UNIVERSITY OF CALIFORNIA, SAN DIEGO

Analysis of Some Higher Order Space-Time Moving Finite Element Methods

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Mathematics

by

Maximilian Sloan Metti

Committee in charge:

Professor Randolph E. Bank, Chair
Professor Michael Holst
Professor Petr Krysl
Professor Julius Kuti
Professor Melvin Leok

2013

The dissertation of Maximilian Sloan Metti is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____
Chair

University of California, San Diego

2013

EPIGRAPH

Be careful to leave your sons well instructed rather than rich,
for the hopes of the instructed are better than the wealth of the ignorant.
—Epictetus

TABLE OF CONTENTS

Signature Page iii

Epigraph iv

Table of Contents v

List of Figures viii

List of Tables xiii

Acknowledgements xv

Vita and Publications xvii

Abstract of the Dissertation xviii

Chapter 1 Introduction 1
1.1 Discretization of time-dependent problems 2
1.2 Moving finite elements 4
1.2.1 A history of moving finite element methods 5
1.2.2 Error analysis of moving finite elements 6
1.3 Layout of the thesis 8

Chapter 2 The Differential Equation 10
2.1 The strong formulation of the linear convection-diffusion-reaction equation 12
2.2 A variational formulation 14
2.2.1 A variational formulation with non-autonomous domain parametrization 16

Chapter 3 A Space-Time Tensor Finite Element Space 22
3.1 The description of a moving finite element mesh 25
3.1.1 The time and space discretizations 26
3.1.2 Example in one spatial dimension (d=1) 27
3.1.3 Mesh discontinuities 27
3.1.4 The reference elements 28
3.2 The isoparametric map 31
3.2.1 Example in one spatial dimension (d=1) 31
3.2.2 The isoparametric map in higher dimensions . . . 38
3.3 Shape regularity of the finite element space 40
3.3.1 Shape regularity in space and time 40

3.3.2 Restrictions on the mesh motion (shape regularity in space-time) . . . 41

Chapter 4 Preliminary Results and Approximation Properties of the Finite Element Space . . . 45
4.1 Norms and Notation . . . 46
4.1.1 Space Norms . . . 46
4.1.2 Bounding the bilinear form . . . 48
4.1.3 Time norms . . . 49
4.1.4 Collocation nodes . . . 51
4.1.5 The energy norm . . . 52
4.2 Preliminary results for moving finite element spaces . . . 52
4.2.1 Shifting finite element functions . . . 53
4.2.2 The shifted characteristic derivative . . . 57
4.3 Approximation properties of the moving finite element space . . . 58
4.3.1 Defining the finite element interpolant . . . 58
4.3.2 Interpolation error bounds for the finite element spaces . . . 60

Chapter 5 A Space-Time Moving Finite Element Method . . . 67
5.1 The finite element formulation . . . 69
5.2 A discrete Galerkin orthogonality . . . 71
5.2.1 A closer look at the time discretization . . . 72
5.2.2 A one parameter family of non-truncating quadrature rules . . . 73
5.3 Well-posedness of the finite element formulation . . . 75
5.3.1 The semi-discrete linear system . . . 75
5.3.2 The fully discrete linear system . . . 77
5.3.3 Invertibility of the linear system . . . 80
5.4 Two discrete Grönwall inequalities . . . 82
5.5 Error estimates of the finite element solution . . . 86
5.5.1 A symmetric error estimate . . . 86
5.5.2 An a posteriori error estimate . . . 90
5.5.3 Closing remarks . . . 91

Chapter 6 Space-Time Moving Finite Elements with Runge-Kutta Methods for Time Integration . . . 92
6.1 A brief overview of some Runge-Kutta schemes . . . 93
6.2 A space-time moving finite element method with Runge-Kutta time integration . . . 96
6.2.1 The importance of stability . . . 97

6.3 The associated linear system and well-posedness . . . 98
6.3.1 Well-posedness of the TR-BDF scheme . . . 99
6.4 Error analysis for TR-BDF . . . 100

