexahedral Element

The preceding detailed description in Secs. 4.1 and 4.2 of a two dimensional discontinuous Galerkin spectral element approximation will facilitate the derivation a three dimensional DGSEM as the discretization follows the same steps. We already have the ALE formulation (4.1.8), metric identities (4.1.13), and GCL (4.1.14) in three dimensions.

However, we are missing a transformation between the reference cube and an arbitrary, moving, curve-sided hexahedron. We, again, select a transfinite interpolation with linear blending to map between \( E = [-1, 1]^3 \) and the physical coordinates \((x, t)\). To create this transformation we use a linear interpolation between the six moving faces \( \Gamma_j, j = 1, \ldots, 6 \) that represent the boundaries of an element, as shown in Fig. 4.1.

To derive the mapping, we first create a linear interpolation between two opposing faces, say \( \Gamma_3 \) and \( \Gamma_5 \)

\[
X_{35}(\xi, \eta, t) = \frac{1}{2}\{(1 - \zeta)\Gamma_3(\xi, \eta, t) + (1 + \zeta)\Gamma_5(\xi, \eta, t)\}.
\]

(4.3.1)

Similarly, we construct the linear interpolation for the other four moving faces as

\[
\begin{align*}
X_{12}(\xi, \zeta, t) &= \frac{1}{2}\{(1 - \eta)\Gamma_1(\xi, \zeta, t) + (1 + \eta)\Gamma_2(\xi, \zeta, t)\}, \\
X_{64}(\eta, \zeta, t) &= \frac{1}{2}\{(1 - \xi)\Gamma_6(\eta, \zeta, t) + (1 + \xi)\Gamma_4(\eta, \zeta, t)\}.
\end{align*}
\]

(4.3.2)
The final mapping will be a combination of the six interpolations, starting with the sum
\[ \mathbf{\Sigma}(\xi, \eta, \zeta, t) = \frac{1}{2} \left\{ (1 - \xi)\Gamma_6(\eta, \zeta, t) + (1 + \xi)\Gamma_4(\eta, \zeta, t) + (1 - \eta)\Gamma_1(\xi, \zeta, t) \\
+ (1 + \eta)\Gamma_2(\xi, \zeta, t) + (1 - \zeta)\Gamma_3(\xi, \eta, t) + (1 + \zeta)\Gamma_5(\xi, \eta, t) \right\}. \] (4.3.3)

In general, the combination (4.3.3) no longer matches the faces, e.g.,
\[ \mathbf{\Sigma}(-1, \eta, \zeta, t) = \Gamma_6(\eta, \zeta, t) + \frac{1}{2} \left\{ (1 - \eta)\Gamma_1(-1, \zeta, t) + (1 + \eta)\Gamma_2(-1, \zeta, t) \\
+ (1 - \zeta)\Gamma_3(-1, \eta, t) + (1 + \zeta)\Gamma_5(-1, \eta, t) \right\}, \] (4.3.4)
\[ \mathbf{\Sigma}(1, \eta, \zeta, t) = \Gamma_4(\eta, \zeta, t) + \frac{1}{2} \left\{ (1 - \eta)\Gamma_1(1, \zeta, t) + (1 + \eta)\Gamma_2(1, \zeta, t) \\
+ (1 - \zeta)\Gamma_3(1, \eta, t) + (1 + \zeta)\Gamma_5(1, \eta, t) \right\}, \] (4.3.5)
\[ \mathbf{\Sigma}(\xi, -1, \zeta, t) = \Gamma_1(\xi, \zeta, t) + \frac{1}{2} \left\{ (1 - \xi)\Gamma_6(-1, \zeta, t) + (1 + \xi)\Gamma_4(-1, \zeta, t) \\
+ (1 - \zeta)\Gamma_3(\xi, -1, t) + (1 + \zeta)\Gamma_5(\xi, -1, t) \right\}, \] (4.3.6)
\[ \mathbf{\Sigma}(\xi, 1, \zeta, t) = \Gamma_2(\xi, \zeta, t) + \frac{1}{2} \left\{ (1 - \xi)\Gamma_6(1, \zeta, t) + (1 + \xi)\Gamma_4(1, \zeta, t) \\
+ (1 - \zeta)\Gamma_3(\xi, 1, t) + (1 + \zeta)\Gamma_5(\xi, 1, t) \right\}, \] (4.3.7)
\[ \mathbf{\Sigma}(\xi, \eta, -1, t) = \Gamma_3(\xi, \eta, t) + \frac{1}{2} \left\{ (1 - \eta)\Gamma_1(\xi, -1, t) + (1 + \eta)\Gamma_2(\xi, -1, t) \\
+ ((1 - \xi)\Gamma_6(\eta, -1, t) + (1 + \xi)\Gamma_4(\eta, -1, t) \right\}, \] (4.3.8)
\[
\Sigma(\xi, \eta, 1, t) = \Gamma_5(\xi, \eta, t) + \frac{1}{2} \left\{ (1 - \eta)\Gamma_1(\xi, 1, t) + (1 + \eta)\Gamma_2(\xi, 1, t) \right. \\
+ (1 - \xi)\Gamma_6(\eta, 1, t) + (1 + \xi)\Gamma_4(\eta, 1, t) \}. \tag{4.3.9}
\]

To match the faces, we need to subtract the linear interpolant in the \(\xi\), \(\eta\), and \(\zeta\) directions of the additional terms that appear in the braces of (4.3.4)–(4.3.9). We find these linear interpolants to be

\[
\mathcal{C}_\xi = \left(\frac{1 - \xi}{2}\right) \left[ \frac{1}{2} \left\{ (1 - \eta)\Gamma_1(-1, \zeta, t) + (1 + \eta)\Gamma_2(-1, \zeta, t) + (1 - \zeta)\Gamma_3(-1, \eta, t) \right. \\
+ (1 + \zeta)\Gamma_5(-1, \eta, t) \right] + \left(\frac{1 + \xi}{2}\right) \left[ \frac{1}{2} \left\{ (1 - \eta)\Gamma_1(1, \zeta, t) \\
+ (1 + \zeta)\Gamma_5(1, \eta, t) + (1 - \zeta)\Gamma_3(1, \eta, t) + (1 + \zeta)\Gamma_5(1, \eta, t) \right. \right], \tag{4.3.10}
\]

\[
\mathcal{C}_\eta = \left(\frac{1 - \eta}{2}\right) \left[ \frac{1}{2} \left\{ (1 - \xi)\Gamma_6(-1, \zeta, t) + (1 + \xi)\Gamma_4(-1, \zeta, t) + (1 - \zeta)\Gamma_3(-1, \xi, t) \right. \\
+ (1 + \zeta)\Gamma_5(-1, \xi, t) \right] + \left(\frac{1 + \eta}{2}\right) \left[ \frac{1}{2} \left\{ (1 - \xi)\Gamma_6(1, \zeta, t) \\
+ (1 + \zeta)\Gamma_4(1, \zeta, t) + (1 - \zeta)\Gamma_3(1, \zeta, t) + (1 + \zeta)\Gamma_5(1, \zeta, t) \right. \right], \tag{4.3.11}
\]

and

\[
\mathcal{C}_\zeta = \left(\frac{1 - \zeta}{2}\right) \left[ \frac{1}{2} \left\{ (1 - \eta)\Gamma_1(\xi, -1, t) + (1 + \eta)\Gamma_2(\xi, -1, t) + (1 - \xi)\Gamma_6(\eta, -1, t) \right. \\
+ (1 + \xi)\Gamma_4(\eta, -1, t) \right] + \left(\frac{1 + \zeta}{2}\right) \left[ \frac{1}{2} \left\{ (1 - \eta)\Gamma_1(\xi, 1, t) \\
+ (1 + \xi)\Gamma_2(\xi, 1, t) + (1 - \xi)\Gamma_6(\eta, 1, t) + (1 + \xi)\Gamma_4(\eta, 1, t) \right. \right]. \tag{4.3.12}
\]

However, subtracting the correction terms (4.3.10), (4.3.11), and (4.3.12) from (4.3.3) removes the interior contributions twice. Thus, to complete the correction to (4.3.3), we add the transfinite map of the reference cube to a straight-sided hexahedral element

\[
\mathbf{X}_H(\xi, t) = \frac{1}{8} \left\{ x_1(1 - \xi)(1 - \eta)(1 - \zeta) + x_2(1 + \xi)(1 - \eta)(1 - \zeta) \\
+ x_3(1 + \xi)(1 + \eta)(1 - \zeta) + x_4(1 - \xi)(1 + \eta)(1 - \zeta) \\
+ x_5(1 - \xi)(1 - \eta)(1 + \zeta) + x_6(1 + \xi)(1 - \eta)(1 + \zeta) \\
+ x_7(1 + \xi)(1 + \eta)(1 + \zeta) + x_8(1 - \xi)(1 + \eta)(1 + \zeta) \right\}, \tag{4.3.13}
\]

where \(x_i, i = 1, \ldots, 8\) are the time dependent corners of the hexahedron. We arrive at the final, time dependent transfinite interpolation with linear blending for a curved-sided hexahedron

\[
\mathbf{X}(\xi, t) = \Sigma(\xi, t) - \frac{1}{2} [\mathcal{C}_\xi + \mathcal{C}_\eta + \mathcal{C}_\zeta] + \mathbf{X}_H(\xi, t), \tag{4.3.14}
\]

44
where we divide the correction terms (4.3.10), (4.3.11), and (4.3.12) in half as they count the contribution from each of the twelve faces twice. The faces are approximated by polynomials so the calculation of the metric terms $X_\xi$, $X_\eta$, and $X_\zeta$ is straightforward. We use the metric terms to approximate the contravariant basis vectors given by (4.1.6) for straight-sided or (4.1.7) for curved sided hexahedrons as well as normal vectors and time dependent Jacobian.

With the three dimensional transformation (4.3.14) we follow the same steps as the two dimensional case, i.e., replace integrals with high-order Gauss quadrature, replace boundary integrals with an appropriate numerical flux, etc. We obtain a three dimensional semi-discrete DG approximation of the form

$$
\frac{d\tilde{\mathbf{Q}}_{ijk}}{dt} + \left\{ \left[ \mathbf{F}^*(1, \eta_j, \zeta_k) \frac{\ell_i(1)}{\omega_i^\eta} - \mathbf{F}^*(-1, \eta_j, \zeta_k) \frac{\ell_i(-1)}{\omega_i^\eta} \right] + \sum_{n=0}^N \tilde{\mathbf{F}}_{njk} \tilde{D}_i^n \right\}
+ \left\{ \left[ \mathbf{G}^*(\xi_i, 1, \zeta_k) \frac{\ell_j(1)}{\omega_j^\zeta} - \mathbf{G}^*(\xi_i, -1, \zeta_k) \frac{\ell_j(-1)}{\omega_j^\zeta} \right] + \sum_{n=0}^N \tilde{\mathbf{G}}_{ink} \tilde{D}_j^n \right\}
+ \left\{ \left[ \mathbf{S}^*(\xi_i, \eta_j, 1) \frac{\ell_k(1)}{\omega_k^\xi} - \mathbf{S}^*(\xi_i, \eta_j, -1) \frac{\ell_k(-1)}{\omega_k^\xi} \right] + \sum_{n=0}^N \tilde{\mathbf{S}}_{ijn} \tilde{D}_k^n \right\}.
$$

(4.3.15)

where $i,j,k = 0,\ldots,N$, the $^*$ represents the numerical flux, and $\tilde{\mathbf{S}}$ is the physical flux augmented by the mesh velocities in the $z$–direction, i.e.,

$$
\tilde{\mathbf{S}} = a_1^3(f - X_t q) + a_2^3(g - Y_t q) + a_3^3(s - Z_t q),
$$

(4.3.16)

with contravariant vector $\mathbf{a}^3$ computed from the explicitly divergence-free form (4.1.7), physical fluxes $(f, g, s)$, and mesh velocity $\mathbf{X}_t = (X_t, Y_t, Z_t)$. Notice in (4.3.15) that, just like in two dimensions, the tensor product approximation decouples the derivatives in the three space directions. We discretize the three dimensional GCL in a way similar to what was shown in (4.1.28).

Just as before, the three dimensional semi-discrete formulation of the system of conservation laws (4.3.15) and geometric conservation law use the same spatial discretization. Also, the spring-mass dashpot analogy extends easily to hexahedral meshes by replacing edges with springs as well as cross springs connecting opposing corners on a face and across the hexahedron. So, we can integrate an augmented system of ordinary differential equations for the solution, GCL, and spring model, similar to (4.2.3), for the three dimensional problem.

45
CHAPTER 5

THE PHYSICS II: RIEMANN SOLVERS FOR WAVE PROPAGATION IN MOVING MEDIA

In this chapter we determine appropriate numerical fluxes, e.g. $\tilde{F}^*$ in (4.1.24), to compute the flux across a moving interface. To do so we explore the solution of the Riemann problem that arises at a moving material interface. In general, a Riemann problem consists of a conservation law together with piecewise constant data which has a single discontinuity. The Riemann problem is particularly useful for understanding hyperbolic PDEs because all properties, such as shocks and rarefaction waves, appear as characteristics in the solution [30, 90].

In numerical analysis, a Riemann problem appears in a natural way at boundaries between elements in finite volume methods [63] as well as discontinuous Galerkin finite element [16] and spectral element [56] approximations. In addition to computing the flux across an interface, the solution of a Riemann problem is used to weakly enforce boundary conditions for these families of numerical approximations. One can use the boundary conditions to define an analytical representation of an external state $q_{\text{ext}}$. This weak enforcement gives the approximation flexibility because one doesn’t have to alter the numerical scheme if the boundary conditions change.

Thus, interpreting a Riemann problem and it’s solution, called a Riemann solver or numerical flux, offers important insight to understand a physical problem and has practical use in its numerical approximation. In Sec. 5.1 we derive exact, upwind numerical fluxes for the TE and TM Maxwell’s equations as well as the classical wave equation using the derived physical fluxes from Chap. 2. We verify consistency of the newly derived numerical fluxes with previous results for wave scattering from a moving mirror in Sec. 5.2.

5.1 Exact, Upwind Numerical Fluxes

To rigorously define the Riemann problem, suppose we have two states, $q_L$ and $q_R$, that describe the solution on the left and right side of a moving material interface. For generality we take the
material properties to differ across the interface and that the dielectric slab moves with velocity $v_o$. We illustrate the single jump at the moving dielectric interface in Fig. 5.1.

![Figure 5.1: Left and right states viewed at a moving boundary. The boundary moves with velocity $v_o$.](image)

We now are ready to derive, for moving media, the exact, fully upwind numerical fluxes for the Maxwell’s equations and the classical wave equation. The details of the physical analysis for the two wave propagation models differ [62, 98]. However, there are similarities, e.g., the moving medium alters the wave speed because the material will “pull” or “push,” with or against, a traveling wave solution.

### 5.1.1 Maxwell’s Equations

We first derive the physical fluxes and exact upwind numerical flux for TE polarized waves. We apply the TE results and the principle of duality (2.1.2) to simplify the discussion of TM polarized waves.

Recall in Sec. 2.2 we wrote the conservation law (2.2.8) for Maxwell’s equations with TE polarization compactly in matrix form as

$$\mathbf{q}_t + \mathbf{B}^{TE} \mathbf{q}_x + \mathbf{C}^{TE} \mathbf{q}_y = 0,$$

(5.1.1)

where $\mathbf{q} = [D_z \ B_x \ B_y]^T$ are the unknown electric and magnetic field quantities and $\mathbf{B}^{TE}$ (2.2.19), $\mathbf{C}^{TE}$ (2.2.20) are the physical flux matrices in the $x$ and $y$ direction respectively.

To solve the general Riemann problem at the moving dielectric interface the flux matrices will incorporate the arbitrary mesh velocity $\mathbf{x}_t = (x_t, y_t)$ that may differ from the object velocity $v_o$. 

47
This is because in the ALE formulation from Sec. 4.1 the physical fluxes incorporate the motion of the mesh (4.1.9). The physical fluxes augmented with the velocity \( \mathbf{x}_t \) are

\[
\mathbf{F} = \mathbf{F}^{TE} - \mathbf{q}_t = (\mathbf{B}^{TE} - x_t \mathbf{I}, \mathbf{C}^{TE} - y_t \mathbf{I}) \mathbf{q} = (\tilde{\mathbf{B}}^{TE}, \tilde{\mathbf{C}}^{TE}) \mathbf{q},
\]

where

\[
\tilde{\mathbf{B}}^{TE} = \begin{bmatrix}
-\nu_x \overline{\omega} - x_t \\
0 \\
-\mu(1 - \beta^2)/\alpha \\
\end{bmatrix}
\begin{bmatrix}
(\varepsilon \mu_L \nu_x \nu_y \overline{\omega})/\mu \\
-x_t \\
\nu_y \overline{\omega} \\
-\nu_x \overline{\omega} - x_t \\
\end{bmatrix},
\]

and

\[
\tilde{\mathbf{C}}^{TE} = \begin{bmatrix}
-\nu_y \overline{\omega} - y_t \\
0 \\
\mu(1 - \beta^2)/\alpha \\
\end{bmatrix}
\begin{bmatrix}
(1 + \varepsilon \mu_L \nu_x \nu_y \overline{\omega})/\mu \\
-y_t \\
-\nu_x \overline{\omega} - y_t \\
0 \\
\end{bmatrix}.
\]

We are interested in a numerical flux that accounts for a jump in material properties at the moving interface. However, the characteristics have a jump discontinuity when the permittivity and permeability abruptly change at the interface. So, instead of requiring characteristic lines to be continuous at an interface, we apply the Rankine-Hugoniot condition to guarantee that the normal flux across an interface is continuous.

So, we require the computation of the fluxes along the normal direction of a moving interface, i.e.,

\[
\hat{n} \cdot \mathbf{F} = (\tilde{\mathbf{B}}^{TE} n_x + \tilde{\mathbf{C}}^{TE} n_y) \mathbf{q} = \mathbf{A}^{TE} \mathbf{q}.
\]

From (5.1.3) and (5.1.4) we define the normal coefficient matrix for the TE problem by

\[
\mathbf{A}^{TE} = \begin{bmatrix}
-\overline{\omega} - s \\
\mu \nu_x (1 - \beta^2) \\
\nu_y \overline{\omega} n_y \\
\end{bmatrix}
\begin{bmatrix}
\varepsilon \mu_L \nu_x \nu_y \overline{\omega} + \nu_x \overline{\omega} N \\
\varepsilon \mu_L \nu_x \nu_y \overline{\omega} + \nu_x \overline{\omega} N \\
\nu_x \overline{\omega} n_x \\
\end{bmatrix},
\]

where we refer to \( s = \hat{n} \cdot \mathbf{x}_t = x_t n_x + y_t n_y \) as the mesh speed.

We now derive the fully upwind numerical flux for the TE polarized Maxwell’s equations at a moving dielectric interface. The numerical flux, \( \mathbf{F}^{*,TE}(\mathbf{q}_L, \mathbf{q}_R; \hat{n}, s) \), is a function of the two solution states \( \mathbf{q}_L \) and \( \mathbf{q}_R \).

To derive the upwind numerical flux we first decompose the coefficient matrix \( \mathbf{A}^{TE} \) by the similarity transformation of its eigenvectors and eigenvalues

\[
\mathbf{A}^{TE} = \mathbf{P} \Lambda \mathbf{P}^{-1}.
\]
The eigenvalues of the matrix $A^{TE}$ are

$$
\lambda_0 = -s, \quad \lambda_{\pm} = c^\pm - s, \quad (5.1.8)
$$

where

$$
c^\pm = -\Xi N \pm \frac{1}{\alpha} \sqrt{(1 - \beta^2) \left\{ \varepsilon \mu (1 - \varepsilon \mu L N^2) - \varepsilon^2 \mu_L^2 \Xi^2 \right\}}, \quad (5.1.9)
$$

with $\Xi = \hat{n} \times v_o = \nu_y n_x - \nu_x n_y$. The eigenvalues (5.1.8) give the altered wavespeeds for the numerical approximation. There are two independent mechanisms that change the wavespeed: the non-physical motion of the mesh incorporated from the ALE mapping and the physical motion of the dielectric described mathematically with the Lorentz transformation. The non-physical motion alters the wavespeeds by the mesh speed $s$ in (5.1.8). The motion of the dielectric changes the wavespeed to $c^\pm$ in (5.1.9). We see from (5.1.9) that, on the left of the dielectric interface, waves travel with speed $\pm c_L$. To the right of the dielectric interface, the wave speed is impeded or boosted by the motion of the dielectric slab.