Chapter 7 Numerical Methods and Experiments with Moving Meshes . . 109
7.1 Two test problems . . . 110
7.2 Moving meshes and static meshes . . . 112
7.3 Non-truncating time integration and other collocation nodes . . . 115
7.4 Mesh Discontinuities: $\mathcal{L}_2$-projection and interpolation . . 121
7.5 Moving finite elements with spatial adaptivity . . . 125
7.5.1 An h-adaptive method . . . 131
7.5.2 Time discretization . . . 132
7.5.3 Experiments with h-adaptivity in space . . . 135
7.6 Moving finite elements and Burger's equation . . . 140
7.6.1 Experiments on Burger's equation . . . 141

Chapter 8 Conclusion . . . 149

Appendix A Final Notes . . . 154
A.1 Tensor quadratic elements with one spatial dimension . . 155
A.1.1 Mid-step perturbation of a trapezoidal element . . 155
A.1.2 End-step perturbation of an extrapolated trapezoidal element . . . 157
A.2 Tensor product linear elements of two spatial dimensions 159

Bibliography . . . 161

LIST OF FIGURES

Figure 3.1: An example space-time mesh $\mathcal{T}_{h,i}^p$ on a partition with $d = 1$ and $p = 2$. The filled circles represent the space-time "hat" basis functions; hollow circles correspond to basis functions that are the product of a "bump" basis function. 28
Figure 3.2: An illustration of e_{ref} with degrees of freedom shown when $d = 2$ and $p = 1$. 30
Figure 3.3: A twisting element with $d = 2$ and $p = 1$ plotted at discrete time slices. The spatial nodes are labeled and their linear trajectories are represented by the dashed lines and superimposed on each time slice. From the time $t = 0$ to $t = 1$, the element seems to merely rotate; however, due to the linear paths traced out by the spatial nodes, the element degenerates when $t = 1/2$. 43

Figure 7.1: Above is the solution to the equation described in (7.1). We have a bell-curve initially centered at $x = 0$ that gradually shifts to the right as t increases. The solution is plotted at times $t = 0, .0.2, 0.4, 0.6, 0.8$, and 1. 111
Figure 7.2: The solution to equation (7.2) is plotted at times $t = 0, .0.2, 0.4, 0.6, 0.8$, and 1. 112
Figure 7.3: An example of a moving mesh given by the method of characteristics for equation (7.1). In this example, we have $m = 5$ time steps and initialize each time partition with $n = 11$ spatial nodes. As a result of using the method of characteristics, the spatial nodes on the interior of the domain satisfy $x_t = 3$; some spatial nodes have been deleted near the outflow boundary and inflow boundary. The hat nodes and the bump nodes are both displayed and the bump nodes are always chosen to be the midpoint of the element's hat nodes. The vertical axis corresponds to the time dimension and the horizontal axis corresponds to the spatial dimension. 113
Figure 7.4: The mesh generated by the method of characteristics applied to (7.2). Though the curvature is subtle, these mesh trajectories are quadratic and approximately satisfy the evolution equation $x_t = 0.1(x^3 - 9x)$. In this example, we take $m = 5$ time steps and initialize each partition with $n = 11$ spatial nodes. The hat nodes and the bump nodes are both displayed and the bump nodes are always chosen to be the midpoint of the element's hat nodes. The vertical axis corresponds to the time dimension and the horizontal axis corresponds to the spatial dimension. 113

Figure 7.5: The $\mathcal{L}_2$-errors for problem (7.1) plotted on a logarithmic scale. The blue line with filled diamonds corresponds to the basis node $\varepsilon = 1/2$. The red line with hollow triangles corresponds to the basis node $\varepsilon = 2 - \sqrt{2}$. The green line with crosses corresponds to the basis node $\varepsilon = 2/3$. The errors are almost exactly the same in this problem, with a slight gain for $\varepsilon = 2 - \sqrt{2}$. This suggests that non-truncating quadrature rules are not the only quadrature rules that can be used to define the time stepping scheme and still maintain second order convergence with respect to the time discretization. 119

Figure 7.6: The $\mathcal{L}_2$-errors for problem (7.2) plotted on a logarithmic scale. The blue line with filled diamonds corresponds to the basis node $\varepsilon = 1/2$. The red line with hollow triangles corresponds to the basis node $\varepsilon = 2 - \sqrt{2}$. The green line with crosses corresponds to the basis node $\varepsilon = 2/3$. This problem shows a clearer advantage in selecting $\varepsilon = 2 - \sqrt{2}$ compared to the results shown in figure 7.5. 120

Figure 7.7: The $\mathcal{L}_2$-errors for problem (7.1) plotted with respect to the number of time steps. For this plot, each discretization employs 101 spatial nodes. As the number of time steps increases, the time discretization is refined, rather than extending the time domain. The most striking feature here is that the $\mathcal{L}_2$-error does not decrease monotonically when using a moving mesh and interpolation at the mesh discontinuities. The plot suggests that using $\mathcal{L}_2$-projection maintains the benefit of smaller error of fewer time steps, as compared to using a static mesh, though it avoids the build-up of error when more time steps are used. . . 128

Figure 7.8: The $\mathcal{L}_2$-errors for problem (7.1) plotted with respect to the number of time steps. For this plot, each discretization uses 1001 spatial nodes. Unlike the case where only 101 spatial nodes are used, the interpolation error does not buildup as the number of time steps increases when using a moving mesh; the benefits of using $\mathcal{L}_2$-projection are not significant compared to using interpolation, and the increase in CPU time is more dramatic as the mesh is more refined than the case when 101 spatial nodes are used. 129

Figure 7.9: The $\mathcal{L}_2$-errors for problem (7.2) plotted with respect to the number of time steps. For this plot, each discretization employs 101 spatial nodes. As the number of time steps increases, the time discretization is refined, rather than extending the time domain. The smallest $\mathcal{L}_2$- error of the computed is attained by using a moving mesh with $\mathcal{L}_2$-projection at the mesh discontinuities, though the slight reduction in error over using interpolation at the mesh discontinuities might not justify the more prominent increase in CPU time. 130
Figure 7.10: The $\mathcal{L}_2$-errors for problem (7.2) plotted with respect to the number of time steps. For this plot, each discretization uses 1001 spatial nodes. In comparison to figure 7.9, there are virtually no gains in using 1001 spatial nodes versus 101 nodes for this problem. 130
Figure 7.11: An example of a spatially adaptive moving mesh, given by the method of characteristics for equation (7.1). In this example, we have $m = 10$ time steps and initialize each time partition with $n = 51$ spatial nodes. As a result of using the method of characteristics, the spatial nodes on the interior of the domain satisfy $x_t = 3$; some spatial nodes have been deleted near the outflow boundary and inflow boundary. Comparing the density of the spatial nodes to the figure of the solution in figure 7.1, the spatial resolution is higher where the solution is changing rapidly. Since the convection term aligns the mesh with the solution of the differential equation, the nodes of the mesh trace out level sets of the solution in time (except near the boundaries), which reduces the impact of the discontinuities between time partitions. 133
Figure 7.12: An adaptive mesh generated by the method of characteristics applied to (7.2). Though the curvature is subtle, these mesh trajectories are quadratic and approximately satisfy the evolution equation $x_t = 0.1(x^3 - 9x)$. In this example, we take $m = 10$ time steps and initialize each partition with $n = 51$ spatial nodes. In this problem, the convection term does not align with the motion of the solution in time; mesh discontinuities are more prominent than in figure 7.11, though they are still an improvement over the mesh depicted in figure 7.4, where the mesh configuration is reconfigured to be uniform on each time partition. 134