The solution to the PDEs (5.1.1) is a linear combination of left-going and right-going waves. The system is strictly hyperbolic since the matrix $A^{TE}$ has real, distinct eigenvalues [86]. The system admits two waves that move with speed $c^\pm - s$ along any wavevector in the plane, and another that moves at speed $-s$. The three right eigenvectors associated with the eigenvalues $\lambda = -s, c^\pm - s$ are

$$
e^0 = \begin{bmatrix}
\Xi \varepsilon \mu L \mu_L - \varepsilon \mu \\
n_x - \varepsilon \mu L \mu_L \nu_y \Xi - \varepsilon \mu \nu_x N \\
n_y + \varepsilon \mu L \mu_L \nu_z \Xi - \varepsilon \mu \nu_y N
\end{bmatrix} = \begin{bmatrix}
\hat{a} \\
\hat{b} \\
\hat{c}
\end{bmatrix}, \quad (5.1.10)
$$

and

$$
e^+ = \frac{1}{2} \begin{bmatrix}
n_y Z^{TE} \\
-n_x Z^{TE}
\end{bmatrix}, \quad e^- = \frac{1}{2} \begin{bmatrix}
n_y Z^{TE} \\
n_x Z^{TE}
\end{bmatrix}, \quad (5.1.11)
$$

where $Z^{TE}$ is the augmented dielectric impedance for the TE polarization

$$
Z^{TE} = \sqrt{\frac{\mu^2 (1 - \beta^2)}{\varepsilon \mu (1 - \varepsilon \mu L N^2) - \varepsilon^2 \mu_L^2 \Xi^2}}. \quad (5.1.12)
$$

Explicitly, in the eigendecomposition (5.1.7) we have $\Lambda = \text{diag}(\lambda_+, \lambda_-, \lambda_0)$, the matrix of right eigenvectors

$$
P = \frac{1}{2} \begin{bmatrix}
n_y Z^{TE} & 1 \\
-n_y Z^{TE} & 2 \hat{a} \\
-n_x Z^{TE} & n_x Z^{TE}
\end{bmatrix}, \quad (5.1.13)$$

49
and the matrix of left eigenvectors

\[ P^{-1} = \frac{1}{bn_x + cn_y} \begin{bmatrix} 1 & \hat{c}Y^{TE} - \hat{a}n_x & -\hat{b}Y^{TE} - \hat{a}n_y \\ 1 & -\hat{c}Y^{TE} - \hat{a}n_x & \hat{b}Y^{TE} - \hat{a}n_y \\ 0 & n_x & n_y \end{bmatrix}, \quad (5.1.14) \]

where \( Y^{TE} = 1/Z^{TE} \). With the eigendecomposition of \( A^{TE} \) we are equipped to solve the Riemann problem at the moving dielectric interface.

To determine the upwind directions, we decouple the solution vector \( q \) into left-going and right-going wave components. Decoupling is done using the left eigenvectors of the coefficient matrix \( A^{TE} \). We define the characteristic variables \( w \) as

\[ w = P^{-1}q = \frac{1}{bn_x + cn_y} \begin{bmatrix} 1 & \hat{c}Y^{TE} - \hat{a}n_x & -\hat{b}Y^{TE} - \hat{a}n_y \\ 1 & -\hat{c}Y^{TE} - \hat{a}n_x & \hat{b}Y^{TE} - \hat{a}n_y \\ 0 & n_x & n_y \end{bmatrix} \begin{bmatrix} D_z \\ B_x \\ B_y \end{bmatrix} = \begin{bmatrix} w^+ \\ w^- \\ w^0 \end{bmatrix}. \quad (5.1.15) \]

Now that we know the characteristic variables \( w \) and eigendecomposition of \( A^{TE} \) we apply the Rankine-Hugoniot condition to separate known and unknown traveling wave components. For Maxwell’s equations we have

\[ A_{L}^{TE}q_L = A_{R}^{TE}q_R, \quad (5.1.16) \]

where the normal flux matrix \( A^{TE} \) (5.1.6) incorporates the motion of the material, the motion of the mesh and depends on \( \varepsilon, \mu \).

The wave speeds (5.1.8) depend on the mesh speed \( s \). If we assume \( |s| < c^\pm \), that is the mesh does not overtake any wave, then we have two cases to consider for the solution of the Riemann problem:

1. If \( s > 0 \), then \( w^+_L, w^-_R \) and \( w^0_R \) are known and \( w^-_L, w^+_R \) and \( w^0_L \) are unknown.

2. If \( s < 0 \), then \( w^+_L, w^-_R \) and \( w^0_L \) are known and \( w^-_L, w^+_R \) and \( w^0_R \) are unknown.

First we assume \( s > 0 \). The unknown values \( w^-_L, w^+_R \), and \( w^0_L \) will be found from the Rankine-Hugoniot condition (5.1.16). We rewrite the Rankine-Hugoniot condition in terms of the characteristic variables (5.1.15)

\[ P_L \Lambda_L w_L = P_R \Lambda_R w_R. \quad (5.1.17) \]

We write the matrix of right eigenvectors \( P \) (5.1.13) in a compressed form using the notation for the right eigenvectors of \( A^{TE} \)

\[ P = [e^+ \; e^- \; e^0]. \quad (5.1.18) \]
Thus,
\[
P_L A_L = [(c_L^+ - s) e_L^+ (c_L^- - s) e_L^- - s e_L^0],
\]
(5.1.19)
and
\[
P_L A_L w_L = (c_L^+ - s) w_L^+ e_L^+ + (c_L^- - s) w_L^- e_L^- - s w_L^0 e_L^0.
\]
(5.1.20)

The Rankine-Hugoniot condition (5.1.17) becomes
\[
(c_L^+ - s) w_L^+ e_L^+ + (c_L^- - s) w_L^- e_L^- - s w_L^0 e_L^0 = (c_R^+ - s) w_R^+ e_R^+ + (c_R^- - s) w_R^- e_R^- - s w_R^0 e_R^0.
\]
(5.1.21)

We separate the known values of \(w_L^+, w_R^-\) and \(w_R^0\) from the unknown values of \(w_L^-, w_L^+\) and \(w_R^0\) and rewrite (5.1.21) in system form to obtain
\[
\begin{bmatrix}
    c_R^+ - s \\
    n_y Z_TE(c_R^+ - s) \\
    -n_z Z_TE(c_R^+ - s)
\end{bmatrix}
\begin{bmatrix}
    w_L^- \\
    w_R^- \\
    w_R^0
\end{bmatrix}
= \begin{bmatrix}
    c_R^+ - s \\
    n_y Z_TE(c_R^+ - s) \\
    -n_z Z_TE(c_R^+ - s)
\end{bmatrix}
\begin{bmatrix}
    w_L^+ \\
    w_R^+ \\
    w_R^0
\end{bmatrix}.
\]
(5.1.22)

We solve for \(w_L^+, w_R^-\), and \(w_R^0\) to find
\[
\begin{align*}
w_L^- &= \frac{(c_L^+ - s)(Z_{TE}^+ - Z_{TE}^-)}{(c_L^- - s)(Z_{LE}^+ + Z_{TE}^-)} w_L^+ + \frac{2Z_{TE}^+(c_L^- - s)}{(c_L^- - s)(Z_{LE}^+ + Z_{TE}^-)} w_R^- \\
&\quad + \frac{2s}{(c_L^- - s)(Z_{LE}^+ + Z_{TE}^-)} \left\{ n_x Z_{TE}^+(\hat{a}_L\hat{b}_R - \hat{a}_R\hat{b}_L) + n_y Z_{TE}^+(\hat{a}_L\hat{c}_R - \hat{a}_R\hat{c}_L) + \hat{b}_R\hat{c}_L - \hat{b}_L\hat{c}_R \right\} w_R^0,
\end{align*}
\]
(5.1.23)
and
\[
w_R^0 = \frac{\hat{b}_R n_x + \hat{c}_R n_y}{\hat{b}_L n_x + \hat{c}_L n_y} w_R^+.
\]
(5.1.24)

The left-going wave to the left of the interface \(w_L^-\) is given by a linear combination of the known left-going and right-going wave solutions \(w_L^-, w_R^-\) and \(w_R^0\). The stationary components are scalar multiples at the interface.

We solve the second case, where \(s < 0\), in a similar fashion. The quantities \(w_L^+\) and \(w_L^0\) are known and we solve for the the final unknown wave component
\[
\begin{align*}
w_L^- &= \frac{(c_L^+ - s)(Z_{LE}^+ - Z_{TE}^-)}{(c_L^- - s)(Z_{LE}^+ + Z_{TE}^-)} w_L^+ + \frac{2Z_{TE}^+(c_L^- - s)}{(c_L^- - s)(Z_{LE}^+ + Z_{TE}^-)} w_R^- \\
&\quad + \frac{2s}{(c_L^- - s)(Z_{LE}^+ + Z_{TE}^-)} \left\{ n_x Z_{TE}^+(\hat{a}_L\hat{b}_R - \hat{a}_R\hat{b}_L) + n_y Z_{TE}^+(\hat{a}_L\hat{c}_R - \hat{a}_R\hat{c}_L) + \hat{b}_R\hat{c}_L - \hat{b}_L\hat{c}_R \right\} w_R^0.
\end{align*}
\]
(5.1.25)

With all characteristic wave components to the left of the interface known, we can define the TE upwind numerical flux, \(F^{*,TE}(q_L, q_R; \hat{n}, s)\) from the Rankine-Hugoniot condition. The characteristic variables at the boundary \(w_*\) will depend on the mesh speed \(s\):
1. If $s > 0$, then $w^+_s = w^+_L$, $w^-_s = (5.1.23)$, and $w^0_s = (5.1.24)$.

2. If $s < 0$, then $w^+_s = w^+_L$, $w^-_s = (5.1.25)$, and $w^0_s = w^0_L$.

The TE numerical flux at the interface is given by

$$\mathbf{F}^{*,TE}(\mathbf{q}_L, \mathbf{q}_R; \hat{n}, s) = P_L \Lambda_L \mathbf{w}_s,$$

(5.1.28)

Similar to the procedure giving (5.1.22), we write $P_L \Lambda_L \mathbf{w}_s$ as a system of equations in terms of $\mathbf{w}_s$

$$P_L \Lambda_L \mathbf{w}_s = \frac{1}{2} \begin{bmatrix} c_L^+ - s & - (c_L^- - s) & - 2 s \hat{a}_L \\ n_y Z_L^{TE}(c_L^+ - s) & n_y Z_L^{TE}(c_L^- - s) & - 2 s \hat{b}_L \\ - n_x Z_L^{TE}(c_L^+ - s) & - n_x Z_L^{TE}(c_L^- - s) & - 2 s \hat{c}_L \end{bmatrix} \begin{bmatrix} w^+_s \\ w^-_s \\ w^0_s \end{bmatrix}.$$ 

(5.1.29)

Thus, the numerical flux that calculates the normal flux for the TE problem is

$$\mathbf{F}^{*,TE}(\mathbf{q}_L, \mathbf{q}_R; \hat{n}, s) = \frac{1}{2} \begin{bmatrix} (c_L^+ - s) w^+_s + (c_L^- - s) w^-_s - 2 s \hat{a}_L w^0_s \\ n_y Z_L^{TE} \left\{ (c_L^+ - s) w^+_s - (c_L^- - s) w^-_s \right\} - 2 s \hat{b}_L w^0_s \\ n_x Z_L^{TE} \left\{ - (c_L^+ - s) w^+_s + (c_L^- - s) w^-_s \right\} - 2 s \hat{c}_L w^0_s \end{bmatrix}.$$ 

(5.1.30)

The TE numerical flux (5.1.30) is valid at moving element boundaries in free space, at a moving material interface, or within a moving object. To compute the upwind numerical flux one uses the altered wavespeeds $c^\pm$ (5.1.9), the eigenvectors (5.1.10), (5.1.11), the impedance $Z^{TE}$ (5.1.12), and the characteristic variables given by (5.1.26) or (5.1.27) depending on the sign of the mesh speed $s$.

Next we derive the numerical flux for the TM polarization. Similar to the TE case we use augmented physical flux components $B^{TM}$ (2.2.26) and $C^{TM}$ (2.2.27)

$$\mathcal{F} = \mathbf{F}^{TM} - \mathbf{q} x_t = (B^{TM} - x_t I, C^{TM} - y_t I) \mathbf{q} = (\mathbf{B}^{TM}, \mathbf{C}^{TM}) \mathbf{q},$$

(5.1.31)

where

$$\mathbf{B}^{TM} = \begin{bmatrix} - \nu_x \varpi - x_t & - (\varepsilon L \mu_L \nu_x \nu_y \varpi) / \varepsilon & (1 + \varepsilon L \mu_L \nu^2_y \varpi) / \varepsilon \\ 0 & - x_t & 0 \\ \varepsilon (1 - \beta^2) / \alpha & \nu_y \varpi & - \nu_x \varpi - x_t \end{bmatrix},$$

(5.1.32)

and

$$\mathbf{C}^{TM} = \begin{bmatrix} - \nu_y \varpi - y_t & - (1 + \varepsilon L \mu_L \nu^2_y \varpi) / \varepsilon & (\varepsilon L \mu_L \nu_x \nu_y \varpi) / \varepsilon \\ - \varepsilon (1 - \beta^2) / \alpha & - \nu_y \varpi - y_t & \nu_x \varpi \\ 0 & 0 & - y_t \end{bmatrix}.$$ 

(5.1.33)

We derive an upwind numerical flux by solving the exact Riemann problem for the TM polarization in the same way as the TE polarization:

1. Find the eigendecomposition of the normal flux matrix.
2. Decompose the solution into its characteristic variables.

3. Solve for any unknown characteristic components, which depend on the sign of the mesh speed \( s \).

First we find the eigendecomposition of the normal flux matrix for the TM polarization. To compute the fluxes along the normal direction of element interfaces we find that

\[
\hat{n} \cdot \mathbf{F} = (\mathbf{B}^{TM} n_x + \mathbf{C}^{TM} n_y) \mathbf{q} = \mathbf{A}^{TM} \mathbf{q}.
\]  (5.1.34)

From (5.1.32) and (5.1.33) we define the normal coefficient matrix for the TM problem by

\[
\mathbf{A}^{TM} = \begin{bmatrix}
-N \omega - s & \frac{-\epsilon \mu \mu_L (\epsilon_L^2 n_y + \nu_y \omega \eta N)}{\epsilon} & \frac{\epsilon \mu \mu_L (\epsilon_L^2 n_x + \nu_x \omega \eta N)}{\epsilon} \\
\frac{-n_x}{\epsilon} (1 - \beta^2) & -\nu_y \omega n_y - s & \nu_x \omega n_y \\
\frac{n_y}{\epsilon} (1 - \beta^2) & \nu_y \omega n_x & -\nu_x \omega n_x - s
\end{bmatrix}. \]  (5.1.35)

The normal flux matrix \( \mathbf{A}^{TM} \) is diagonalized in terms of its eigenvectors and eigenvalues,

\[
\mathbf{A}^{TM} = \mathbf{P} \Lambda \mathbf{P}^{-1},
\]  (5.1.36)

where \( \Lambda = \text{diag}(c^+ - s, c^- - s, s) \), \( c^\pm \) are the altered wavespeeds (5.1.9) and the matrix of right eigenvectors

\[
P = \frac{1}{2} \begin{bmatrix}
1 & 1 & -2\hat{a} \\
-n_y Z^{TM} & n_y Z^{TM} & 2\hat{b} \\
n_x Z^{TM} & -n_x Z^{TM} & 2\hat{c}
\end{bmatrix},
\]  (5.1.37)

with \( \mathbf{e}^0 = \begin{bmatrix} \hat{a} & \hat{b} & \hat{c} \end{bmatrix}^T \) given in (5.1.10) and

\[
Z^{TM} = \frac{1}{Y^{TM}} = \sqrt{\frac{\epsilon^2(1 - \beta^2)}{\epsilon \mu (1 - \epsilon \mu_L \eta^2) - \epsilon^2 L \mu_L \eta^2}}. \]  (5.1.38)

We invert (5.1.37) to find the matrix of left eigenvectors

\[
P^{-1} = \frac{1}{bn_x + \hat{c}n_y} \begin{bmatrix}
1 & -\hat{c} Y^{TM} + \hat{n}_x & \hat{b} Y^{TM} + \hat{n}_y \\
1 & \hat{c} Y^{TM} + \hat{n}_x & -\hat{b} Y^{TM} + \hat{n}_y \\
0 & n_x & n_y
\end{bmatrix}. \]  (5.1.39)

We next decouple the solution \( \mathbf{q} \) into characteristic variables \( \mathbf{w} \)

\[
\mathbf{w} = P^{-1} \mathbf{q} = \frac{1}{bn_x + \hat{c}n_y} \begin{bmatrix}
1 & -\hat{c} Y^{TM} + \hat{n}_x & \hat{b} Y^{TM} + \hat{n}_y \\
1 & \hat{c} Y^{TM} + \hat{n}_x & -\hat{b} Y^{TM} + \hat{n}_y \\
0 & n_x & n_y
\end{bmatrix} \begin{bmatrix}
B_z \\
D_x \\
D_y
\end{bmatrix} = \begin{bmatrix}
w^+ \\
w^- \\
w^0
\end{bmatrix}. \]  (5.1.40)

The characteristic variables have a jump discontinuity at the moving dielectric interface. We apply the Rankine-Hugoniot condition (5.1.16) and, assuming \( s \leq |c^\pm| \), arrive at two cases for the characteristic wave components:
For $s > 0$, we find
\[
\begin{align*}
    w^-_L &= \frac{(c_L^+ - s)(Z_L^T - Z_R^T)}{(c_L^+ - s)(Z_L^T + Z_R^T)} w^+_L + \frac{2Z_R^T(c_r^- - s)}{(c_L^+ - s)(Z_L^T + Z_R^T)} w^-_R \\
    &\quad + 2s\{n_xZ_R^T(\tilde{a}_L\hat{b}_R - \tilde{a}_R\hat{b}_L) + n_yZ_R^T(\tilde{a}_L\hat{c}_R - \tilde{a}_R\hat{c}_L) + \hat{b}_L\hat{c}_R - \hat{b}_R\hat{c}_L\} w^0_R,
\end{align*}
\] (5.1.41)
and $w^0_L$ is given by (5.1.24), just as in the TE polarization. In the second case, where $s < 0$, the quantities $w^+_L$ and $w^0_L$ are known and we find unknown wave component to be
\[
\begin{align*}
    w^-_L &= \frac{(c_L^- + s)(Z_L^T - Z_R^T)}{(c_L^- - s)(Z_L^T + Z_R^T)} w^+_L + \frac{2Z_R^T(c_r^+ - s)}{(c_L^- - s)(Z_L^T + Z_R^T)} w^-_R \\
    &\quad + 2s\{n_xZ_R^T(\tilde{a}_L\hat{b}_R - \tilde{a}_R\hat{b}_L) + n_yZ_R^T(\tilde{a}_L\hat{c}_R - \tilde{a}_R\hat{c}_L) + \hat{b}_L\hat{c}_R - \hat{b}_R\hat{c}_L\} w^0_L,
\end{align*}
\] (5.1.42)
We find the upwind numerical flux $F^{*,TM}(q_L, q_R; \hat{n}, s)$ for the TM polarization by applying the Rankine-Hugoniot conditions to the characteristic variables at the boundary $w_*$ given by:
\[
\begin{align*}
    1. & \text{ If } s > 0, \text{ then } w^*_+ = w^+_L, \quad w^-_* = (5.1.41), \text{ and } w^0_* = (5.1.24). \quad (5.1.43) \\
    2. & \text{ If } s < 0, \text{ then } w^*_+ = w^+_L, \quad w^-_* = (5.1.42), \text{ and } w^0_* = w^0_L. \quad (5.1.44)
\end{align*}
\] The TM numerical at the interface is given by
\[
F^{*,TM}(q_L, q_R; \hat{n}, s) = P_L\Lambda_L w_*,
\] (5.1.45)
The TM numerical flux, which is valid at moving element boundaries in free space, at a moving material interface, or within a moving object is
\[
F^{*,TM}(q_L, q_R; \hat{n}, s) = \frac{1}{2} \left[ \begin{array}{c}
    (c_L^+ - s)w^*_+ + (c_L^- - s)w^-_* + 2s\hat{a}w^0_* \\
    + n_xZ_R^T\left\{-(c_L^+ - s)w^*_+ + (c_L^- - s)w^-_*\right\} - 2s\hat{b}_Lw^0_* \\
    - n_xZ_R^T\left\{(c_L^- - s)w^*_+ - (c_L^+ - s)w^-_*\right\} - 2s\hat{c}_Lw^0_* \end{array} \right].
\] (5.1.46)
To compute the upwind numerical flux one uses the altered wavespeeds $c^\pm$ (5.1.9), the eigenvectors (5.1.10), (5.1.11), the impedance $Z^T$ (5.1.38), and the characteristic variables given by (5.1.43) or (5.1.44) depending on the sign of the mesh speed $s$. 