Figure 7.13: The $\mathcal{L}_2$-errors for problem (7.1) plotted with respect to the number of time steps. For this plot, each discretization employs 101 spatial nodes. The solutions computed on adaptively generated meshes display significantly higher errors. In comparing the error to the number of nodes at the end of the simulation, it is clear that predictor for the number of spatial nodes requires adjustment. 138
Figure 7.14: The $\mathcal{L}_2$-errors for problem (7.1) plotted with respect to the number of time steps. For this plot, each discretization uses 1001 spatial nodes. The solutions found by adaptive meshing show a great improvement in terms of their overall performance. The fact that the final errors do not shrink as much as in the non-adaptive cases may be a result of the error indicator staying relatively constant throughout the simulation, even as the bump structure is exiting the right boundary (see figure 7.1). 138
Figure 7.15: The $\mathcal{L}_2$-errors for problem (7.2) plotted with respect to the number of time steps. For this plot, each discretization employs 101 spatial nodes. Table 7.10 reports that the adaptive meshing scheme adds many spatial nodes to the mesh, though we see that the error is not significantly reduced. The adaptive method is over-solving the problem. 139
Figure 7.16: An illustration of the initial condition used for Burger's equation. As the PDE propagates the solution in time, the structure on the left side should move toward the right, creating a thin shock layer that propagates from the middle of the domain toward the right boundary. 141
Figure 7.17: An example of a moving mesh, given by the method of characteristics for equation (7.3). In this example, we have $m = 25$ time steps and initialize each time partition with $n = 61$ spatial nodes. As a result of using the method of characteristics, the spatial nodes on the interior of the domain approximately satisfy $x_t = u(x)$. Comparing the density of the spatial nodes to the figures of the computed solutions in figures 7.18 and 7.19, the spatial nodes properly congregate near the shock layer and track it throughout the simulation. The trailing node near the left boundary is the bump node on the leftmost element. Its motion is slower as it must bisect the hat nodes, one of which is fixed on the boundary. 143

Figure 7.18: Computed solutions for equation (7.3) with $R = 100$ using a static mesh (top) and a moving mesh (bottom). The meshes are initialized to have 61 spatial nodes and 10 time steps are used to advance the solution to time $t = 2$. In both cases, the time step is too large and the computed solutions form artificial structures near and within the shock layer. 144
Figure 7.19: Computed solutions for equation (7.3) with $R = 100$ using a static mesh (top) and a moving mesh (bottom). The meshes are initialized to have 61 spatial nodes and 25 time steps are used to advance the solution to time $t = 2$. The solution computed using moving meshes has dampened the oscillatory behavior occurring near the shock layer, though the solution computed using a static mesh still displays this undesirable behavior. . . . 145
Figure 7.20: In this simulation, the Reynolds number is set to 1000, causing a thinner shock layer. Solutions were computed using $m = 100$ time steps and initialized with $n = 301$ spatial nodes. The static mesh in the top figure displays the usual artificial oscillations that occur with finite element solutions near the shock layer, whereas the moving mesh has suppressed this behavior. 146
Figure 7.21: The computed solution and its mesh using the adaptive method of section 7.5.1. The adaptive method effectively coarsens the mesh where the solution is unchanging, while maintaining a high resolution near the shock layer. The computed solution still displays small oscillations, though the mesh generated by the adaptive method seems satisfactory. Mesh grading near the shock layer could potentially dampen these oscillations. 147

Figure A.1: Three tensor product quadratic elements. The element labeled (a) has not been perturbed at the middle collocation node. Element (b) has been perturbed at the middle collocation node, though it remains non-degenerate. Element (c) is an example of an element that degenerates because the middle collocation nodes move too close to each other. Notice how the intersection does not necessarily occur at a collocation time slice. 156
Figure A.2: Three tensor product quadratic elements. Element (a) is an element that has can be successfully extrapolated from the mid-step collocation node. Element (b) has been perturbed at the end collocation node, though it remains non-degenerate. The third element (c) is an example of an element that cannot be extrapolated; such elements should not be permitted into time partitions that are built one collocation time slice at a time. . . 158