54
5.1.2 The Classical Wave Equation

We also derive the fluxes for the classical wave equation. The wave equation model differs from the Maxwell’s equation model. Recall that the form of the wave equation changes with the frame of reference and includes a local drag component in the laboratory frame $\mathcal{S}$, which explicitly shows how the motion of the medium alters the wave speed.

We follow the same procedure as for the TE and TM polarized Maxwell’s equations model to derive the fully upwind numerical flux. First, we determine the eigendecomposition of the normal flux matrix for the wave equation in a moving medium. We find the coefficient matrix for the normal flux to be

$$A^W = \hat{n} \cdot \mathcal{F}$$

$$= \hat{B}^W n_x + \hat{C}^W n_y$$

$$= \begin{bmatrix} 2abN - s & n_x b(c^2 - a^2) - n_y ab\nu_x \nu_y & n_y b(c^2 - a^2) - n_x ab\nu_x \nu_y \\ n_x & -s & 0 \\ n_y & 0 & -s \end{bmatrix},$$  \hspace{1cm} (5.1.47)

where $\hat{B}^W$, $\hat{C}^W$, $a$, and $b$ are given in (2.2.37). The matrix $A^W$ is diagonalized in terms of its eigenvectors and eigenvalues,

$$A^W = P \Lambda P^{-1}.$$  \hspace{1cm} (5.1.48)

The eigenvalues of the normal flux coefficient matrix (5.1.47) are

$$\lambda_0 = -s, \quad \lambda_{\pm} = c^{\pm} - s,$$  \hspace{1cm} (5.1.49)

where

$$c^{\pm} = abN \pm \sqrt{c^2b + abN^2(ab - 1)}.$$  \hspace{1cm} (5.1.50)

The values of $c^{\pm}$ (5.1.50) represent the altered wave speeds of the wave equation in the moving medium. To complete the eigendecomposition we have the matrices of eigenvectors

$$P = \frac{1}{2} \begin{bmatrix} \frac{1}{\hat{r}} & \frac{1}{\hat{r}} & 0 \\ \frac{n_x}{\hat{r}c^t} & \frac{n_x}{\hat{r}c^t} & 2\hat{s} \end{bmatrix}, \quad \text{and} \quad P^{-1} = \begin{bmatrix} \frac{2c^{\pm} - c^t - c^-}{c^t - c^-} & \frac{2\hat{r}}{c^t - c^-} & \frac{2\hat{s}}{c^t - c^-} \\ -\frac{n_y}{c^{\pm} c^-} & \frac{n_x}{c^{\pm} c^-} & \frac{n_x}{c^{\pm} c^-} \\ -\frac{2\hat{s}}{c^t - c^-} & \frac{2\hat{r}}{c^t - c^-} & \frac{2c^{\pm} - c^t - c^-}{c^t - c^-} \end{bmatrix},$$  \hspace{1cm} (5.1.51)

where $\hat{r} = n_x b(c^2 - a^2) - n_y ab\nu_x \nu_y$, $\hat{s} = n_y b(c^2 - a^2) - n_x ab\nu_x \nu_y$.

Next, we decouple the solution $\mathbf{q}$ into its characteristic wave components $\mathbf{w}$ by

$$\mathbf{w} = P^{-1} \mathbf{q} = \begin{bmatrix} \frac{2c^{\pm} - c^t - c^-}{c^t - c^-} & \frac{2\hat{r}}{c^t - c^-} & \frac{2\hat{s}}{c^t - c^-} \\ -\frac{n_y}{c^t c^-} & \frac{n_x}{c^t c^-} & \frac{n_x}{c^t c^-} \\ 0 & -\frac{2\hat{s}}{c^t - c^-} & \frac{2\hat{r}}{c^t - c^-} \end{bmatrix} \begin{bmatrix} p \\ u \\ v \end{bmatrix} = \begin{bmatrix} w^+ \\ w^- \\ w^0 \end{bmatrix}.$$  \hspace{1cm} (5.1.52)
We assume $s \leq |c^\pm|$, just as in the TE problem, and arrive at two cases for the unknown characteristic wave components. If $s > 0$, we solve for the unknown $w^+_L$ to find

\[
w^+_L = -\frac{c_L^+(c_L^- - s)(c_R^+ - c_R^-)}{c_L^+(c_L^- - s)(c_R^- - c_R^+)} w^+_R + \frac{c_L^-(c_R^- - s)(c_R^- - c_R^+)}{c_R^-(c_L^- - s)(c_L^- - c_R^+)} w^-_R - \frac{2sc_R^+(\hat{r}_L\hat{s}_R - \hat{r}_R\hat{s}_L)}{c_L^+(c_L^- - s)(c_L^- - c_R^+)} w^0_R, \tag{5.1.53}\]

\[
w^0_L = \frac{c_R^- c_R^0}{c_L^+ c_L^0} w^0_R. \tag{5.1.54}\]

If $s < 0$, we find that $w^+_L$ remains the same as (5.1.53) and the component $w^0_L$ is known.

Finally, we define the upwind numerical flux, $F^{*,W}(q_L, q_R; \hat{n}, s)$, for the Riemann problem in a moving material medium. To do so we once again apply the Rankine-Hugoniot conditions to the characteristic variables at the boundary $w_*$ given by:

1. If $s > 0$, then $w^+_* = w^+_L$, $w^-_* = (5.1.53)$, and $w^0_* = (5.1.54)$. \tag{5.1.55}

2. If $s < 0$, then $w^+_* = w^+_L$, $w^-_* = (5.1.53)$, and $w^0_* = w^0_L$. \tag{5.1.56}

The wave equation numerical flux at the interface is given by

\[F^{*,W}(q_L, q_R; \hat{n}, s) = P_L A_L w_* .\tag{5.1.57}\]

The general upwind numerical flux for the wave equation is

\[F^{*,W}(q_L, q_R; \hat{n}, s) = \frac{1}{2} \begin{bmatrix} (c_L^+ - s)w^+_* + (c_L^- - s)w^-_* \\ n_x \left[ \frac{c_L^+-s}{c_L^-} w^+_* + \frac{c_L^-}{c_L^-} w^-_* \right] - 2s\hat{s}_L w^0_* \\ n_y \left[ \frac{c_L^+-s}{c_L^-} w^+_* + \frac{c_L^-}{c_L^-} w^-_* \right] + 2s\hat{r}_L w^0_* \end{bmatrix} . \tag{5.1.58}\]

To compute the numerical flux (5.1.58) one uses the altered wavespeeds for the wave equation $c^\pm$ (5.1.50), the eigenvectors (5.1.51), and the characteristic variables given by (5.1.55) or (5.1.56) depending on the sign of the mesh speed $s$.

**Three Dimensional Wave Equation on a Moving Hexahedral Mesh.** Suppose we have a wave propagation problem in a homogeneous material on a three dimensional domain that contains a moving physical boundary, e.g., a moving cylindrical object. As before, we break the domain into a number of non-overlapping hexahedral subdomains that are allowed to move in response to the object’s motion. What is the appropriate Riemann solver for the flux across the moving faces?
We use the three dimensional wave equation but the discussion is much simpler for this problem because we assume a homogeneous material, i.e., the wavespeed is constant throughout the domain.

We, again, use the constitutive relations (2.2.36) with \(w_t = -p_z\) and rewrite the wave equation as a system of conservation laws

\[
\begin{pmatrix}
p \\ u \\ v \\ w
\end{pmatrix}_t + \begin{pmatrix}
0 & c^2 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
p \\ u \\ v \\ w
\end{pmatrix}_x + \begin{pmatrix}
0 & 0 & c^2 & 0 \\
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
p \\ u \\ v \\ w
\end{pmatrix}_y + \begin{pmatrix}
0 & 0 & 0 & c^2 \\
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
p \\ u \\ v \\ w
\end{pmatrix}_z = 0, \quad (5.1.59)
\]
or in a compact matrix form

\[
q_t + \mathcal{B}q_x + \mathcal{C}q_y + \mathcal{D}q_z = 0. \quad (5.1.60)
\]

Just as before, the flux matrices will incorporate the arbitrary mesh velocity \(x_t = (x_t, y_t, z_t)\) to account for the motion of the mesh. Thus, we have the augmented fluxes

\[
\tilde{\mathcal{B}} = \begin{bmatrix}
-x_t & c^2 & 0 & 0 \\
1 & -x_t & 0 & 0 \\
0 & 0 & -x_t & 0 \\
0 & 0 & 0 & -x_t
\end{bmatrix}, \quad (5.1.61)
\]

\[
\tilde{\mathcal{C}} = \begin{bmatrix}
-y_t & 0 & c^2 & 0 \\
0 & -y_t & 0 & 0 \\
1 & 0 & -y_t & 0 \\
0 & 0 & 0 & -y_t
\end{bmatrix}, \quad (5.1.62)
\]

\[
\tilde{\mathcal{D}} = \begin{bmatrix}
-z_t & 0 & 0 & c^2 \\
0 & -z_t & 0 & 0 \\
0 & 0 & -z_t & 0 \\
1 & 0 & 0 & -z_t
\end{bmatrix}. \quad (5.1.63)
\]

We require the evaluation of the fluxes along the normal direction of the element interfaces

\[
\hat{n} \cdot \mathcal{F} = (\tilde{\mathcal{B}}n_x + \tilde{\mathcal{C}}n_y + \tilde{\mathcal{D}}n_z)q = \mathcal{A}_3^Wq, \quad (5.1.64)
\]

where

\[
\mathcal{A}_3^W = \begin{bmatrix}
-s & n_x c^2 & n_y c^2 & n_z c^2 \\
n_x & -s & 0 & 0 \\
n_y & 0 & -s & 0 \\
n_z & 0 & 0 & -s
\end{bmatrix}, \quad (5.1.65)
\]

where \(s = x_t \cdot \hat{n}\) is the mesh speed. The normal flux matrix is decoupled in terms of its eigenvectors and eigenvalues, i.e., \(\mathcal{A}_3^W = P\Lambda P^{-1}\). The eigenvalues of the matrix \(\mathcal{A}_3^W\) (5.1.65) are

\[
\lambda_0 = -s, \quad \lambda_1 = -s, \quad \lambda_{\pm} = \pm c, \quad (5.1.66)
\]
with corresponding right eigenvectors

\[
P = \begin{bmatrix}
\frac{1}{2} & \frac{1}{2} & 0 & 0 \\
\frac{n_x}{2c} & -\frac{n_y}{2c} & n_y & n_z \\
\frac{n_z}{2c} & -\frac{n_x}{2c} & n_x & 0 \\
\frac{n_y}{2c} & -\frac{n_z}{2c} & 0 & -n_x \\
\end{bmatrix},
\]

(5.1.67)

and left eigenvectors

\[
P^{-1} = \begin{bmatrix}
1 & n_xc & n_yc & n_zc \\
1 & -n_xc & -n_yc & -n_zc \\
0 & n_y & \frac{n_z^2+n_y^2}{n_x} & \frac{n_y n_z}{n_x} \\
0 & n_z & \frac{n_y n_x}{n_z} & \frac{n_z^2+n_y^2}{n_x} \\
\end{bmatrix}.
\]

(5.1.68)

Therefore, we can decouple the system into left-going and right-going wave components with characteristic variables

\[
w = \begin{bmatrix} w^+ \\ w^- \\ w_1^0 \\ w_2^0 \end{bmatrix} = P^{-1} q = \begin{bmatrix} 1 & n_xc & n_yc & n_zc \\
1 & -n_xc & -n_yc & -n_zc \\
0 & n_y & \frac{n_z^2+n_y^2}{n_x} & \frac{n_y n_z}{n_x} \\
0 & n_z & \frac{n_y n_x}{n_z} & \frac{n_z^2+n_y^2}{n_x} \\
\end{bmatrix} \begin{bmatrix} p \\ u \\ v \\ w \end{bmatrix}.
\]

(5.1.69)

If \(|s| < c\), then we know \(w^+\) is computed from the left and \(w^-\) is computed from the right. However, the direction in which the tangential velocity components \(w_1^0\) and \(w_2^0\) travel depends upon the sign of \(s\). If \(s > 0\), then \(w_1^0\) and \(w_2^0\) are computed from the right since as they are left-going waves, otherwise, \(w_1^0\) and \(w_2^0\) are computed from the left.

We assumed that the wave propagation occurs in a homogeneous material, so there is no jump discontinuity in the characteristic variables. Because of the continuity in the characteristic variables, we can derive the Riemann solver using a much simpler method known as flux splitting [56, 90]. Let us consider that \(s > 0\); the other case is solved in a similar way. The ability to diagonalize \(A_{3D}^W\) allows us to split the system into left-going and right-going components. Let

\[
\Lambda = \begin{bmatrix}
c - s & 0 & 0 & 0 \\
0 & -c - s & 0 & 0 \\
0 & 0 & -s & 0 \\
0 & 0 & 0 & -s \\
\end{bmatrix} = \Lambda^+ + \Lambda^-, 
\]

(5.1.70)
be the splitting of the three wave components. Then

$$A_{3D}^W = P\Lambda^+ P^{-1} + P\Lambda^- P^{-1} = A^+ + A^-,$$  \hfill (5.1.71)

splits the matrix $A_{3D}^W$ into components that have right-going and left-going waves with respect to the direction $\hat{n}$.

With the normal matrix splitting (5.1.71) we are equipped to write the Riemann solver in terms of the characteristic wave components

$$F_{3D}^W (q^L, q^R; \hat{n}, s) = A^+ q + A^- q$$

$$= P(\Lambda^+ P^{-1} q + \Lambda^- P^{-1} q)$$

$$= P \begin{bmatrix} (c-s)w_L^+ \\ -(c+s)w_R^- \\ -sw_{1,R}^0 \\ -sw_{2,R}^0 \end{bmatrix}$$

$$= \frac{n_x}{2c} \begin{bmatrix} (c-s)w_L^+ + (c+s)w_R^- - sn_xw_{1,R}^0 - sn_xw_{1,R}^0 \\ 1 \left\{ (c-s)w_L^+ - (c+s)w_R^- \right\} \\ \frac{n_y}{2c} \left\{ (c-s)w_L^+ + (c+s)w_R^- - sn_xw_{2,R}^0 \right\} \\ \frac{n_z}{2c} \left\{ (c-s)w_L^+ + (c+s)w_R^- - sn_xw_{1,R}^0 \right\} \end{bmatrix}.$$  \hfill (5.1.72)

In the matrix of left eigenvectors, $P^{-1}$ (5.1.68) we see $n_x$ in the denominator. If $n_x = 0$ we rewrite the eigendecomposition to avoid a division by zero in the eigenvectors and find

$$P = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{n_y}{2c} & \frac{n_y}{2c} & -n_x & 0 \\ \frac{n_z}{2c} & \frac{n_z}{2c} & -n_x & 0 \end{bmatrix}, \quad P^{-1} = \begin{bmatrix} 1 & 0 & n_y & n_z \\ 0 & 1 & -n_y & -n_z \\ 0 & 0 & n_z & -n_y \\ 0 & 1 & 0 & 0 \end{bmatrix}. \hfill (5.1.73)$$

Following the same procedure we find the numerical flux

$$F_{3D}^W|_{n_x=0} (q^L, q^R; \hat{n}, s) = \begin{bmatrix} \frac{1}{2} \left\{ (c-s)w_L^+ - (c+s)w_R^- \right\} \\ -sw_{2,R}^0 \\ \frac{n_y}{2c} \left\{ (c-s)w_L^+ + (c+s)w_R^- - sn_xw_{1,R}^0 \right\} \\ \frac{n_z}{2c} \left\{ (c-s)w_L^+ + (c+s)w_R^- + sn_yw_{1,R}^0 \right\} \end{bmatrix},$$ \hfill (5.1.74)

where we see that the Riemann solver collapses to the homogeneous material, two dimension numerical flux in the $n_y - n_z$ plane, and the solution in the $x-$direction is simply advected. A similar result is found if we set $n_y = 0$ or $n_z = 0$ in (5.1.72).
5.2 Consistency Checks for the Numerical Fluxes

Next, we show that the numerical fluxes for the Maxwell’s equations (5.1.30) and (5.1.46) and the classical wave equation (5.1.58) reduce to the true normal flux if the states on each side of an element boundary are identical. We also show that the numerical fluxes implement the correct set of boundary conditions at a moving, perfectly reflecting mirror when $c_R = 0$.

5.2.1 Maxwell’s Equations

The numerical fluxes (5.1.30) and (5.1.46) for the TE and TM polarization reduce to the fluxes in the normal direction (5.1.6) and (5.1.35), respectively. For example, with the solution $\mathbf{q}$ the same at the left and right of the interface, $c^+ = c$, $c^- = -c$, and

$$
\mathbf{F}^{*,TE}(\mathbf{q}, \hat{n}, s) = \begin{bmatrix}
-(N\nabla + s)D_z + \frac{\varepsilon \mu L}{\mu} (c_T^2 n_y + \nu_y \nabla N)B_x - \frac{\varepsilon \mu LL}{\mu} (c_T^2 n_x + \nu_x \nabla N)B_y \\
\frac{\mu_n}{\alpha} (1 - \beta^2)D_z - (\nu_y \nabla n_y + s)B_x + \nu_x \nabla n_y B_y \\
-\frac{\mu_n}{\alpha} (1 - \beta^2)D_z + \nu_y \nabla n_x B_x - (\nu_x \nabla n_x + s)B_y
\end{bmatrix}
$$

$$
= \mathbf{A}^{TE} \mathbf{q}.
$$

(5.2.1)

The numerical flux reduces to the moving mesh flux with no material interface [2].

The numerical fluxes for the TE polarization (5.1.30) and the TM polarization (5.1.46) reduce to the correct relativistic boundary conditions if $c_R = 0$. The source-free boundary conditions at a moving mirror are [20]

$$
-(\hat{n} \cdot \mathbf{v}_o)\mathbf{[D]} + \hat{n} \times \mathbf{[H]} = 0, \quad (5.2.2)
$$

$$
-(\hat{n} \cdot \mathbf{v}_o)\mathbf{[B]} + \hat{n} \times \mathbf{[E]} = 0, \quad (5.2.3)
$$

$$
\hat{n} \cdot \mathbf{[D]} = 0, \quad (5.2.4)
$$

$$
\hat{n} \cdot \mathbf{[B]} = 0. \quad (5.2.5)
$$

The brackets represent a jump discontinuity in the electric and magnetic fields defined in terms of the incident ($i$) and reflected ($r$) components by

$$
\mathbf{[E]} = \mathbf{E}^r + \mathbf{E}^i,
$$

$$
\mathbf{[H]} = \mathbf{H}^r - \mathbf{H}^i,
$$

$$
\mathbf{[D]} = \mathbf{D}^r - \mathbf{D}^i,
$$

$$
\mathbf{[B]} = \mathbf{B}^r + \mathbf{B}^i.
$$

(5.2.6)
The first two equations, (5.2.2) and (5.2.3), impose boundary conditions in the tangential direction. The last two, (5.2.4) and (5.2.5), impose boundary conditions in the normal direction.

Setting $c_R = 0$ models the reflection of a wave from a constantly moving, perfectly reflecting mirror. For the perfectly reflecting mirror problem, $w^+_s = w^+_L$, $w^-_s = w^-_L$, and $w^0_s = w^0_L$ in (5.1.15) and $s = N$, since the mesh and dielectric interface must move in lockstep to retain spectral accuracy [56]. Then, we substitute the constitutive relations in free space (2.2.6) and the TE polarization into the numerical flux (5.1.30) to obtain

$$F^*_{TE} = \begin{bmatrix} -ND^1_z + \hat{n} \times \mathbf{H}^1 \hfill \\
-NB^2_z + n_y E^2_z \\
-NB^1_y - n_x E^1_z \end{bmatrix}.$$

(5.2.7)

From (5.2.2), (5.2.3), and (5.2.6) the first two components of the relativistic boundary conditions are enforced by (5.2.7). The incident wave becomes a reflected wave with the correct sign. The third equation (5.2.4) is always true for the TE polarization since $\hat{n} = (n_x, n_y, 0)$ and $D = (0, 0, D_z)$. The fourth relativistic boundary condition (5.2.5) is enforced by the standing wave characteristic $w_0$ in (5.1.15).

From the duality transformation (2.1.2) applied to the TE problem we immediately see that the TM numerical flux (5.1.46) also enforces the relativistic boundary conditions (5.2.2)–(5.2.5).

### 5.2.2 The Classical Wave Equation

We verify that the numerical flux for the wave equation (5.1.58) reduces to the true normal flux (5.1.47) and the correct set of reflecting wall boundary conditions [2] at the moving material interface when $c_R = 0$ in a similar fashion.

The numerical flux for the moving material problem (5.1.58) reduces to the flux in the normal direction (5.1.47) when the states are equal. If we take the solution $\mathbf{q}$ to be the same at the left and right of the interface, then $c^+ = c$, $c^- = -c$, and

$$F^*_{W}(\mathbf{q}, \mathbf{q}; \hat{n}, s) = \begin{bmatrix} (2abN - s)p + (n_x b(c^2 - a\nu_x^2) - n_y ab\nu_x\nu_y)u + (n_y b(c^2 - a\nu_y^2) - n_x ab\nu_x\nu_y)v \\
n_x p - su \\
n_y p - sv \end{bmatrix}$$

$$= \begin{bmatrix} 2abN - s & n_x b(c^2 - a\nu_x^2) - n_y ab\nu_x\nu_y & n_y b(c^2 - a\nu_y^2) - n_x ab\nu_x\nu_y \\
n_x & -s & 0 \\
n_y & 0 & -s \end{bmatrix} \begin{bmatrix} p \\
u \\
v \end{bmatrix}$$

$$= A^W \mathbf{q}.$$

(5.2.8)
Also, (5.1.58) is equivalent to the numerical flux presented in [56, Section 8.4.3] when \( s = 0 \). Similarly, the three dimensional, moving mesh Riemann flux (5.1.72) is consistent with the true normal flux (5.1.65), which is easily verified by

\[
F^{*,W}_{3D}(q; q; \hat{n}, s) = \begin{bmatrix}
-s p + n_x c^2 u + n_y c^2 v + n_z c^2 w \\
n_x p - s u \\
n_y p - s v \\
n_z p - s w
\end{bmatrix}
\begin{bmatrix}
p \\
u \\
v \\
w
\end{bmatrix}
\]

(5.2.9)

We substitute \( c_R = 0, w^+_x = w^+_L, w^-_x = w^-_L, w^0_x = w^0_L \) in (5.1.52) and \( s = N \) into the numerical flux (5.1.58) for the moving wall problem to find

\[
F^{*,W} = \begin{bmatrix}
n_x p - N\{(n_y^2 - n_x^2)u - 2n_x n_y v\} \\
n_y p - N\{2n_x n_y u + (n_x^2 - n_y^2)v\}
\end{bmatrix}
\]

(5.2.11)

The boundary condition for a moving boundary in linear acoustics is the same as if the boundary is at rest [2]. That is, the pressure \( p \) and the normal component of the velocity \( \mathbf{u} = (u, v) \) are continuous across the moving interface. The wall condition implies that the normal velocity is zero at the wall,

\[
\hat{n} \cdot \mathbf{u} = n_x u + n_y v = 0.
\]

(5.2.10)

We substitute \( c_R = 0, w^+_x = w^+_L, w^-_x = w^-_L, w^0_x = w^0_L \) in (5.1.52) and \( s = N \) into the numerical flux (5.1.58) for the moving wall problem to find

\[
F^{*,W} = \begin{bmatrix}
n_x p - N\{(n_y^2 - n_x^2)u - 2n_x n_y v\} \\
n_y p - N\{2n_x n_y u + (n_x^2 - n_y^2)v\}
\end{bmatrix}
\]

(5.2.11)

The numerical flux for the moving wall (5.2.11) simply reflects the pressure \( p \) and imposes the standard external state in \( \mathbf{u} \) for a static wall/reflection boundary [2, 56].
In this chapter we derive an explicit local time stepping (LTS) method to integrate the semi-discrete discontinuous Galerkin spectral element approximation. In Sec. 6.1 we outline an explicit LTS strategy using Adams-Bashforth linear multistep methods. The LTS procedure exploits the minimal coupling between elements in the spatial approximation and is straightforward to implement. The motion of the mesh has little affect on the structure of the LTS method. The ALE mapping $X(\xi, \tau)$ automatically handles the computation of fluxes at moving boundaries. We describe the explicit LTS strategy for a two dimensional problem, with a straightforward extension to three dimensions. Note that the LTS procedure will be valid on static and moving meshes. On a static mesh we can use a standard, constant coefficient Adams-Bashforth method. However, a moving mesh LTS method will need a variable coefficient time integration technique, which we derive in Sec. 6.2. In Sec. 6.3, we present pseudocode for the LTS method on static or moving meshes to integrate the solution, GCL, and spring-mass dashpot system from $t^n$ to $t^{n+1}$. Finally, Sec. 6.4 outlines the storage complexity of the LTS algorithm as well as predicts the possible speed-up in computation time.

### 6.1 Derivation of a Local Time Stepping Procedure for the DGSEM

The LTS Adams-Bashforth time integrator that we use is a simplification of a fastest first multirate integrator [34, 85]. To reduce complexity, we assume (i) the time scales have intermittent synchronization levels, (ii) the elements of the mesh are grouped so there are $m + 1$ time scales, as opposed to each element having its own, and (iii) the time scales differ from one another by integer factors. A further simplification is allowed by the DGSEM spatial discretization itself because elements are coupled only through the boundary flux from immediate neighbors. Once an element
evolves in time with its stable time step, it does not require additional information from the small
time scales. Thus, the coupling between slow and fast time scales is also weak.

To describe a multirate time integration method, we consider the coupled ODE system \( \dot{y} = F(t, y) \). For simplicity of the discussion, we assume there are two time scales, as in [17, 85], denoted as fast and slow. We can write the right hand side \( F(t, y) \) for a general problem with strong coupling between fast and slow time scales as

\[
F(t, y) = \begin{bmatrix}
F_{ff} & F_{fs} \\
F_{sf} & F_{ss}
\end{bmatrix},
\]

(6.1.1)

and the state vector

\[
y(t, y) = \begin{bmatrix}
y_f \\
y_s
\end{bmatrix},
\]

(6.1.2)

where the subscript \( f \) denotes the fast time scale and \( s \) denotes the slow time scale. For (4.2.3) the analogy is that \( y_s \), the solution on large elements, can run on a large step size \( \Delta t_s \) whereas \( y_f \), the solution on small elements, has to run with a much smaller step size \( \Delta t_f \).

Formally, the diagonal terms \( F_{ff} \) and \( F_{ss} \) in (6.1.1) can be calculated immediately since the time scales match. However, to calculate cross terms (coupling) requires extrapolation for the fast to slow time scale calculation \( F_{fs} \), and interpolation for the slow to fast time scale calculation \( F_{sf} \). The difficulty comes from how to calculate the fast to slow time scale coupling accurately, since extrapolation is inherently unreliable. Fortunately, the weak temporal coupling between elements in the DGSEM causes \( F_{fs} \) to vanish, leaving

\[
F(t, y) = \begin{bmatrix}
F_{ff} & 0 \\
F_{sf} & F_{ss}
\end{bmatrix}.
\]

(6.1.3)

The LTS method for the DGSEM needs only to construct time interpolants to compute the solution at intermediate times in the fast time scale.

We limit the number of time scales in our LTS integrator so that the construction of interpolants in time does not become so expensive as to negate speedup. Extra work is required to create interpolants between different time scales and evaluate the coupling terms. As such, we classify the elements of the mesh according to size into \( m + 1 \) groups. The smallest elements are placed in Group \( m \), the next smallest elements are placed in Group \( m - 1 \), and so on. The largest elements are in Group 0. The LTS strategy that we derive allows each group of elements to evolve at its largest stable time step.
To group the elements, we assume that the quadrilateral mesh is generated using a quadtree with 2-refinement [81]. The quadtree method refines the mesh by subdividing elements using quadrilateral templates. Thus, each subsequent group of elements is approximately a factor of two smaller than the previous group. For example, elements in Group 1 are approximately half the size of elements in Group 0. If a quadrilateral mesh is not created using a quadtree with 2-refinement, say with Q-morph [89], it is still advantageous to group the elements, since a small number of groups significantly reduces the number of interpolants in time that must be constructed. It is always possible to look at the distribution of elements according to a measure of size, e.g., area. Once the distribution is created one can decide on the number of groups, \( m \). The grouping strategy is inherently ad hoc depending on the size measure one uses and how one decides to divide the distribution of elements. Also, the time scales will differ by general integer factors rather than factors of two, but otherwise the time stepping algorithm can still be used.

After the elements are grouped according to size, the algorithm finds the largest stable time step for each group. First, the stable time step in Group 0 (the largest elements), called \( \Delta t_0 \), is found from a time step restriction similar to those given in [16, 33]

\[
\Delta t_0 = \frac{1}{|\lambda_{\text{max}}|} \left( \frac{\Delta x}{2N + 1} \right),
\]

where \( N \) is the order of the approximation, \( \Delta x \) is the average side length of elements in Group 0, and \( \lambda_{\text{max}} \) is the maximum eigenvalue of the operator \( L^p_N \) in (4.1.24). The time step in the other groups is determined with

\[
\Delta t_k = \frac{\Delta t_0}{2^k}, \quad k = 1, \ldots, m.
\]

The time step selection strategy in (6.1.5) forces all of the elements to synchronize at intermediate times during the integration. We adopt the notation that the synchronization levels are

\[
t^n = n\Delta t_0,
\]

as shown in Fig. 6.1.

The algorithm cannot integrate the system (4.2.3) at the initial synchronization level \( t^0 = 0 \) because Adams-Bashforth methods are not self-starting. So it uses a global time step \( r + 1 \) order Runge-Kutta method to integrate from \( t^0 = 0 \) to \( t^r = r\Delta t_0 \). The Runge-Kutta method creates the necessary solution history on each element. We use a Runge-Kutta method of one order higher than
the Adams-Bashforth method to ensure that errors in the temporal approximation are dominated by those introduced by the LTS method. Once the integration has been started by an appropriate Runge-Kutta method, the solution, GCL, and spring-mass dashpot system are at a synchronization level.

To provide a concrete discussion of the LTS method we assume that the base time integrator is a third order Adams-Bashforth method for the remainder of this section. It is straightforward to extend the LTS method to a higher temporal order if desired.

We outline the procedure to integrate the system (4.2.3) from one synchronization level, \( t^n \), to the next, \( t^{n+1} \). To augment an ALE-DGSEM implementation to include the LTS method we require two extra procedures: (i) an evolve condition and (ii) the construction of the polynomial interpolants in time for the solution and GCL. We give detailed descriptions of these two procedures later in this section.

At the synchronization level \( t^n \) the boundary flux data necessary to evolve every element in time is available, so we sweep through the entire mesh and evolve each element one local time step as shown in Fig. 6.1(b). After the global time evolution from a synchronization level, the LTS method relies on an evolve condition to ensure that an element is ready to step forward in time.

One can define an evolve condition in different ways. Gassner et al. [33] use an evolve condition that checks the future time level of edge-connected neighbors to guarantee consistent edge flux calculations. However, for the LTS algorithm to operate on moving meshes using the ALE-DGSEM,
an evolve condition that updates the mesh position and velocities at the smallest time scale is needed. Here, we define a local time $t^*$ that is intermediate between the two synchronization levels $t^n$ and $t^{n+1}$, that is, $t^* \in [t^n, t^{n+1}]$. Then the evolve condition is: *if the local time on element $k$ is equal to $t^*$, then the element is ready to evolve one local time step.* The algorithm also checks to see if the time on element $k$ is equal to $t^{n+1}$, since this means that the solution on element $k$ is at the next synchronization level and should not evolve. The algorithm continues to apply the evolve condition to each element until it reaches the next synchronization level, as is shown in Fig. 6.2(a)–6.2(d).

![Figure 6.2: (a)–(c) The progression of repeated application of the evolve condition to a three element mesh. Elements ready to evolve are shaded. (d) Each element has reached the next synchronization level.](image)

The evolve condition just described guarantees that all the neighbor boundary flux data can be computed on element $k$. However, the solution on an element and the solution on the element’s neighbors may be at different times, as we see in Fig. 6.2(a) with $e_1$ and $e_2$. We describe the procedure to calculate the boundary flux on the left interface of $e_2$, but the process is the same for the flux calculation at any of the element interfaces.

To compute the numerical fluxes, $\tilde{F}^*, \tilde{G}^*$ in (4.1.24) or $\tilde{E}^*, \tilde{F}^*$ in (4.1.28) at the intermediate time, the algorithm must construct a solution on the boundary. We show the procedure to construct the intermediate value in time in Fig. 6.3(a)–(c). To calculate the boundary flux at the left interface of $e_2$, the algorithm gets the solution from the left using the polynomial interpolant in time. The solution from the right is found from the known value. The left and right solutions are sent to the numerical fluxes to complete the calculation.

The algorithm uses the three values of the solution history on the boundary to construct a polynomial interpolant in time as required by the third order Adams-Bashforth method time in-
The implementation stores a history of solution values on the boundary at two previous time steps, $t_{n+1}^{n+1} - \Delta t_1$ and $t_{n+1}^{n+1} - 2\Delta t_1$, where $\Delta t_1$ is the time step on $e_1$. This polynomial interpolant enables the algorithm to calculate the solution value on the boundary at any intermediate time in the interval $[t_{n+1}^{n+1} - 2\Delta t_1, t_{n+1}^{n+1}]$. The calculation of the boundary fluxes for the fast time evolving element $e_2$ in Fig. 6.2 occurs after the global evolution at the synchronization level. The implementation also stores a history of the Jacobian values, $\tilde{J}$, at two previous time steps on the boundaries. The algorithm uses the same process to construct a polynomial interpolant in time for Jacobian values as for the solution.

Note that if two neighboring elements are in the same group, the algorithm does not need to construct a time interpolant. All of the boundary information in the two neighbors is at the same time, so the numerical flux is calculated as with a global time stepping method. In the implementation, the algorithm checks if an element and its neighbors are at the same time to avoid the creation of unnecessary polynomial interpolants. This check reduces the computational cost of the AB3LTS time integrator.

The evolve condition and construct time interpolant procedures can be used on static or moving meshes, but when a mesh is moved it is possible for elements to stretch or shrink. In the development
of the LTS algorithm we made the assumption that elements are placed into \( m + 1 \) groups. The worst case for a moving mesh is that an element shrinks so that it should move from its current group, \( j \), to the next smallest group, \( j + 1 \). In that case, the time step on the shrinking element will no longer be small enough to maintain stability. Conversely, if an element stretches enough, then it should move from its current group to the next largest group, \( j - 1 \). The reclassification of an element’s size ensures that the element uses a stable time step as it changes shape, but prevents the algorithm from over-resolving the element in time.

If an element is reclassified from one group to another during the integration, the time derivative histories for the solution, \( \tilde{Q} \), and the Jacobian, \( \tilde{J} \), are no longer in the correct time scale. The same is true of the history of the boundary values used to create time interpolants and compute boundary fluxes. The discrepancy in time scale introduces an \( O(\Delta t) \) error in the temporal approximation. We remedy this by introducing a variable coefficient Adams-Bashforth time integrator.

### 6.2 Variable Coefficient Adams-Bashforth Time Integrator

We derive a variable coefficient Adams-Bashforth method to retain full time accuracy for moving meshes where an element’s size may be reclassified at any time. We use a Lagrange polynomial to determine the coefficients, which inherently includes a variable step size factor.

The variable coefficient Adams-Bashforth method is derived for the general ODE

\[
\frac{dy}{dt} = y' = f(t, y). \tag{6.2.1}
\]

For an \( r \)th order method the right hand side \( f(t, y) \) of (6.2.1) is approximated by a Lagrange interpolating polynomial of the form

\[
p_r(t) = \sum_{j=0}^{r-1} \ell_j(t) f_{n-j}, \quad \text{where} \quad \ell_j(t) = \prod_{i=0 \atop i \neq j}^{r-1} \frac{t - t_{n-i}}{t_{n-j} - t_{n-i}}. \tag{6.2.2}
\]

To determine the general, variable coefficients, we integrate (6.2.1) from \( t_n \) to \( t_{n+1} \) and obtain

\[
y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} p_r(s) \, ds = y_n + \sum_{j=0}^{r-1} \beta_j f_{n-j}, \quad \text{with} \quad \beta_j = \int_{t_n}^{t_{n+1}} \ell_j(s) \, ds. \tag{6.2.3}
\]

For example, a third order method can be derived by expanding (6.2.2) to write the third order polynomial

\[
p_3(t) = \frac{(t - t_n)(t - t_{n-1})}{\Delta t_{n-1}(\Delta t_n + \Delta t_{n-1})} f_{n-2} - \frac{(t - t_{n-2})(t - t_n)}{\Delta t_n \Delta t_{n-1}} f_{n-1} + \frac{(t - t_{n-2})(t - t_{n-1})}{\Delta t_n(\Delta t_n + \Delta t_{n-1})} f_n, \tag{6.2.4}
\]
where \( \Delta t_n = t_n - t_{n-1} \), \( \Delta t_{n-1} = t_{n-1} - t_{n-2} \), and \( t_n - t_{n-2} = \Delta t_n + \Delta t_{n-1} \). We use the substitution 
\[
u = (t_{n+1} - s)/\Delta t_{n+1}
\]
to determine \( \beta_j \) in (6.2.3) and find the third order variable coefficient Adams-Bashforth method
\[
y_{n+1} = y_n + \frac{\Delta t_{n+1}}{6} \left[ r_1 r_0 \rho (2r_1 + 3)f_{n-2} - r_1 r_0 \left( \frac{3}{\rho} + 2r_1 \right) f_{n-1} + (2\rho r_1^2 + 3r_1(\rho + 1) + 6)f_n \right],
\]
where
\[
r_1 = \frac{\Delta t_{n+1}}{\Delta t_n}, \quad r_0 = \frac{\Delta t_n}{\Delta t_{n-1}}, \quad \rho = \frac{r_0}{r_0 + 1}.
\]
For consistency when \( \Delta t = \Delta t_{n+1} = \Delta t_n = \Delta t_{n-1}, r_1 = r_0 = 1, \rho = 1/2, \) we recover the constant coefficient, third order Adams-Bashforth method
\[
y_{n+1} = y_n + \Delta t \left[ \frac{5}{12} f_{n-2} - \frac{4}{3} f_{n-1} + \frac{23}{12} f_n \right].
\]
The variable coefficient Adams-Bashforth method is \( r \)th order accurate. In general, the local truncation error for the variable coefficient Adams-Bashforth family of methods is
\[
\tau_{n+1}(\Delta t_{n-r+2}, \ldots, \Delta t_{n+r-2}) = \frac{1}{\Delta t_{n+r-2}} \int_{t_n}^{t_{n+1}} \frac{f^{(r)}(\xi, y(\xi))}{r!}(t - t_n)(t - t_{n-1}) \cdots (t - t_{n-r+1}) dt,
\]
where \( r \) is the order of the method and \( \xi \in [t_n, t_{n+1}] \). For the third order method (6.2.5), \( r = 3 \) and (6.2.8) becomes
\[
\tau_{n+1}(\Delta t_{n-1}, \Delta t_n, \Delta t_{n+1}) = \frac{1}{\Delta t_{n+1}} \int_{t_n}^{t_{n+1}} \frac{f^{(3)}(\xi, y(\xi))}{3!}(t - t_n)(t - t_{n-1})(t - t_{n-2}) dt.
\]
The substitution \( u = (t_{n+1} - t)/\Delta t_{n+1} \) yields the local truncation error
\[
\tau_{n+1} = \frac{y^{(4)}(\xi)}{72} \left( 3\Delta t_{n+1}^3 + 8\Delta t_{n+1}^2 \Delta t_n + 4\Delta t_{n+1}^2 \Delta t_{n-1} + 6\Delta t_{n+1} \Delta t_n^2 + 6\Delta t_{n+1} \Delta t_n \Delta t_{n-1} \right),
\]
where we used the fact that \( y' = f \), so \( f^{(3)} = y^{(4)} \).
It remains to show that the variable coefficient explicit time integrator (6.2.5) retains third order temporal accuracy in each time scale. We show that the temporal approximation is third order in the largest possible time scale \( \Delta t_0 \). A similar process can be applied to show that it is third order in each of the others. From the time step selection strategy, (6.1.5), we define some real constants \( a, b, \) and \( c \) such that
\[
\Delta t_{n+1} = a\Delta t_0, \quad \Delta t_n = b\Delta t_0, \quad \Delta t_{n-1} = c\Delta t_0.
\]
We note that, by design, the mesh motion will not change the shape of an element abruptly. It will never be the case that, for example, one of the largest elements will become one of the smallest elements in a single time step. A worst case is that an element shrinks from Group 0 to Group 2 in three time steps where the ratio of $a$ to $c$ is 4, $a$ to $b$ is 2, and $b$ to $c$ is 2. In practice, we can expect the ratio between the constants $a$, $b$, and $c$ to be near 1.