LIST OF TABLES

Table 7.1: A comparison table for problem (7.1) between static mesh methods and moving mesh methods using interpolation. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative increase in CPU time of the moving mesh solution to the static mesh solution. Moving the mesh leads to clear advantages when the spatial mesh is sufficiently refined, though buildup in error occurs when the time discretization is too refined. 116
Table 7.2: A comparison table for problem (7.2) between static mesh methods and moving mesh methods using interpolation. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative increase in CPU time of the moving mesh solution to the static mesh solution. 117
Table 7.3: The error table for problem (7.1). The bottom row is the ratio of the differences of the log errors and the log of the number of time steps; the error displays second order convergence. 119
Table 7.4: The error table for problem (7.2). The bottom row is the ratio of the differences of the log errors and the log of the number of time steps; the error displays second order convergence. 120
Table 7.5: A comparison table for problem (7.1) of using interpolation instead of $\mathcal{L}_2$-projection at the mesh discontinuities. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative speedup. It is clear that $\mathcal{L}_2$-projection stabilizes the error when the time-discretization is highly refined compared to the spatial discretization, though there are significant saving in CPU time when the spatial mesh is sufficiently refined. 123
Table 7.6: A comparison table for problem (7.2) of using interpolation instead of $\mathcal{L}_2$-projection at the mesh discontinuities. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative speedup. The effects of interpolation are not nearly as present for this problem. 124
Table 7.7: A comparison table for problem (7.1) between static mesh methods and moving mesh methods using $\mathcal{L}_2$-projection. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative increase in CPU time of the moving mesh solution to the static mesh solution. It is clear that $\mathcal{L}_2$-projection stabilizes the error for coarse spatial meshes. 126

Table 7.8: A comparison table for problem (7.2) between static mesh methods and moving mesh methods using $\mathcal{L}_2$-projection. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative increase in CPU time of the moving mesh solution to the static mesh solution. 127
Table 7.9: This table reports the count of spatial nodes in the mesh at the end of the simulation using adaptive meshing. The problem solved is (7.1) and these simulations were initialized using 101 (top) and 1001 (bottom) spatial nodes. The final count of nodes is expected to be low as the bump structure in the solution is exiting through the right boundary, leaving a very flat solution on the majority of the spatial domain. 136
Table 7.10: This table reports the count of spatial nodes in the mesh at the end of the simulation using adaptive meshing. The problem solved is (7.2) and simulations were initialized using 101 (top) and 1001 (bottom) spatial nodes. The final count of nodes is rather high since the errors displayed in figure 7.15 are not significantly reduced; this indicates that our mesh selection scheme needs improvement. 137

ACKNOWLEDGEMENTS

I extend my sincere gratitude to my advisor, Professor Randolph E. Bank, for his unwavering support and dedication to this project. Thank you for all of your advice, your time, and a welcoming open door that always encouraged my intellectual curiosity. Your instruction has equipped me with a great education and deep desire to continue learning. I have really enjoyed my time working with you. Thank you, Professor.

Chapter 1, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 2, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 3, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 4, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 5, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 6, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 7, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 8, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

I would also like to acknowledge my close colleagues in the UCSD mathematics department; thank you to Adam Mihalik for all of our thought provoking discussions relating to finite elements and academia; thank you to Angelo Scandaliato for entertaining my long monologues about my research and insights from your experiences with mathematical research; and thank you to Stefanos Poulis, who has always presented me with great opportunities and pushed me to realize my maximum potential. I am glad to have spent my graduate studies with you and look forward in anticipation of where our life paths will take us, and where they may converge again.

Furthermore, my family have provided me with the greatest support that any student could request. To my father, Michael Metti, I truly appreciate the encouraging willingness to listen to my ideas and your supportive perspective on my research and career. To my mother, Hannah Ligné-Metti, thank you for your emotional support and stoic-at-heart philosophy to help me manage my work and myself. To my brother, Sebastian Metti, for providing me with such intellectual stimulation and encouragement to recognize my greater potential in my career and my personal life. To Christine Kwong, the support you have given me during this final year of graduate school is immeasurable; I cannot thank you enough for all of the time and effort that you dedicated to help me reach my goals.