From the relationships (6.2.11), we see that the local truncation error in the largest time scale is

$$
\tau(\Delta t_0) = C y^{(4)}(\xi) \Delta t_0^3, \quad \xi \in [t_n, t_{n+1}],
$$

(6.2.12)

where

$$
C = \frac{1}{72} \left( 3c^3 + 8a^2b + 4a^2c + 6ab^2 + 6abc \right).
$$

(6.2.13)

The constant $C$ is bounded, and is usually close to the constant of the traditional third order Adams-Bashforth, 3/8. We verify by numerical example that the temporal accuracy in each time scale of the variable coefficient AB3LTS time integrator is third order in Sec. 8.2.1.

### 6.3 Outline of the LTS Algorithm

We present general pseudocode in Algorithm 1 (StepByABLTS) that outlines the integration of the solution from a synchronization level $t^n$ to the next synchronization level $t^{n+1}$. Important features of the LTS procedure are:

- It is independent of the set of equations to be solved, linear or non-linear. All one needs to provide are procedures to compute the physical and numerical fluxes.
- It is readily applied to three dimensional computations, where the time interpolants are along adjacent faces rather than edges. We verify a three dimensional implementation of the AB3LTS time integrator in Sec. 8.2.1.
- It can be any temporal order provided one extends the solution histories and uses an appropriate variable coefficient time integrator of the form (6.2.3).

The procedure requires as input the elements, the time for the next synchronization level $t^{n+1}$, a Boolean variable MOVINGMESH, the number of element groups $m$, and the time step for the group of smallest elements $\Delta t_m$. Initially, the integration is at a synchronization level, so we set a Boolean variable SYNCH. The algorithm visits every element in the mesh and progresses the solution and
GCL if the element satisfies the evolve condition at $t^\ast$. The construction of any necessary time interpolants occurs in “Assemble time derivatives.” The phrasing “Update solution $\tilde{Q}_k$ and Jacobian $\tilde{J}_k$” is purposefully vague. If the mesh is static the constant coefficient Adams-Bashforth is used. If the mesh is moving, a variable coefficient Adams-Bashforth like (6.2.5) replaces it. An array of Boolean variables keeps track of which elements have evolved. Finally, if an element has been evolved, the time derivative and boundary solution histories are updated.

The differences between the static and the moving mesh implementations are minimal: (i) “Assemble time derivatives” includes an update of the element geometry to the current time $t^\ast$. This update includes the recalculation of a moving element’s size using some measure, (ii) a variable coefficient time integrator, and (iii) the spring-mass dashpot system is integrated at the smallest time scale $\Delta t_m$.

**Algorithm 1**: (StepByABLTS) Integrate from $t^n$ to $t^{n+1}$ with local time stepping

```plaintext
Procedure StepByABLTS
Input: Elements $\{e_k\}_{k=1}^K$, MOVINGMESH, $t^{n+1}$, $m$, $\Delta t_m$ // time step in Group $m$

synch ← TRUE // integration is at a synchronization level
for $j = 2^m$ to 1 decrement 1 do
  // loop bounds depend on grouping strategy
  \{evolved\}$_{k=1}^K$ ← FALSE
  $t^\ast$ ← $t^{n+1} - j\Delta t_m$
  for $k = 1$ to $K$ do
    READYTOEVOLVE ← EvolveCondition(Local time on $e_k$, $t^\ast$)
    if synch or READYTOEVOLVE then
      EVOLVED$_k$ ← TRUE
      Assemble time derivatives $\tilde{Q}_k$ and $\tilde{J}_k$
      Update solution $\tilde{Q}_k$ and Jacobian $\tilde{J}_k$ one local time step on $e_k$
    synch ← FALSE
  if MOVINGMESH then
    Update spring-mass dashpot system with $\Delta t_m$
  for $k = 1$ to $K$ do
    if EVOLVED$_k$ then
      Update time derivative and boundary solution histories on $e_k$
  End Procedure StepByABLTS
```

72
6.4 Storage and Computational Cost of the LTS Method

The ability to perform local time stepping comes at the cost of higher storage. We will compare the storage and computational cost requirements of the AB3LTS against the third order low-storage explicit Runge-Kutta (LSRK3) scheme of Williamson [94]. Thus, we compare the AB3LTS to a Runge-Kutta time integrator already optimized to be low storage.

6.4.1 Storage Requirements of the LTS Method Compared to LSRK3

In two dimensions, the storage complexity for LSRK3 is \(2(N + 1)(3N + 11)\) per equation. It stores six \((N + 1)^2\) arrays and 16 \((N + 1)\) arrays on each element for each equation to be solved. The six \((N + 1)^2\) arrays hold the solution \(\tilde{Q}\), the time dependent Jacobian \(\tilde{J}\), the time derivatives \(\dot{\tilde{Q}}, \dot{\tilde{J}}\), and an intermediate storage array \(G\). In addition, each edge of the quadrilateral requires four \((N + 1)\) arrays to store the solution \(\tilde{Q}\), the Jacobian \(\tilde{J}\), and the numerical fluxes on the boundary, totaling 16 arrays.

The storage complexity for AB3LTS in two dimensions is \(8(N + 1)(N + 5)\) per equation. It requires the time derivative at previous times, two for third order. Because elements no longer evolve at the same rate, the algorithm cannot simply recalculate \(\dot{Q}\) at the previous times. The implementation therefore stores a history of the time derivatives \(\dot{Q}\) and \(\dot{J}\) on \(e_i\) to avoid the introduction of an \(O(\Delta t)\) error. The AB3LTS scheme stores eight \((N + 1)^2\) arrays and 32 \((N + 1)\) arrays for each equation. The eight \((N + 1)^2\) arrays store the solution \(\tilde{Q}\), the time dependent Jacobian \(\tilde{J}\), and the time derivatives \(\dot{Q}\) and \(\dot{J}\) at the three times, \(t^n_i, t^n_i - \Delta t_i, t^n_i - 2\Delta t_i\). Because the coupling is weak in the DGSEM, we do not need to store a history of the solution \(\tilde{Q}\) in the element. Each edge of the quadrilateral requires eight \((N + 1)\) arrays that store the solution \(\tilde{Q}\) and Jacobian \(\tilde{J}\) on the boundary at the three times, \(t^n_i, t^n_i - \Delta t_i, t^n_i - 2\Delta t_i\), as well as the numerical fluxes on the boundary at time \(t^n_i\), totaling 32 arrays.

The memory requirements for the local time stepping method scales well as \(N\) increases. Table 6.1 presents the estimated factor of increase in storage for \(N = 4, \ldots, 10\). The storage size for these commonly used values of \(N\) increases by approximately 50 percent. As \(N \to \infty\), the storage of AB3LTS is 33 percent larger than LSRK3 for two dimensional problems.

The explicit local time stepping strategy remains an attractive option for three dimensional computations, again from a memory standpoint; the storage grows in \(N\) identically for two dimen-
sional or three dimensional problems. Following the same analysis as above, the storage complexity of LSRK3 and AB3LTS for a three dimensional problem is, ignoring $O(1)$ terms,

\[
\begin{align*}
\text{LSRK3} & : 2(N + 1)^2(3N + 11) \text{ per equation}, \\
\text{AB3LTS} & : 8(N + 1)^2(N + 5) \text{ per equation}.
\end{align*}
\]

(6.4.1)

So (6.4.1) shows that the storage of AB3LTS is 33 percent larger than LSRK3 for three dimensional problems, too, as $N \to \infty$.

Table 6.1: Estimated factor of increase in storage complexity for AB3LTS versus global time stepping LSRK3 for various $N$.

<table>
<thead>
<tr>
<th>$N$</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor</td>
<td>1.57</td>
<td>1.54</td>
<td>1.52</td>
<td>1.50</td>
<td>1.49</td>
<td>1.47</td>
<td>1.46</td>
</tr>
</tbody>
</table>

6.4.2 Reducing Mesh Induced Stiffness: Theoretical Predictions

We also predict how the AB3LTS algorithm can reduce the computational work for static and moving meshes with local refinement. Again we compare the AB3LTS method with LSRK3. For each time integrator we assume that the calculation of the time derivative $\dot{\mathbf{Q}}$ dominates the computation. The work estimate for each time integrator uses the number of times $\dot{\mathbf{Q}}$ must be calculated to progress the solution from one synchronization level to the next. We include the recalculation of the element geometry as part of the assembly of the time derivative $\dot{\mathbf{Q}}$ to simplify the discussion of the work estimates.

The work estimate for AB3LTS is

\[
\text{AB3LTS} : \sum_{k=0}^{m} 2^k G(k),
\]

(6.4.2)

where $G(k)$ is the number of elements in the $k^{\text{th}}$ group. Each group of elements has its own time step $\Delta t_0, \ldots, \Delta t_m$ in the AB3LTS algorithm. The LTS method stores a history of the time derivative $\dot{\mathbf{Q}}$ at two previous times on each element. On any element in the mesh it calculates $\dot{\mathbf{Q}}$ once to evolve one local time step. So, progress of the solution on the largest elements, Group 0, from $t^n$ to $t^{n+1}$ requires one calculation of $\dot{\mathbf{Q}}$. From (6.1.5), we know that

\[
t^{n+1} = t^n + \Delta t_0 = t^n + 2\Delta t_1 = \cdots = t^n + 2^m \Delta t_m.
\]

(6.4.3)
Thus, the elements in Group 1 require two calculations of $\dot{\mathbf{Q}}$ to progress the solution from $t^n$ to $t^{n+1}$. If we continue this argument for the number of $\dot{\mathbf{Q}}$ calculations in each group of elements, we find the work estimate (6.4.2).

The work estimate for LSRK3 is

$$\text{LSRK3} : 3K2^{m-1}.$$  \hspace{1cm} (6.4.4)

To derive it, we first denote the time step for the global time stepping LSRK3 as $\Delta t_{RK}$. The region of absolute stability for third order Runge-Kutta is significantly larger than that of the third order Adams-Bashforth. Also, we take

$$\Delta t_{RK} = \frac{\Delta t_0}{2^{m-1}},$$  \hspace{1cm} (6.4.5)

where $m$ is the group of the smallest elements in the mesh. The LSRK3 time integrator must calculate the time derivative $\dot{\mathbf{Q}}$ three times to progress the solution from $t^n$ to $t^n + \Delta t_{RK}$. From (6.4.5),

$$t^{n+1} = t^n + 2^{m-1}\Delta t_{RK},$$  \hspace{1cm} (6.4.6)

and we arrive at the work estimate (6.4.4).

The two time integrators perform the time derivative calculation differently. The work estimate for AB3LTS (6.4.2) does not take into account the extra computational work required by the local time stepping method to create the polynomial interpolant in time described in Sec. 6.1. We will see in the numerical verification, Sec. 8.2.2, that this discrepancy can show up as an overestimate of the reduction in computation time.

To study the speedup, we generate four test meshes with a two dimensional spectral element mesh generator. The four meshes, shown in Fig. 6.4, are (i) an inner circular boundary and refinement center, (ii) nested circles, (iii) an array of circles, and (iv) a $[-20,20] \times [-20,20]$ domain with the NACA0012 airfoil placed at the origin. We report the predictions in Table 6.2. The estimates reinforce intuition: if a mesh has a high proportion of “large” elements to “small” elements then the predicted speedup is greater.

The number of elements, number of time scales, and distribution of the elements affect the predicted speedup. Histograms of the element size distributions for the four test meshes are given in Fig. 6.5 to help understand the predicted speedup. The “Circle Array” mesh provides an example
where a couple “bad” elements introduce severe time step restrictions for global time stepping methods to remain stable. Similarly, as the mesh becomes more locally refined, the problem becomes stiffer for a global time stepping algorithm. For example, the “Circle in Circle” (Fig. 6.4(b)) mesh has a distribution of element sizes slightly skewed to the left (towards large elements).
Figure 6.5: Histograms depict the distribution of element sizes for four test meshes.

and a factor of 2.8 predicted speedup with its five levels of refinement. The mesh distribution of the NACA0012 airfoil (Fig. 6.4(d)) is also slightly skewed to the left. But it has nine levels of refinement to resolve the region around the airfoil. We predict a larger speedup on the NACA0012
mesh than the “Circle in Circle” mesh when we use AB3LTS even though the two meshes have a
similar distribution of element sizes.

Table 6.2: Estimations of work reduction for AB3LTS versus LSRK3 on four meshes.

<table>
<thead>
<tr>
<th>Mesh</th>
<th>( \hat{Q} ) evals LSRK3</th>
<th>( \hat{Q} ) evals AB3LTS</th>
<th>Predicted speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Box Around Circle Refined</td>
<td>2520</td>
<td>1629</td>
<td>1.6</td>
</tr>
<tr>
<td>Circle in Circle</td>
<td>6528</td>
<td>2326</td>
<td>2.8</td>
</tr>
<tr>
<td>Circle Array</td>
<td>18912</td>
<td>1780</td>
<td>10.6</td>
</tr>
<tr>
<td>NACA0012 Airfoil</td>
<td>163584</td>
<td>17366</td>
<td>9.4</td>
</tr>
</tbody>
</table>

6.4.3 For What Kinds of Meshes is LTS Appropriate?

We can use the results of the previous section to find what kinds of meshes are most appropriate
for use with LTS. One would expect that meshes with few small elements will show the best
performance, whereas for meshes with mostly small elements LTS should show little advantage.
These expectations are borne out by analysis.

The work estimates for AB3LTS (6.4.2) and LSRK3 (6.4.4) allow us to derive a theoretical
speedup

\[
S = \frac{1}{2^3 \sum_{k=0}^{m} 2^{k-m} \hat{G}_k},
\]

where \( \hat{G}_k \) is the fraction of the total number of elements in the \( k^{th} \) group. For a uniform distribution
of element sizes with

\[
\hat{G}_k = \frac{1}{m+1},
\]

the speedup \( S \) (6.4.7) is

\[
S = (m+1) \frac{2^m}{2^{m+1} - 1}.
\]

Asymptotically, as \( m \to \infty \), i.e., the number of levels becomes large,

\[
S \sim \frac{m+1}{2}.
\]

For example, nine levels of refinement should have a speedup of about five. An example of a mesh
with mostly small elements is a distribution

\[
\hat{G}_k = \frac{2^k}{\sum_{n=0}^{m} 2^n}.
\]
Its speedup would be
\[ S = \frac{2^m \sum_{n=0}^{m} 2^n}{2^{m+1} - 1} \to 1 \quad \text{as} \quad m \to \infty. \quad (6.4.12) \]

If \( m = 9 \) then \( S = 1.006 \). Thus, if the bulk of the elements are in the smallest group, the LTS procedure becomes ineffective, as one might expect. Finally, if the mesh is weighted towards large elements, e.g.,
\[ \hat{G}_k = \frac{2^{-k}}{\sum_{n=0}^{m} 2^{-n}}, \quad (6.4.13) \]
we expect significant speedups. In fact,
\[ S = \frac{2^m}{m+1} \sum_{n=0}^{m} 2^{-n} = \frac{2^{m+1} - 1}{m+1}, \quad (6.4.14) \]
and as \( m \) gets large,
\[ S \sim \frac{2^{m+1}}{m+1}. \quad (6.4.15) \]

These examples confirm the expectation that the LTS procedure is most useful when the fraction of small elements drops off rapidly.
CHAPTER 7
EFFICIENT IMPLEMENTATION:
PARALLELIZATION OF DGSEM AND LTS

In this chapter we describe a practical framework for a parallel, computationally efficient implementation of the DGSEM. As we described in Chap. 4, the discontinuous Galerkin family of methods are robust for wave propagation problems [3], operate on unstructured meshes [56], easily handle boundary conditions through the solution of a Riemann problem [56, 90], and, most importantly, are weakly coupled through neighboring edges (faces in three dimensions) [12, 43, 56]. We shall see that it is the weak coupling of elements in the DGSEM that make it an ideal candidate for large scale computations. In Sec. 7.1 we outline some necessary background information and context for high performance computing and the DGSEM. In Sec. 7.2 we provide an overview of the parallelization of the DGSEM with a traditional global time stepping integrator. Finally, in Sec. 7.3 we describe the procedure one could use to parallelize the local time stepping integrator derived in Chap. 6.

7.1 Background for Parallel DGSEM

For the computational solution of large scale problems like those found in geophysics [32], electromagnetics [58], or aeroacoustics [77], efficient numerical methods are a key requirement. Processor hardware of today has made a transition to CPUs with many-cores, thus an efficient parallelization for large numbers of processor cores is necessary. A promising candidate for efficient parallelization are numerical methods that are *embarrassingly parallel*. An embarrassingly parallel method has two important features:

1. Little or no effort is required to separate the problem into a number of parallel tasks.
2. Require little or no communication of results between tasks.

With this precise definition we verify that the explicit DGSEM is an embarrassingly parallel numerical method. The weak coupling at adjacent neighbor boundaries makes for little communication between elements. Because the coupling in only at edges (faces in three dimensions) it
is easy to see that the computation is almost entirely local and the only information one must communicate between tasks is the solution along a neighbor’s boundary.

Once a problem is broken apart to be computed on different processors the next logical question is: *How do different processors share information?* We require a way to send and/or receive data between processors. Also, we want flexibility for the implementation to work regardless of the memory space layout, i.e., shared or distributed. The message passing interface (MPI) standard provides such an approach to parallel processing [39]. MPI offers a systematic specification of how a parallel program is to package data, send/receive it among processes, as well as a host of other functions, see [9] for a complete description.

We explore two important types of communication available to MPI, blocking and non-blocking. A blocking communication refers to when one processor communicates information and *pauses execution* to wait for confirmation that the data was communicated correctly. One could think of it like trying to get into a building with a doorman, you need your friend upstairs to confirm who you are, until then you are stuck at the entrance. In contrast, a non-blocking communication is when one processor posts a request to the system to initiate communication and then continues execution until all tasks reach a wait statement. This wait command provides a user defined barrier in a program where all processors are guaranteed to synchronize. Such barriers are necessary to prevent *race conditions*, i.e., points in a program where the output of a task depends on the sequence or timing of other uncontrollable events. To continue the prior analogy, a non-blocking communication would be like entering a building with no doorman, you buzz your friend to let them know you’re here but you are free to continue upstairs to their door where you wait.

The advantage of non-blocking over blocking communication is that it allows for *latency hiding* in the program. That is, rather than idling while communication occurs, a processor is allowed to perform local computations which hide the cost of communicating data.

Next, we discuss a few aspects of an object oriented implementation on the DGSEM, for complete details see [56]. At a basic level object oriented programming represents concepts as “objects,” which contain data and procedures that operate on the data. As an example, for the DGSEM, we create an unstructured mesh object comprised of elements, edges (faces), and nodes as well as routines necessary to construct the mesh. To simplify the discussion we assume the mesh is algebraically and geometrically conforming, i.e., the polynomial order in each element is the same.
and there are no hanging nodes [56]. We also have a solution object on each element that stores the computed solution in the interior, the solution on the boundary, and the discrete time derivative, e.g., (4.1.24) in 2D. Many procedures are required to compute the semi-discrete approximation [56], but almost all the work to compute the discrete time derivative is local to an element.

The coupling in the approximation, through a Riemann flux, appears only in the final computation of the right hand side (4.1.24) in 2D or (4.3.15) in 3D. We provide the basic procedure to compute the discrete DG time derivative on an element in serial. This will make the later discussion of breaking the task apart to parallel processors more clear.

**Algorithm 2:** SerialDGSEMTimeDerivative: Compute the time derivative for the discontinuous Galerkin approximation

```markdown
Procedure SerialDGSEMTimeDerivative
    for each element do
        Interpolation of the solution onto element boundaries
        for each edge (face) do
            The numerical flux computation
        for each element do
            The time derivative computation for the DG approximation (4.1.24) (or (4.3.15))
End Procedure SerialDGSEMTimeDerivative
```

The discussion of the parallelization of the DGSEM is quite simple, but what is often overlooked is the partitioning of a global mesh into pieces on which each task will operate. This important process contributes to the question of load balancing between different tasks. Heuristically, we think of load balancing using the metric, “We want each processor to do an equal amount of work.”

The partitioning of a mesh into $L$ pieces can be done in an automated way. The most popular options for mesh partitioning are METIS [51] and SCOTCH [74]. The partitioning of the mesh simplifies because we presume the mesh for the DGSEM is unstructured.

We chose METIS to partition the meshes for the numerical results given in Chap. 8. METIS can be used to partition quadrilateral and hexahedral meshes (among others) and offers flexibility on how to balance the workload between partitions [51]. By default, load balancing is done by splitting the number of elements as evenly as possible between tasks while minimizing the number of edges along partitions [51], an example is shown in Fig 7.1. Alternatively, a user can guide how a mesh may be partitioned by assigning weights to elements, edges, etc. or by weighting the amount of communication between elements [36, 51, 82]. An example of a partition where each element is assigned a weight according to size is provided in Fig. 7.2. These weighting strategies
Figure 7.1: A mesh where each partition has the same approximate number of elements.

prove useful to balance the load when partitioning a mesh for a parallel problem that uses local time stepping integration.

Many of the differences between a serial and a parallel implementation of the DGSEM occur in the unstructured mesh construction. First, we construct a global mesh to act as a skeleton. The skeleton is partitioned and then each local mesh in constructed on a specific process. For convenience, we store global and local IDs to make it easier to reference where an element/edge is in the mesh or on which partition it lies. Most importantly, we label if an element or edge (face) lies in the interior or along a partition between processes. This allows us to easily identify if the solution on an element boundary needs to be communicated to another process or if the solution is on an interior edge where no communication is necessary.

7.2 Parallel DGSEM with a Standard Time Integrator

We first discuss the major changes one would make to transform a serial DGSEM implementation to a parallel one that uses a standard, global time stepping integrator. The end goal is to
divide Alg. 2 into tasks that require communication between processes and tasks that can be used for latency hiding.

Once the local mesh is constructed on each process we are ready for the parallel time derivative computation. The algorithm we describe may not be optimal under certain circumstances. That is, we assume an unstructured mesh and use a naïve approach to communicate the solution along element boundaries. We loop through each boundary and send all the solution information and then hide the latency by computing the Riemann flux along interior boundaries. However, as a mesh partition shrinks, i.e., we add more processors to the computation this latency hiding becomes less effective. One could optimize the sending of adjacent neighbor solution information by a ping-pong type model to further hide latency (provided the mesh is structured) as was shown in Altmann et al. [6].

In Alg. 3 we give a straightforward approach, for unstructured meshes, to parallelize the DGSEM. Because we label those boundaries that lie on a partition, we can decompose the calculation and immediately prolong to the boundary and send those boundary solutions that are required

\[ \text{Processor} \quad \text{# of Elements} \]
\begin{tabular}{ll}
1 & 13 \\
2 & 15 \\
3 & 14 \\
4 & 13 \\
5 & 44 \\
6 & 42 \\
7 & 15 \\
8 & 38 \\
9 & 16 \\
\end{tabular}

Figure 7.2: Partition of a mesh where elements are weighted according to size.
by other processes. We use non-blocking communications for latency hiding. While we communicate the boundary solution, the numerical flux on boundaries interior to the partition is calculated. We then place a wait barrier to ensure all the communications are complete. Once the solution is communicated among adjacent neighbors the algorithm computes the remaining numerical fluxes on each edge along the partition. Then we loop over each element in the local mesh to compute the time derivative. It is true that the procedure in Alg. 3 will compute the edge flux at partition edges twice, however because we solve a linear problem we follow the philosophy that “flops are free” [53]. That is, we assume the communication between tasks is far more expensive than the computation of the numerical flux. This assumption may not be true for non-linear problems where the numerical flux, such as a Roe flux, is more expensive [2].

Algorithm 3: ParDGSEMTimeDerivative: Compute the time derivative for the discontinuous Galerkin approximation on a local mesh

<table>
<thead>
<tr>
<th>Procedure ParDGSEMTimeDerivative</th>
</tr>
</thead>
<tbody>
<tr>
<td>// Send the solution along the boundary on the partition</td>
</tr>
<tr>
<td>for each partition element do</td>
</tr>
<tr>
<td>_ The interpolation of the solution onto element boundaries</td>
</tr>
<tr>
<td>for each partition edge (face) do</td>
</tr>
<tr>
<td>_ A non-blocking send of the boundary solution along the partition</td>
</tr>
<tr>
<td>// Hide latency by interpolating to boundaries and computing edge fluxes</td>
</tr>
<tr>
<td>for interior elements</td>
</tr>
<tr>
<td>for each interior element do</td>
</tr>
<tr>
<td>_ The interpolation of the solution onto element boundaries</td>
</tr>
<tr>
<td>for each interior edge (face) do</td>
</tr>
<tr>
<td>_ The numerical flux computation</td>
</tr>
<tr>
<td>// Receive the solution along the boundary on the partition and compute the numerical flux</td>
</tr>
<tr>
<td>for each partition edge (face) do</td>
</tr>
<tr>
<td>_ A non-blocking receive of boundary solution along the partition</td>
</tr>
<tr>
<td>Wait for all sends/receives to occur</td>
</tr>
<tr>
<td>for each partition edge (face) do</td>
</tr>
<tr>
<td>_ The numerical flux computation</td>
</tr>
<tr>
<td>// Complete time derivative computation</td>
</tr>
<tr>
<td>for each element do</td>
</tr>
<tr>
<td>_ The time derivative computation for the DG approximation</td>
</tr>
</tbody>
</table>

End Procedure ParDGSEMTimeDerivative
7.3 Local Time Stepping in Parallel

The local time stepping procedure from Chap. 6 remains highly parallelizable. Our goal is to divide the loop over every element step in Alg. 1, in particular the “Assemble time derivatives” step, into tasks that require communication between processes and tasks that can be used for latency hiding. The LTS integrator keeps the DGSEM approximation highly localized to an element. The coupling between adjacent neighboring elements still only occurs at element boundaries.

The process of partitioning the mesh between tasks remains largely the same, except that elements know their size and the sizes of their adjacent neighbors. The main issue when partitioning a mesh for the LTS integrator is load balancing. It is no longer a good heuristic to try and make each partition have the same number of elements. Because small elements are integrated in time more often than large elements we do not want to pack a lot of small elements into a single partition, for example. A first guess at a tactic to balance the load would be to weight elements in METIS according to their size. Some work has been done exploring the load balancing of Runge-Kutta LTS methods [82], but load balancing LTS linear multistep methods remains ad hoc. Thus, weighting elements according to size may not be an optimal process to balance the load. For instance, as we saw in the analysis of element size distribution in Sec. 6.4.3, it might be that forcing a distribution of a few small elements and many large elements in each partition could yield better load balancing.

The structure of the parallel computation of time derivatives on the local DG mesh is similar to that in Sec. 7.2. We still use non-blocking communications for all boundary information in order to hide latencies. A key difference is the added check that if the elements on either side of the partition are the same size. If elements are the same size then we send the boundary solution, otherwise we construct a time interpolant to send between the elements which integrate on different time scales. We hide latency by computing the discrete time derivative, e.g. (4.1.24), on interior elements. Next, we receive the boundary information along the partition and complete the time derivative computations on partition elements. Now we can step every element that is ready to evolve forward in time one local time step using the Adams-Bashforth time integrator. Again, note the adoption of the “flops are free” mentality as the numerical flux (and possibly the time interpolant) are performed twice along partition boundaries.
**Algorithm 4:** LTS_ParDGSEMTimeDerivative: Compute the time derivative for the LTS method with DGSEM on a local mesh

```
Procedure LTS_ParDGSEMTimeDerivative
  // Send the solution along the boundary on the partition
  for each partition edge (face) do
    if element sizes match then
      A non-blocking send of boundary solution along the partition
    else
      Time interpolant construction
      A non-blocking send of time interpolant along the partition
  // Hide latency by computing the time derivative on interior elements
  for each interior element do
    An evolve condition check
    if ReadyToEvolve then
      The time derivative computation for the DG approximation
  // Receive the solution on the boundary or time interpolant on the partition
  for each partition edge (face) do
    A non-blocking receive of boundary solution or time interpolant along the partition
  Wait for all sends/receives to occur
  // Complete time derivative computation
  for each partition element do
    An evolve condition check
    if ReadyToEvolve then
      The time derivative computation on the partition for the DG approximation
End Procedure LTS_ParDGSEMTimeDerivative
```
CHAPTER 8

NUMERICAL RESULTS

In this chapter we present a set of numerical experiments to demonstrate the results presented in Chaps 2-7. In each of the numerical experiments the speed of light is scaled so that $c = 1.0$, thus the wavespeeds $c_L$ and $c_R$ are interpreted as a fraction of the speed of light.

First in Sec. 8.1, we present fourteen examples to demonstrate the reflection and transmission of a plane wave from a moving material interface with a moving mesh. We demonstrate spectral convergence and full time accuracy for propagation of TE, TM, and acoustic waves. First, we use test solutions [21, 46, 48, 72, 84, 97, 98], outlined in the Chap. 3, to show that the approximation is spectrally accurate. To exercise all the terms in the numerical fluxes (5.1.30), (5.1.46), and (5.1.58), we study the convergence in the numerical solution from a uniformly moving interface at different angles of incidence and different angles of translation. Next, we verify numerically the results from Sec. 5.2, which say that as the moving interface becomes a moving mirror, the numerical flux enforces the correct set of relativistic boundary conditions. We examine the error in the transmitted wave as a function of the angle of incidence. We also examine the static electromagnetic phenomena of Brewster’s angle and TIR from Sec. 3.5. Finally, we perform a grid convergence study where we approximate wave reflection and transmission of a TE plane wave from a moving, embedded dielectric object.

In the second, Sec. 8.2, we combine the ALE-DGSEM spatial discretization with the AB3LTS time integrator in two and three spatial dimensions. The first demonstrates spectral convergence and full time accuracy of the approximation on static and moving meshes with local refinement. The second compares the observed reduction in computational cost for the LTS procedure applied to both static and moving test meshes to the predictions given in Table 6.2 in Sec. 6.4.2. Last we apply the LTS procedure to a steady state calculation for comparison with a global time integrator.

Finally, in Sec. 8.3, we present speedup results for parallel implementations of the DGSEM using a standard time integrator as well as the LTS integrator from Chap. 6. Speedup of the parallel DGSEM is examined for two dimensional and three dimensional static meshes.
8.1 Reflection and Transmission of a Plane Wave from a Moving Material Interface

We provide six numerical examples for the TE Maxwell’s equations, four examples for the TM Maxwell’s equations, and four examples for the classical wave equation to demonstrate the computation of reflection and transmission of a plane wave from a moving material interface.

8.1.1 Transverse Electric Waves

To study reflection and transmission of TE electromagnetic waves, we choose the incident wave to be a Gaussian plane wave pulse

\[
\begin{bmatrix}
E_z \\
B_x \\
B_y
\end{bmatrix} = \begin{bmatrix}
k_y \\
\frac{1}{\omega} \\
-k_x
\end{bmatrix} e^{-\frac{(k_x x^2 + k_y y^2 - \omega t)^2}{w^2}},
\]

(8.1.1)

with the wavevector \( \mathbf{k} \). The parameter \( w \) is computed from the full width at half maximum by \( w = 0.2/\sqrt{\ln(2)} \). We allow the relative permittivity \( \varepsilon \) to take two values: one for free space and one for the dielectric slab. The permeability \( \mu \) for the magnetic field is kept as the same constant value for both sides.

We consider three examples to demonstrate the spectral accuracy of the approximation. In the first we keep the interface normal to the direction of motion. The second, to assess all the terms of the numerical flux (5.1.30), rotates the interface obliquely relative to the direction of motion. For these tests, we set the permeability \( \mu = 1 \) and the permittivity as the piecewise constant

\[
\varepsilon = \begin{cases}
\varepsilon_L = 1.0, & \text{if } x \leq x_0 + v_0 t \\
\varepsilon_R = 2.25, & \text{if } x > x_0 + v_0 t
\end{cases},
\]

(8.1.2)

so that the index of refraction is \( n = 1.5 \). The third convergence test sets the moving interface to act as a moving mirror and verifies that the numerical flux imposes the correct boundary conditions. Finally, we provide a study of the incident angles and its relation to the error in the transmitted wave.

With a thorough convergence study complete, we demonstrate the use on the reflection and transmission of a TE plane wave from an oscillating dielectric elliptical cylinder. This more complex test problem does not have an analytical solution, so we provide a grid convergence study.
**Vertical Interface.** For the first example we take the physical domain to be \([-10, 10] \times [0, 10]\), with the material interface initially placed at \(x = 0\) that moves with velocity \(v_o = (0, 3, 0)\). The incident wave is (8.1.1) with angle of incidence \(\theta_i = \pi/4\).

We construct the analytical solution from the normal phase angles from Sec. 3.2, the reflection and transmission coefficients from (3.3.6), and recover the other electric and magnetic fields using (3.4.1).

Contour plots in Figs. 8.1(a) and (b) for times \(t = 0.0\) and \(t = 3.0\) show the propagation of the \(D_z\) wave using polynomial order \(N = 6\) in each direction on each element and \(\Delta t = 1/2000\). We used the analytical solution as the external state on each boundary. Notice that the reflection angle is not equal to the incident angle when the boundary moves, as expected from theory [72].

![Figure 8.1](image)

(a) \(t = 0.0\)

(b) \(t = 3.0\)

Figure 8.1: Reflection and transmission of a TE Gaussian plane wave from a moving dielectric interface. Dashed lines are negative contours, the moving dielectric is shaded gray, and the overlay of squares shows the locations of the element boundaries.

Fig. 8.2 shows exponential convergence in space until \(N = 13\) where the error is dominated by time integrator errors. When the value of \(\Delta t\) is halved, the error in the approximation is reduced by a factor of 8, as expected for the third order Adams-Bashforth time integration scheme used.
Oblique Interface. To test all the terms in the numerical flux (5.1.30), we make the dielectric interface oblique to the velocity $\mathbf{v}_o$, with normal $\hat{n}$. The analytical solution takes the form (8.1.1) and is constructed from the rotated phase angles from Sec. 3.2.1, the rotated reflection and transmission coefficients from Sec. 3.3.1, and the other electric and magnetic fields are recovered using (3.4.2), (3.3.7), and (3.4.3).

The physical domain is $[-5, 5] \times [1, 6]$, with a dielectric interface initially placed oblique to the $x$–axis with angle $\theta = \pi/16$ and moved with constant velocity $\mathbf{v}_o = (-0.1, 0)$. The incident wave is (8.1.1) with angle of incidence $\theta_i = \pi/4$ with respect to the $x$–axis.

Contour plots in Figs. 8.3(a) and (b) for $t = 0.0$ and $t = 2.0$ show the propagation of the $D_z$ wave for polynomial order $N = 6$ in each direction on each element and $\Delta t = 1/2000$. As in the previous example, we used the analytical solution as the state on each external boundary.

Fig. 8.4 shows exponential convergence in space until $N = 12$ where the error is dominated by time integrator errors. When the value of $\Delta t$ is halved the error in the approximation is again reduced by a factor of 8.

Oblique Mirror. Next, we verify that the numerical flux imposes the correct set of relativistic boundary conditions if the moving interface acts as a mirror, i.e., when $\varepsilon_R \to \infty$. In the test problem the mirror is oblique to the velocity $\mathbf{v}_o$, with normal $\hat{n}$. The computational domain is identical
Figure 8.3: Reflection and transmission of a TE Gaussian plane wave from an oblique moving dielectric interface at two times. Dashed lines are negative contours. The moving dielectric is shaded gray.

Figure 8.4: Semi-log plot shows the spectral convergence of the approximation for the oblique moving interface problem.

to the previous example and moves with constant velocity \( \mathbf{v}_o = (-0.1, 0) \) and angle of incidence \( \theta_i = \pi/4 \).

Figs. 8.5(a) and (b) for \( t = 0.0 \) and \( t = 2.0 \) show contour plots of the \( D_z \) wave for polynomial order \( N = 6 \) in each direction on each element and \( \Delta t = 1/2000 \). As usual, we use the analytical solution as the external state on each boundary. Fig. 8.2 shows exponential convergence in space
until \( N = 13 \) where the error is dominated by time integrator errors. We also see third order accuracy in time when the time step is halved.

Figure 8.5: Reflection of a TE Gaussian plane wave from an oblique moving mirror. Dashed lines are negative contours. The moving mirror is shaded gray.

Figure 8.6: Semi-log plot shows the spectral convergence of the approximation for the oblique moving mirror problem.

**Angle Study.** We quantify the error of the approximation to the TE Maxwell’s equations associated with the angle of incidence. The numerical fluxes (5.1.30) transmit information in the normal direction along characteristics, but move tangential terms non-characteristically [90]. We take the domain for the computation to be \([-10, 10] \times [0, 10]\) with the material interface initially
placed along $x = 0$ moving with $v_0 = (-0.2, 0)$. The incident wave is $(3.1.1)$, but this time we calculate the reflection and transmission for different angles of incidence $k\theta_i = k\pi/20$, $k = 0, \ldots, 10$. The final time is $T = 1.0$ and we fix the time step at $\Delta t = 1/2000$.

Fig. 8.7(a) shows the error in the TE transmitted wave as a function of the incident angle. The error decreases as the angle of incidence becomes close to normal with the material interface, as we expect for a normal numerical flux. We see that smaller angles require higher polynomial order to reach the same error as larger angles. However, in all cases after refinement we retain spectral convergence as shown for $\theta_i = \pi/10$ and $\theta_i = 2\pi/5$ in Fig. 8.7(b).

Figure 8.7: (a) Error in the transmitted $D_z$ electric field for three values of $N$. (b) Spectral convergence for incident angles $\theta_i = \pi/10$ and $\theta_i = 2\pi/5$.

**Rocking Ellipse.** Next, we consider the reflection and transmission of a TE plane wave from an oscillating dielectric elliptical cylinder. After the previous thorough convergence study, we expect the numerical approximation to be accurate for an arbitrary angle of incidence at an oblique material interface. We provide a grid convergence study for the reflection and transmission from an oscillating cylinder as this more complex test problem does not have an analytical solution.
The physical domain is \([-10, 10] \times [-10, 10]\) with an elliptical dielectric cylinder initially centered at \((x_0, y_0) = (0, 0.5)\). The inner cylindrical boundary moves according to

\[
\begin{align*}
x(t) &= x_0 \cos(a \sin(ft)) - y_0 \sin(a \sin(ft)), \\
y(t) &= x_0 \sin(a \sin(ft)) + y_0 \cos(a \sin(ft)) ,
\end{align*}
\]  

(8.1.3)

where \(a = \pi/10\) and \(f = \pi/4\). The state on each external boundary is set to be the analytical solution as if there was no wave scatterer, i.e., the incident wave propagating through the domain. We choose the outer boundary far enough away so that the reflected electromagnetic waves have not yet reached it at the final time \(T = 6.0\).

Contour plots in Figs. 8.8(a)-(d) for times \(t = 0.0\), \(t = 2.0\), \(t = 4.0\), and \(t = 6.0\) show the propagation of the \(D_z\) using polynomial order \(N = 6\) in each direction on each element and \(\Delta t = 1/2000\). The initial transmission and reflection of the incident wave is shown in Fig. 8.8(b). In Fig. 8.8(c) and (d) we see the scattering of this initial reflection as well as the internal transmissions and reflections that occur within the cylindrical wave scatterer.

We see, in Fig. 8.9, exponential convergence for a spatial grid convergence study. There is no analytical solution for this reflection and transmission problem. As such, we ran the approximation at polynomial order \(N = 24\) and re-interpolated to a uniform grid of 24 points in each direction to act as the analytical solution. The final time is \(T = 3.0\) and we fix the time step at \(\Delta t = 1/4000\).

**Total Internal Reflection & Frustrated Total Internal Reflection.** Finally, we examine of the accuracy of the TE numerical flux (5.1.30) on a static problem where the wave propagates from a denser medium to a lighter medium and exhibits TIR. This test problem examines an unusual physical phenomenon and how well the numerical approximation captures the true physics. We saw in Sec. 3.5 that all the energy is reflected back into the dense medium along with finite penetration into the lighter medium by an evanescent wave. We set the material to be

\[
\varepsilon = \begin{cases}
\varepsilon_T & = 2.25, \quad \text{if } y \geq 0 \\
\varepsilon_B & = 1.0 , \quad \text{if } y < 0
\end{cases}
\]  

(8.1.4)

We consider a plane wave with incidence angle \(\theta_i = \pi/4\), angular frequency \(\omega = 2\pi f\), frequency \(f = 3.5\), and \(w = 1\).