VITA and PUBLICATIONS

2008	B. A. in Applied Mathematics, University of California, San Diego
2008-2010	M. A. in Applied Mathematics, University of California, San Diego
2010-2013	Ph. D. in Mathematics, University of California, San Diego
2010-2013	Graduate Teaching Assistant, University of California, San Diego
2011-Present	Analytics Architect, Premlo Care Intuitive, Del Mar
2011-2012	Student Leader, UCSD MathStorm Consulting Group, La Jolla
2012-Present	Senior Statistician, Petco Animal Supplies, San Diego

PUBLICATIONS

R. E. Bank, M. S. Metti, *An error analysis for some higher order space-time moving finite element methods*, manuscript in preparation.

R. E. Bank, M. S. Metti, *Generalized time integration schemes for space-time moving finite elements*, manuscript in preparation.

ABSTRACT OF THE DISSERTATION

Analysis of Some Higher Order Space-Time Moving Finite Element Methods

by

Maximilian Sloan Metti

Doctor of Philosophy in Mathematics

University of California, San Diego, 2013

Professor Randolph E. Bank, Chair

This is a study of an application of finite element methods designed for convection-dominated, time-dependent partial differential equations. Specifically, this work analyzes finite element discretizations that employ moving meshes in order to solve linear differential equations over space-time domains. These methods can lead to significant savings in computation costs for problems having solutions that develop steep moving fronts, as moving meshes have the ability to track these fronts continuously with a high concentration of nodes; this flexibility allows for much larger time steps than standard tensor product finite elements, while maintaining high resolution of fine structures that sweep through the spatial domain.

The main results are a priori and a posteriori error bounds for some moving

finite element methods of high order and general time-stepping schemes. These finite element methods follow a method of lines approach for propagating the solution in time, though the error analysis places a strong emphasis on the properties inherited by the finite element aspects of the discrete problem. Another focus of this work is to determine practical and efficient schemes for adaptive meshing and mesh motion. As a result of this research, a solver has been written in C++ that is applicable to time-dependent linear convection-diffusion-reaction equations with a single dimension for the spatial.

Chapter 1

Introduction

Computing accurate solutions to convection-dominated partial differential equations using standard finite element methods can be computationally expensive, and sometimes prohibitively so. Consequently, the use of adaptive methods can lead to great savings in computation time and maintain accuracy of the computed solution ([17],[19],[9],[57]). It is often the case that regions in which the solution to a partial differential equation (PDE) is rough or rapidly changing are relatively small compared to the overall domain and adaptive methods leverage this fact by focusing more computational effort by placing a higher concentration of the degrees of freedom in these regions and avoiding "over-solving" where the solution is smooth [19]. Effectively, adaptive methods are designed to automate the process of finding a finite element space that is well-suited to solving a given differential equation. For elliptic problems, many adaptive methods can be described as some combination of three basic adaptive processes ([9],[45],[17]): *h-methods* locally refine or coarsen an existing mesh [9]; *r-methods* move the nodes in an existing mesh [20]; and *p-methods* locally raise and lower the order of an approximation [11]. Typically, adaptive methods are iterative procedures, where an approximate solution is computed and then used to find a better suited approximation space to the differential equation, which is in turn used find a more accurate approximate solution and the process repeats. In this thesis, adaptive methods for space-time problems are explored in which approximate solutions to the PDE are computed using a moving mesh.

1.1 Discretization of time-dependent problems

To start, we introduce two common approaches for solving time-dependent problems and then discuss how adaptive methods can be applied to these approaches. The first approach is the *method of lines* approach. In this approach, one discretizes the spatial domain using a standard finite element triangulation, which gives a system of ordinary differential equations (ODEs) that is equal in length to the number of degrees of freedom assigned to the spatial discretization. Then, numerical time integration techniques are used to advance the system in time [44]. The method of lines naturally separates the discretization of space and

time, which can be convenient for error analysis.