Contour plots in Figs. 8.10(a) and (b) for \(t = 0.0\) and \(t = 1.0\) show the propagation of the \(D_z\) wave using polynomial order \(N = 6\) in each direction on each element and \(\Delta t = 1/2000\). Fig. 8.16 shows exponential convergence in space until \(N = 11\), where the error is dominated by time
integrator errors which are, again, third order. So, for a static problem, the Riemann flux (5.1.30) imposes the true physical property of TIR at the material interface.
Figure 8.9: Semi-log plot shows the spectral convergence of the approximation for the reflection and transmission from an oscillating cylinder.

Figure 8.10: Total internal reflection of a TE Gaussian plane wave. We see the finite penetration depth of the evanescent wave into the lighter material. The dielectric material is shaded gray.

With the ability to model TIR accurately we next run a qualitative experiment of modeling frustrated total internal reflection. To the best of our knowledge the problem of fTIR does not have an analytical solution. So, the main focus is to test whether the numerical approximation will
properly exhibit the ability of a plane wave to jump a small gap of lighter material between two dense materials provided the gap is smaller than the penetration depth of the evanescent wave. As such, we consider the three material problem with

$$\varepsilon = \begin{cases} 
\varepsilon_T = 2.25, & \text{if } y \geq 0 \\
\varepsilon_M = 1.0, & \text{if } -0.1 < y < 0 \\
\varepsilon_B = 2.35, & \text{if } y < -0.1 
\end{cases} \quad (8.1.5)$$

We take the plane wave to have the same incidence angle and angular frequency as the TIR test problem.

The computational results in Fig. 8.12 capture the main physical features of fTIR, that a plane wave can penetrate across a gap between dense materials. We account for the strange look in the reflected and “jumped” waves, particularly near the gap, for two reasons. The boundary conditions for the simulation are set by the plane wave problem as if the gap was not present. The complex valued amplitudes, and thus the Goos-Hächen shift, change in the case of fTIR [47]; however, we used the traditional complex Fresnel coefficients discussed in Sec. 3.5.

### 8.1.2 Transverse Magnetic Waves

Next, we verify the ALE formulation and numerical flux for the TM polarization. Recall that the orthogonally polarized TE and TM fields may be superimposed to construct any arbitrary electromagnetic field [72].
Figure 8.12: Frustrated total internal reflection of a TE Gaussian plane wave. Note the evanescent wave allows the plane wave to jump a small gap (in this case of air). The dielectric materials are shaded gray.

For the examples of reflection and transmission of TM electromagnetic waves, we apply the duality transformation (2.1.2) to the Gaussian plane wave pulse (8.1.1) and find that

$$
\begin{bmatrix}
H_z \\
D_x \\
D_y
\end{bmatrix} = \begin{bmatrix}
1/k_y \\
-k_x/\omega \\
k_x/\omega
\end{bmatrix} e^{-\frac{(k_x x^2 + k_y y^2 - \omega t)^2}{w^2}},
$$

with wavevector \( \mathbf{k} \).

**Vertical Interface.** For the first example we take the domain for the computation to be \([-10, 10] \times [0, 10]\) with the material interface initially placed along \( x = 0 \). The interface then moves at speed \( \mathbf{v}_o = (-0.1, 0) \). The incident wave is (8.1.6) with angle of incidence \( \theta_i = \pi/4 \).

Contour plots in Figs. 8.13(a) and (b) for \( t = 0.0 \) and \( t = 2.0 \) show the propagation of the \( B_z \) wave for polynomial order \( N = 6 \) in each direction on each element and \( \Delta t = 1/2000 \). Fig. 8.14 shows exponential convergence in space until \( N = 14 \) where the error is dominated by time integrator errors. It also shows that third order accuracy in time is retained.
Figure 8.13: Reflection and transmission of a TM Gaussian plane wave from a moving dielectric interface initially located at \( x = 0 \). The moving dielectric is shaded gray.

Figure 8.14: Semi-log plot shows the spectral convergence and design time accuracy of the approximation.

**Oblique Interface.** To test all the terms in the TM numerical flux (5.1.46), we set the dielectric interface to be oblique to the velocity \( \mathbf{v}_o = (-0.1, 0) \) at an angle \( \theta = \pi/16 \) with the \( x \)-axis. The incident wave is (8.1.6) with angle of incidence \( \theta_i = \pi/4 \) with respect to the \( x \)-axis.
Contour plots in Figs. 8.15(a) and (b) for \( t = 0.0 \) and \( t = 2.0 \) show the propagation of the \( B_z \) wave using polynomial order \( N = 6 \) in each direction on each element and \( \Delta t = 1/2000 \). Fig.

![Contour plots for different times](image)

(a) \( t = 0.0 \)  
(b) \( t = 2.0 \)

Figure 8.15: Reflection and transmission of a TM Gaussian plane wave from an oblique moving dielectric interface. The moving dielectric is shaded gray.

8.16 shows exponential convergence in space until \( N = 12 \), where the error is dominated by time integrator errors which are, again, third order.

![Semi-log plot showing spectral convergence](image)

Figure 8.16: Semi-log plot shows the spectral convergence of the approximation for the oblique moving interface problem.

**Oblique Mirror.** Just as with the TE case, we verify that the TM numerical flux (5.1.46) imposes the correct set of relativistic boundary conditions if the moving interface acts as a mirror.
We solve the problem where the mirror is oblique to the velocity \( \mathbf{v}_o \), with normal \( \hat{n} \). The analytical solution is constructed from the rotated phase angles and reflection coefficient. The computational domain is identical to the previous example and moves with constant velocity \( \mathbf{v}_o = (-0.1, 0) \) and angle of incidence \( \theta_i = \pi/4 \).

Figs. 8.17(a) and (b) for \( t = 0.0 \) and \( t = 2.0 \) show contour plots of the \( B_z \) wave using polynomial order \( N = 6 \) in each direction on each element and \( \Delta t = 1/2000 \). Fig. 8.18 shows exponential convergence in space until \( N = 14 \) where the error is dominated by time integrator errors. We also show third order accuracy in time when we halve the time step.

![Figure 8.17: Reflection of a TM Gaussian plane wave from an oblique moving mirror. The moving mirror is shaded gray.](image)

**Brewster’s Angle.** Finally, we examine of the accuracy of the TM numerical flux (5.1.46) on a static problem where the wave propagates at Brewster’s angle. It is important to test the accuracy and limitations of the numerical approximation’s ability to capture the physics of a problem, particularly in extreme cases. We saw for Brewster’s angle in Sec. 3.5 that all the energy is transmitted across the material interface and there is no reflected wave. We set the material to be the same as (8.1.2) with \( \mathbf{v}_o = \mathbf{0} \). Because Brewster’s angle is material dependent we set the incidence angle according to the formula (3.5.7).

Contour plots in Figs. 8.19(a) and (b) for \( t = 0.0 \) and \( t = 5.0 \) show the propagation of the \( B_z \) wave using polynomial order \( N = 6 \) in each direction on each element and \( \Delta t = 1/2000 \). Fig. 8.20 shows exponential convergence in space until \( N = 11 \), where the error is dominated by time...
Figure 8.18: Semi-log plot shows the spectral convergence of the approximation for the oblique moving mirror problem.

Figure 8.19: Pure transmission of a TM Gaussian plane wave traveling at Brewster's angle. The dielectric material is shaded gray.

integrator errors which are, again, third order. So, for a static problem, the Riemann flux (5.1.46) imposes the true physical property of full transmission at the material interface.
2.10 − 8
2.10 − 7
2.10 − 6
2.10 − 5
2.10 − 4
2.10 − 3
2.10 − 2
2.10 − 1
L∞ Error
\Delta t = 1/2000
\Delta t = 1/4000

Figure 8.20: Semi-log plot shows the spectral convergence of the approximation for Brewster’s angle problem.

8.1.3 Acoustic Waves

We also show that the ALE formulation and numerical fluxes for the classical wave equation wave scattering model lead to a spectrally accurate approximation in space and is third order accurate in time. The incident Gaussian plane wave pulse for pressure and velocity will be

$$\frac{\mathbf{p}}{u} = \begin{bmatrix} \frac{1}{k_x} \\ \frac{k_x}{k_y} \end{bmatrix} e^{-(k_x x^2 + k_y y^2 - \omega t)^2}$$

with $|k| = k_x^2 + k_y^2$.

We consider three examples: a moving normal interface, a moving oblique interface and a moving oblique mirror, to exercise all the terms of the wave equation numerical flux (5.1.58). We take the wave speed $c$ to be the piecewise constant

$$c = \begin{cases} 
  c_L = 1.0, & \text{if } \mathbf{x} \leq \mathbf{x}_0 + \mathbf{v}_o t \\
  c_R = 0.8, & \text{if } \mathbf{x} > \mathbf{x}_0 + \mathbf{v}_o t
\end{cases}$$

Vertical Interface. For the first example we take the domain for the computation to be $[-10,10] \times [0,10]$ with the material interface initially placed along $x = 0$ and moving with velocity $\mathbf{v}_o = (-0.1,0)$. The incident wave is (8.1.7) with angle of incidence $\theta_i = \pi/4$.

Pressure contour plots in Fig. 8.21(a) and (b) for $t = 0.0$ and $t = 2.0$ show the computed reflection and transmission of the wave from the moving material interface. We show that the
The numerical solution converges to the analytical solution exponentially fast, as expected, in Fig. 8.22.

Figure 8.21: Reflection and transmission of the pressure plane wave from a moving material interface initially located at \( x = 0 \). The moving medium is shaded gray.

Figure 8.22: Semi-log plot showing exponential convergence of the error for the numerical solution of the constantly moving vertical material interface test problem.
**Oblique Interface.** Next we let the material interface be oblique to the velocity \( \mathbf{v}_o \), with normal \( \mathbf{n} \) to assess all terms in the numerical flux (5.1.58). In Fig. 8.23 we show contour plots of the pressure for \( t = 0.0 \) and \( t = 2.0 \). Fig. 8.24 shows exponential convergence of the error in space.

Figure 8.23: Reflection and transmission of the pressure from a moving oblique material interface with a normal angle \( \theta = \pi/16 \). The moving medium is shaded gray. (a) The pressure, \( p \), of the initial wave pulse. (b) The reflection and transmission of the incident wave at a later time.

Figure 8.24: Semi-log plot showing exponential convergence of the error for the numerical solution of the constantly moving oblique material interface.
Oblique Wall. We verify that the numerical flux imposes the correct boundary conditions at a moving wall as $c_R \to 0$ in (5.1.58) for the wave equation. We solve the problem where the reflecting wall is oblique to the velocity $v_o$, with normal $\hat{n}$. The computational domain is identical to the previous example and the interface moves with constant velocity $v_o = (-0.1, 0)$ and angle of incidence $\theta_i = \pi/4$.

Fig. 8.25(a) and (b) for $t = 0.0$ and $t = 2.0$ show pressure contours using polynomial order $N = 6$ in each direction on each element and $\Delta t = 1/2000$. Fig. 8.26 shows exponential convergence in space until $N = 10$ where the error is dominated by time integrator errors. We also see third order accuracy in time when we halve the time step.

Figure 8.25: Pure reflection of the pressure from an oblique moving wall, which is shaded gray.

Angle Study. We quantify the error of the approximation to the classical wave equation associated with the angle of incidence. We take the domain for the computation to be $[-10, 10] \times [0, 10]$ with the material interface initially placed along $x = 0$ moving with $v_o = (-0.2, 0)$. The incident wave is (8.1.7), where we calculate the reflection and transmission for different angles of incidence $k\theta_i = k\pi/20$, $k = 0, \ldots, 10$. The final time is $T = 1.0$ and we fix the time step at $\Delta t = 1/2000$.

Fig. 8.27(a) shows the error in the transmitted pressure wave as a function of the incident angle. The error decreases slightly as the angle of incidence becomes close to normal with the material interface. For each incidence angle tested we see that, after refinement, we retain spectral convergence. This is shown explicitly for $\theta_i = \pi/10$ and $\theta_i = 2\pi/5$ in Fig. 8.27(b).
Figure 8.26: Semi-log plot shows the spectral convergence of the approximation for the oblique moving wall problem.

Figure 8.27: (a) Error in the transmitted pressure $p$ for three values of $N$. (b) Spectral convergence for incident angles $\theta_i = \pi/10$ and $\theta_i = 2\pi/5$. 
8.2 Local Time Stepping on Static and Moving Quadrilateral and Hexahedral Meshes

In this section we provide examples that combine the ALE-DGSEM spatial discretization with the AB3LTS time integrator in two and three spatial dimensions. We demonstrate spectral convergence and design time accuracy of the approximation on static and moving meshes with local refinement. Also, we compare the observed reduction in computational cost for the LTS procedure with the predictions from Sec. 6.4.2, and, lastly, apply the LTS procedure to a steady state calculation for comparison with a global time integrator.

8.2.1 Convergence Study of the ALE-DGSEM with AB3LTS

We first show spectral convergence and full time accuracy in each time scale for the ALE-DGSEM with LTS on static and moving meshes. For this study, we solve the wave equation written as a conservation law (5.1.59) with initial and boundary conditions set by a Gaussian plane wave solution (8.1.7) or in three dimensions

$$\begin{pmatrix} p \\ u \\ v \\ w \end{pmatrix} = \begin{pmatrix} \frac{1}{k_x} \\ \frac{k_x}{c} \\ \frac{k_y}{c} \\ \frac{k_z}{c} \end{pmatrix} e^{-\frac{(k_x(x-x_0)^2+k_y(y-y_0)^2+k_z(z-z_0)-ct)^2}{\omega^2}},$$  \tag{8.2.1}

with the wavevector \( \mathbf{k} \) normalized to satisfy \( k_x^2 + k_y^2 + k_z^2 = 1 \). We take \( c = 1 \) and vary \( x_0, y_0, \) and \( z_0 \) to adjust the initial position.

For the simple linear wave equation (5.1.59), the linear fluxes and exact Riemann solver are inexpensive to evaluate compared to the matrix-vector products required to compute the time derivative. Thus, numerical experiments using the classical wave equation will give worst case results for the speedup offered by the LTS method. For equations with non-linear fluxes, it will be far more expensive to compute fluxes and solve the Riemann problem relative to other operations, e.g., interpolation.

**Static Quadrilateral.** We obtain spectral accuracy in space and design accuracy in time for the DGSEM with LTS on a static mesh with local refinement. For the example, we take \( w = \omega/2\sqrt{\ln(2)}, \omega = 0.2 \), and choose \( x_0 = y_0 = -0.8 \). The contour plots in Fig. 8.28 show the propagation of the Gaussian plane wave up to the final time \( T = 0.5 \). We re-interpolate
the computed solution to a uniform mesh of 10 points in each direction per element for plotting purposes.

The left of Fig. 8.29 shows exponential convergence in space until \( N = 16 \), where the error is dominated by time integrator errors. Recall that we chose \( \Delta t_0 \) to be the time step on the largest elements in Group 0. Note that when the value of \( \Delta t_0 \) is halved the error in the approximation is reduced by a factor of eight.

The right plot of Fig. 8.29 demonstrates design third order temporal accuracy in each of the groups of elements. Fig. 8.29 indicates that the error is largest on the Group 0 elements, as expected. We add a time convergence plot for each local time step level to ensure that third order convergence is achieved on every level of refinement and not just the error-dominating elements in Group 0. To produce the plot, we fixed \( N = 20, T = 0.5 \) and let \( \Delta t \) range from \( 1/5000 \) to \( 1/10000 \).

![Contour plots of the plane wave propagation across a refined mesh. The overlay of squares shows the element boundaries.](image)

Figure 8.28: Contour plots of the plane wave propagation across a refined mesh. The overlay of squares shows the element boundaries.

**Moving Quadrilateral.** Next we demonstrate spectral convergence and design time accuracy of the AB3LTS time integrator with the ALE-DGSEM on a moving quadrilateral mesh with local refinement. We also verify free stream preservation. The convergence study for this numerical test exercises the variable coefficient formulation of the AB3LTS time integrator derived in Sec. 6.2.
The LTS algorithm is spectrally convergent in space and third order convergent in time for each group of elements of the moving mesh. For the test problem, we choose the same parameters for the plane wave as in Sec. 8.2.1, except $x_0 = 0.0$ and $y_0 = -0.5$. The contour plots in Fig. 8.30 show the propagation of the plane wave up to the time $T = 1.0$.

The left of Fig. 8.31 shows spectral convergence in the spatial error at $T = 1.0$ until $N = 13$, after which the error is dominated by errors introduced by the time integrator. When the value of $\Delta t_0$ is halved the error in the approximation is reduced by the expected factor of eight. The convergence rate in time is also third order for each group of elements, which we show at the right in Fig. 8.31. Again, the error is largest on the elements in Group 0. To produce the plot we fix $N = 16$, $T = 1.0$ and let $\Delta t$ range from $1/3500$ to $1/7000$.

**Free-Stream Preservation.** To show that the LTS method does not introduce spurious waves and preserves a free-stream when a mesh moves, we consider a uniform solution in time

$$
q = \begin{bmatrix} p \\ u \\ v \end{bmatrix} = \begin{bmatrix} \pi \\ \pi \\ \pi \end{bmatrix},
$$

(8.2.2)
Figure 8.30: Contour plots of the plane wave propagation across a refined mesh with a moving circular boundary. The overlay of quadrilaterals shows the element boundaries.

Figure 8.31: (left) Semi-log plot shows spectral convergence for the ALE-DGSEM with AB3LTS. (right) A log-log plot showing third order temporal accuracy of AB3LTS in each group of elements.

on a mesh with an inner circular boundary and local refinement center shown previously in Fig. 6.4(a). The mesh has fixed outer boundaries, whereas the inner circular boundary moves according
to

\[ X(t) = X(0) + a \sin(ft), \]
\[ Y(t) = Y(0) + a \cos(ft), \]  

(8.2.3)

where \( a = 0.15 \) and \( f = \pi/2 \).

In the test of free stream preservation, we used the variable coefficient AB3LTS time integrator and a spring-dashpot model to calculate the mesh velocity. The error was calculated using the maximum norm over the component \( p \) and the exact constant solution at \( T = 2.0 \), which corresponds to a complete period in the oscillation of the inner circular boundary. Table 8.1 shows the computed error for double precision computations.

Table 8.1: Computed error of a constant solution to the wave equation on a moving mesh with local time stepping.

<table>
<thead>
<tr>
<th>N</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>(2.0 \times 10^{-14})</td>
<td>(2.1 \times 10^{-14})</td>
<td>(2.1 \times 10^{-14})</td>
<td>(3.6 \times 10^{-14})</td>
<td>(6.6 \times 10^{-14})</td>
<td>(6.9 \times 10^{-14})</td>
</tr>
</tbody>
</table>

**Static Hexahedral.** We obtain spectral accuracy in space and design accuracy in time for the DGSEM with LTS on a three dimensional static mesh with local refinement. For the example, we take \( w = \omega/2\sqrt{\ln(2)}, \ \omega = 0.2 \), and choose \( x_0 = y_0 = z_0 = 0 \) in the Gaussian plane wave (8.2.1). The plots in Fig. 8.32 show the propagation of the Gaussian plane wave up to the final time \( T = 1.0 \). For plotting purposes we re-interpolate the computed solution to a uniform mesh of 10 points in each direction per element.

The left of Fig. 8.33 shows exponential convergence in space until \( N = 10 \), where the error is dominated by time integrator errors. Also, when the value of \( \Delta t_0 \) is halved the error is reduced by a factor of eight. The right plot of Fig. 8.33 demonstrates design third order temporal accuracy in each of the groups of elements. Again we see the error is dominated by elements in Group 0. To produce the plot, we fixed \( N = 12, \ T = 1.0 \) and let \( \Delta t \) range from \( 1/2000 \) to \( 1/4000 \).

**Free-Stream Preservation.** We use the conservative curl form (4.1.7) of the contravariant basis vectors to prevent spurious waves on hexahedral meshes with curved boundaries [55]. We use the constant solution (8.2.2) to demonstrate the free stream preservation of the approximation.

Table 8.2 shows the computed error in the pressure \( p \) for double precision computations. As we anticipate, the error is on the order of rounding error. Note that the rounding errors are larger and
Figure 8.32: Pseudocolor plots of plane wave propagation across a refined mesh. The overlay of hexagons show element boundaries.

Figure 8.33: (left) Semi-log plot of the spectral convergence for the DGSEM with AB3LTS on a 3D refined mesh. (right) A log-log plot of the third order temporal accuracy of AB3LTS in each group of elements.
increase faster than when the cross product form was used in two dimensions. This is because the numerical derivatives are sensitive to round-off errors \cite{19, 56}. The curl form (4.1.7) requires two applications of the derivative operator, whereas the cross product form (4.1.6) requires only one.

Table 8.2: Maximum error for a constant solution to the wave equation on a three dimensional mesh with local time stepping.

<table>
<thead>
<tr>
<th>N</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>$6.3 \times 10^{-14}$</td>
<td>$2.2 \times 10^{-13}$</td>
<td>$2.8 \times 10^{-13}$</td>
<td>$9.4 \times 10^{-13}$</td>
<td>$2.4 \times 10^{-12}$</td>
<td>$3.4 \times 10^{-12}$</td>
</tr>
</tbody>
</table>

**Moving Hexahedral.** We obtain spectral accuracy in space and design accuracy in time for the DGSEM with LTS on a moving three dimensional mesh with local refinement. For the example, we take \( w = \omega / 2 \sqrt{\ln(2)} \), \( \omega = 0.2 \), and choose \( x_0 = 0.1 \) and \( y_0 = z_0 = 0 \) in the Gaussian plane wave (8.2.1). The plots in Fig. 8.34 show the propagation of the Gaussian plane wave up to the final time \( T = 0.5 \). For plotting purposes we re-interpolate the computed solution to a uniform mesh of 10 points in each direction per element.

The left of Fig. 8.35 shows exponential convergence in space until \( N = 10 \), where the error is dominated by time integrator errors. The right plot of Fig. 8.35 demonstrates design third order temporal accuracy in each of the groups of elements. To produce the plot, we fixed \( N = 11 \), \( T = 0.5 \) and let \( \Delta t \) range from \( 1/3500 \) to \( 1/7000 \).

**Free-Stream Preservation.** Table 8.3 shows the expected double precision error in the pressure \( p \) to demonstrate the free stream preservation. The circular boundary in Fig. 8.34 moves according to the formulae (8.2.3). Again, because the numerical derivatives are sensitive to round-off errors, the rounding errors are larger and increase faster than when the cross product form was used in two dimensions.

Table 8.3: Error for a constant solution to the wave equation on a three dimensional moving mesh with local time stepping.

<table>
<thead>
<tr>
<th>N</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>$2.7 \times 10^{-14}$</td>
<td>$2.2 \times 10^{-13}$</td>
<td>$3.5 \times 10^{-13}$</td>
<td>$3.9 \times 10^{-13}$</td>
<td>$9.9 \times 10^{-13}$</td>
<td>$3.7 \times 10^{-12}$</td>
</tr>
</tbody>
</table>
Figure 8.34: Pseudocolor plots of plane wave propagation across a refined mesh. The overlay of hexagons show element boundaries.

Figure 8.35: (left) Semi-log plot of the spectral convergence for the DGSEM with AB3LTS on a moving 3D refined mesh. (right) A log-log plot of the third order temporal accuracy of AB3LTS in each group of elements.
8.2.2 Reducing Non-Physical Stiffness on Quadrilateral Meshes: Numerical Verification

We now use the LSRK3 and AB3LTS to compare the actual speedup to the predicted ones found in Sec. 6.4.2 for the same linear plane wave propagation as in Sec. 8.2.1. We investigate the stiffness introduced by a mesh with local refinement and its affect on the overall computation time. Because the hexahedral test meshes in Secs. 8.2.1 are generated with simple extrusion, we expect the size and speedup estimates to be similar to the two dimensional case. However, on moving meshes one might expect that the LTS in 3D will offer significant speedup because, heuristically, recomputing a hexahedron’s geometry is very expensive, particularly the conservative curl form of the contravariant basis vectors (4.1.7) [55]. Thus, limiting the number of times one updates a three dimensional element’s geometry will be computationally efficient.

We set exact boundary conditions at the edges of the computational domain and any interior boundaries, such as at the surface of the NACA0012 airfoil. With the exact boundary conditions, any internal objects act as transparent boundaries. As such, the test problems are unphysical. However, the point is that the LTS procedure is designed to reduce the stiffness introduced by unphysical local refinement. The test problems are designed to show that it does.

We present in Table 8.4 the measured speedup for the static meshes listed in Table 6.2. We choose the time step for AB3LTS with the procedure described in (6.1.5). The time step for LSRK3 is given by (6.4.5). For the tests on static and moving meshes we take \( N = M = 6 \) and \( \Delta t_0 = 1/1000 \). We see that the speedup formula (6.4.7) tends to overestimate the predicted reduction of computational cost on static meshes. This is because we neglected the additional computation time required to create the polynomial interpolant.

To examine the computational work required to construct time interpolants, we profiled a simulation with LTS on the Circle Array mesh. We find the assembly of the time derivative is 55% of the computation time. The construction of the time interpolants accounts for 4%. Table 8.4 shows that as the number of time scales increases, the time interpolants become less important to the total computation time, and the work estimate becomes more accurate.

The work estimate is better for moving meshes with local refinement. In Table 8.5 we show the speedup for the same test meshes used in Table 8.4, but now the interior boundaries move sinusoidally by (8.2.3). Recall that the work estimate included the recalculation of the element
Table 8.4: Comparison of the predicted and measured speedup of AB3LTS versus the global time stepping LSRK3 on several static meshes.

<table>
<thead>
<tr>
<th>Mesh</th>
<th>LSRK3 (m)</th>
<th>AB3LTS (m)</th>
<th>Predicted</th>
<th>Measured</th>
<th>% Diff</th>
</tr>
</thead>
<tbody>
<tr>
<td>Box Around Circle</td>
<td>1.03</td>
<td>0.77</td>
<td>1.6</td>
<td>1.3</td>
<td>-19</td>
</tr>
<tr>
<td>Circle in Circle</td>
<td>3.01</td>
<td>1.19</td>
<td>2.8</td>
<td>2.5</td>
<td>-12</td>
</tr>
<tr>
<td>Circle Array</td>
<td>11.50</td>
<td>1.18</td>
<td>10.6</td>
<td>9.7</td>
<td>-9</td>
</tr>
<tr>
<td>NACA0012 Airfoil</td>
<td>104.74</td>
<td>11.22</td>
<td>9.4</td>
<td>9.3</td>
<td>-1</td>
</tr>
</tbody>
</table>

geometry in the assembly of the time derivative. The work estimate for AB3LTS (6.4.2) underestimates the benefit of LTS when the number of time scales is \( \leq 5 \) and can overestimate the benefit when the number of time scales is greater than 5.

To explain the underestimation of speedup shown in Table 8.5, we profile the “Box Around Circle Refined” test case. We find that 38\% percent of the computation time is spent assembling the time derivative and 41\% is spent updating the mesh geometry. However, the computation of the time derivative is essentially matrix-vector products, which are very efficiently computed \[56, 57\]. A profile of the global time stepping Runge-Kutta method reveals that 26\% of computation time is spent evaluating the time derivative and 63\% is spent updating the mesh geometry. The update to the mesh geometry is always expensive regardless of time integrator, so it is advantageous to limit the number of times the procedure updates the mesh. Thus, we account for the underestimation in speedup because AB3LTS is not dominated as much by the mesh update as in LSRK3.

We explore why the work estimate becomes more accurate for meshes with a wide distribution of element sizes by profiling the “Circle Array” test case. We find that the update of the geometry of a moving element is 83\% of the computation time, whereas the evaluation of the time derivatives is 7\%. Also, the extra computational effort to create time interpolants for the LTS procedure becomes negligible on moving meshes, just one percent of computation time. For the global time stepping LSRK3, the profile shows that 84\% of computation time is spent updating the mesh geometry whereas only 7\% is spent assembling the time derivative. Thus, we account for the accuracy of the work estimate in the last two entries of Table 8.5.
8.2.3 Steady-State Calculation with NACA0012 Airfoil

Explicit local time stepping also reduces the computational work in a steady state calculation, but the LTS result fails to outperform implicit time integration methods, which have been shown to greatly accelerate steady-state calculations [11, 57, 76]. The LTS procedure has advantages over implicit methods though it is slower. Implicit methods accelerate convergence to steady state for the same NACA0012 airfoil problem by a factor of 67, but are difficult to code, may encounter memory issues in higher dimensions [57, 76, 93], and require preconditioners derived for each set of equations [92]. In contrast, the LTS method is portable to any set of equations.

We solve the steady inviscid flow past a fixed airfoil as a model problem. The governing equations for this compressible flow calculation are the Euler equations of gas dynamics, which we write in conservation law form

\[
\begin{bmatrix}
\rho \\
\rho u \\
\rho v \\
\rho E \\
\end{bmatrix}_t + \begin{bmatrix}
\rho u \\
p + \rho u^2 \\
\rho uv \\
u(\rho E + p) \\
\end{bmatrix}_x + \begin{bmatrix}
\rho v \\
p + \rho v^2 \\
\rho uv \\
v(\rho E + p) \\
\end{bmatrix}_y = 0. \tag{8.2.4}
\]

In (8.2.4) \( \rho \) is the density, \( u \) and \( v \) are the \( x \) and \( y \) components of the velocity, and \( E \) is the internal energy. The system (8.2.4) is enclosed by the equation of state for an ideal gas

\[
p = (\gamma - 1) \left[ E - \frac{1}{2} (u^2 + v^2) \right], \quad \gamma = \frac{C_p}{C_V} = 1.4. \tag{8.2.5}
\]

We prescribe free stream boundary conditions on the edges of the computational domain and a no-slip condition on the boundary of the airfoil. The tolerance that defines steady state is set to be \( 10^{-8} \) in the calculation. For the other parameters in the calculation we choose the polynomial order
on each element to be $N = 6$ in each direction, the free-stream Mach number $M = 0.3$, and the Courant number $1/3$. The computation is on the domain $[-20, 20] \times [-20, 20]$ with the NACA0012 airfoil placed at the origin.

We compare the CPU time for the explicit time integrators LSRK3 and AB3LTS as well as the rate of convergence to steady state. Table 8.6 shows a comparison in the CPU time required to reach steady state using LSRK3 and AB3LTS. The reduction in the amount of work is a factor of approximately 6.3. To compare the convergence rates of the two methods we present the logarithm of the normalized residual as a function of CPU time in Fig. 8.36.

Table 8.6: LSRK3 versus AB3LTS to progress a solution of the Euler equations (8.2.4) to steady state.

<table>
<thead>
<tr>
<th>Time Integrator</th>
<th>CPU Time (m)</th>
<th>speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSRK3</td>
<td>893.00</td>
<td>–</td>
</tr>
<tr>
<td>AB3LTS</td>
<td>141.46</td>
<td>6.3</td>
</tr>
</tbody>
</table>

### 8.3 Speedup of Parallel DGSEM in Two and Three Spatial Dimensions

In this section we provide examples that implement the parallel DGSEM outlined in Chap. 7 with a standard time integrator and the local time stepping algorithm in two and three spatial dimensions. Their purpose is to demonstrate that the approximation with LTS remains embarrassingly parallel. The test meshes chosen to demonstrate the scaling capabilities in the DGSEM approximation are rather small (< 7,000 elements) in comparison to meshes used in engineering computations, where the number of elements can be in the hundreds of thousands [6, 82]. In Sec. 8.3.1 we look at the speedup results in two dimensions and provide a shootout between the two methods with analysis. Then, Sec. 8.3.2 gives speedup of a three dimensional implementation.

#### 8.3.1 Quadrilateral Results

First we explore the speedup of the DGSEM on a static quadrilateral mesh. We use a mesh, pictured in Fig. 8.37, with 6248 elements.
Figure 8.36: Comparison of the convergence of AB3LTS and LSRK3 for steady flow over the NACA0012 airfoil.

**Standard Time Integrator.** We verify the assertion that our implementation of the DGSEM is embarrassingly parallel for a standard time integrator. We select a third order Runge-Kutta for the computations. We use the default load balancing strategy of assigning each process approximately the same number of elements. We approximate the solution of the wave equation with polynomial order $N = 6$ in each direction up to $T = 1.0$ with $\Delta t = 1/20,000$ in a homogeneous material with plane wave solution (8.1.7). The benefit of this simple, linear test problem is it will give worst case scenario results for speedup, because in a non-linear problem the physical and numerical fluxes are more expensive to compute.

We run the test problem on up to 200 processors and show the speedup in Fig. 8.38. At the high
Figure 8.37: Quadrilateral mesh used in two dimensional parallel speedup tests.

Figure 8.38: Speedup of two dimensional parallel DGSEM approximation.
end we see that the parallel implementation has 94% efficiency and see that adding more processors results in a linear reduction in computation time. At the low end of the number of processors in Fig. 8.38 we see that the speedup can be super linear. This is the result of cache effects. Cache effects are seen when a working set of data can fit into caches and, thus, memory access time is reduced dramatically. This results in extra speedup in addition to that from the actual parallel computation.

**Local Time Stepping Integrator.** Next we show that the DGSEM with local time stepping remains embarrassingly parallel. To balance the computational load we assign each element in the mesh Fig. 8.37 a weight according to size in METIS. For the LTS parallel speedup test we set the polynomial order to be $N = 6$ in each direction and solve the wave equation up to $T = 1.0$ with $\Delta t_0 = 1/1,250$ in a homogeneous material with plane wave solution (8.1.7).

We run the test problem up to 200 processors and show the speedup in Fig. 8.39. For a large number of processors we see that the parallel implementation of local time stepping has 93% efficiency. For a small number of processors we, again, see that speedup can be super linear.

![Figure 8.39: Speedup of two dimensional parallel DGSEM approximation with local time stepping.](image)
Shootout Between Parallel Standard and Local Time Stepping Integrator. We use the ‘Circle Array’ mesh from Fig. 6.4(c) to compare the reduction in computation time between a standard and local time stepping integrator. The number of processors is kept low at 12. The comparison functions as an example of how parallel LTS can offer significant speedup without the use of a high performance cluster. Both problems use $N = 6$ and integrate the wave equation up to the final time $T = 1.0$ with time step $\Delta t = 1/14,000$ and $\Delta t_0 = 1/1,000$ for the standard and LTS integrators respectively.

We plot the global and local time stepping speedups relative to their own serial implementation against the ideal in Fig. 8.40. Again, we see both implementations are embarrassingly parallel where cache effects can yield super linear speedup.

![Figure 8.40: Comparison of speedup for parallel global and local time stepping relative to their own serial implementation.](image)

For the shootout between the LTS and GTS implementations we note that the speedup of a parallel LTS implementation compounds with the natural speedup of the new time integrator. Recall we observed a factor of 9.7 speedup between global and local time stepping in Tab. 8.4. So, even though the parallel speedup properties of both time integrator implementations are similar (as seen in Fig. 8.40), the local time stepping offers significant speedup without a large number of processors.

124
processors. In Fig. 8.41 we present a shootout of the total speedup of a parallel, LTS implementation against a parallelized standard time integrator. For twelve processors the standard integrator has a speedup of 13.3, but the local time integrator has a speedup of \((9.7)(12.45) = 120.8\). Thus, the parallel LTS algorithm offers a competitive choice when one wants to solve large scale problems in parallel without the use of a high performance cluster.

![Speedup Graph](image)

Figure 8.41: A shootout between the parallel implementations of a standard time integrator against the local time stepping integrator.

### 8.3.2 Hexahedral Results

Next we examine the speedup of the DGSEM on a static hexahedral mesh. We use a simple extrusion of the Circle Array mesh, shown in Fig. 8.42, with 4890 elements. We approximate the solution to the wave equation by a polynomial of degree \(N = 6\) is each direction. We integrate the plane wave (8.1.7) propagating in a homogeneous material up to \(T = 1.0\) with \(\Delta t = 1/20,000\) for the standard time integrator and \(\Delta t_0 = 1/1,300\) for the LTS method.

**Standard Time Integrator.** We, again, see that the three dimensional DGSEM implementation is embarrassingly parallel with Williamson’s third order Runge-Kutta integrator [94]. We assign each processor approximately the same number of elements, for load balancing. We run
the test problem on up to 200 processors and show the speedup in Fig. 8.43. At the low end of number of processors we again see the possibility of super linear speedup because of cache effects. At the high end we see that the parallel implementation has 94% efficiency and see that adding more processors may result in a linear reduction in computation time.

**LTS Integrator.** The three dimensional DGSEM with local time stepping also remains embarrassingly parallel. The hexahedral elements are weighted according to size to help balance the computational load between processes. The speedup test problem is run up to 200 processors and results are shown in Fig. 8.44. For a large number of processors we see that the parallel implementation of local time stepping has 93% efficiency. For a small number of processors we, again, see the benefit of cache effects when the number of processors is below 30.
Figure 8.43: Speedup of three dimensional parallel DGSEM approximation.

Figure 8.44: Speedup of three dimensional parallel DGSEM approximation with local time stepping.
CHAPTER 9

SUMMARY AND CONCLUSIONS

In this dissertation we provided background for the physical model and derived and evaluated exact, upwind numerical fluxes for wave propagation problems in moving materials. These new numerical fluxes were used in the discontinuous Galerkin spectral element approximation with an arbitrary Lagrangian-Eulerian moving mesh mapping to compute the plane wave reflection and transmission from moving material interfaces. We applied the approximations to problems in electromagnetics and acoustics. We also derived a local time stepping strategy to reduce stiffness introduced by mesh refinement. We demonstrated that we can integrate groups of elements with different time scales while retaining design time accuracy. The local time stepping method operates on static or moving meshes. Also, we derived estimates for how useful the local time stepping integrator would be for different spectral element meshes and distributions of element sizes. Finally, we examined the parallelization of the local time stepping method in two and three spatial dimensions.

We presented time-step refinement studies with moving material interfaces and meshes to validate the numerical fluxes and overall approximation. The wave propagation test problems included electromagnetic Gaussian plane waves, acoustic plane waves, and free-stream preservation to ensure the mesh motion did not introduce spurious waves. We also verified, theoretically and numerically, limiting cases of the new numerical fluxes, e.g. the moving material interface becomes a moving perfectly reflecting mirror as the material property tends to zero. Additional test problems demonstrated that the local time stepping integrator retains design time accuracy and spectral accuracy in space for moving (and static) quadrilateral and hexahedral meshes. The approximations were also tested for speedup in parallel implementations, where we saw that the local time stepping method scales in a similar way to DGSEM with a standard time integrator, i.e., the local time stepping algorithm was shown to be embarrassingly parallel.

There is still work to be done. In particular, a parallel implementation of the moving mesh formulation of the DGSEM. In the moving mesh approximation there is weak coupling in the DGSEM approximation as well as coupling at mesh nodes used to compute the mesh velocities.
in the spring-dashpot physical analogy. Also, we used an intuitive mesh partitioning strategy of weighting elements in a mesh by size. This load balancing tactic offered efficient speedup. However, other strategies to balance the load and maintain speedup might be necessary on larger scale problems such as limiting communication between processes [36, 82].

9.1 Conclusions

Explicitly, the main contributions of this dissertation to the computational wave reflection and transmission from moving material interface problems are:

- We derive numerical fluxes at moving material interfaces for both electromagnetic problems and acoustic problems.
- We develop and analyze a local time stepping integrator to reduce problem stiffness that can be used on moving meshes.
- We demonstrate that the approximations can be parallelized including the newly developed local time stepping strategy.
- We compute analytical solutions for reflection and transmission problems with a Gaussian incident wave.

In closing, from the examples presented, the time-step convergence study, and the theoretical description presented we conclude that the DGSEM using an ALE formulation to compute wave reflection and transmission on moving meshes offers an effective way to solve more complex problems where analytical solutions cannot be provided.
REFERENCES


[41] Han, J. Y. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18th annual ACM symposium on User interface software and technology* (2005), ACM, pp. 115–118.


133


BIographical Sketch

Andrew was born February 16, 1987 in Oregon. He earned a B.Sc. degree in Mathematics (2005-2009) at Washington State University in Pullman, WA. He then enrolled in the doctoral program of applied and computational mathematics at Florida State University in the Fall of 2009. He received his Master’s degree while en route to his Ph.D. Andrew also had the opportunity to take a break from his doctoral projects and work at Los Alamos National Laboratory in New Mexico. While at FSU Andrew worked as a teaching assistant and taught several undergraduate mathematics classes. After graduation, he intends to continue with postdoctoral research in high-order methods, wave propagation, and moving meshes.