A second approach is the direct discretization of the space-time domain that treats the time dimension as an additional spatial dimension. Therefore, one discretizes the space-time domain as a higher-dimensional spatial domain. The software package for solving elliptic equations PLTMG [16] uses this approach to discretize Burger's equation, for example. The direct discretization approach consequently falls into the framework of finite element methods for autonomous differential equations for which there is extensive research; books [7] and [52] are useful references.

The benefits of the method of lines approach is that it allows a solution to be computed rather quickly, as we are essentially solving a system of ODEs, which means that the computational complexity is proportional to the number of time steps, given a fixed spatial discretization. In the case of one-step time integration schemes, the system of ODEs effectively breaks up into m disjoint problems that must be solved sequentially.

This is in contrast to the space-time discretization approach, which requires computing the solution over the entire space-time domain simultaneously; this leads to a much larger linear system to solve than the method of lines. The benefit of a direct discretization of the space-time domain, however, is that the resulting methods readily live within a pure finite element framework, unlike the method of lines, which means that finite element theory can be readily applied.

Finite element theory provides a convenient framework for proving *quasi-optimal* error bounds for the computed solution. To describe what quasi-optimal error bounds are, let u be the solution to some differential equation and u_h be a computed solution living in a finite element space, $\mathcal{V}_h^p$. Then, the computed solution u_h is quasi-optimal if there exists a norm $\|\!| \cdot |\!\|$ in which the error $u - u_h$ satisfies the symmetric error estimate

$$\|\!| u - u_h |\!\| \leq C \inf_{v \in \mathcal{V}_h^p} \|\!| u - v |\!\|. \tag{1.1}$$

This is an *a priori* error bound and is commonly sought for elliptic problems, but we can choose an energy norm that is also capable of measuring error introduced by the time discretization ([37],[38],[18],[40]). Quasi-optimality of u_h in the sense

of (1.1) tells us that the computed solution is comparable to the best possible solution in the finite element space $\mathcal{V}_h^p$ when measured in the energy semi-norm $|\!|\!| \cdot |\!|\!|$.

This thesis studies adaptive techniques for solving time-dependent problems. For direct space-time discretizations, the h, p, and r methods are naturally extended to the space-time mesh in the obvious way. To improve the performance of method of lines approaches, it is possible to apply the aforementioned adaptive methods to the discretizations of the spatial domain and employ adaptive time stepping to improve efficiency in the time discretization ([47],[33]). Additionally, periodic re-meshing is an option that can help track the region where the solution is rough in order to make this approach viable in instances where this region moves. Unfortunately, if the convection velocity in the differential equation is large, steep moving fronts may develop in the solution and very short time steps may be required to maintained a desired level of accuracy ([44],[57]). In order to avoid these short time steps, a moving mesh can be used to continuously track this moving region. The use of moving meshes for these purposes have been studied and tested on a variety of problems ([57],[55],[12],[13],[14],[28],[29],[50],[56],[65]) and demonstrated great success for some convection dominated problems. The adaptive methods studied in this thesis are designed to make use moving meshes and efficiently compute accurate solutions to differential equations that potentially have large convection velocities.

1.2 Moving finite elements

As suggested above, moving finite element methods follow a method of lines approach and are designed to efficiently and accurately solve convection-dominated problems, where standard finite element methods often lead to computational inefficiencies [57]. As regions where the solution is rough can shift in time-dependent problems, standard finite element methods require short time steps to track this behavior. In the case of hyperbolic problems, this is characterized by Courant-Friedrichs-Levy conditions, where the Courant number must be bounded to ensure

stability [44]. (The Courant number is defined by $|b|\Delta t/\Delta x$, where b represents the convection velocity, Δt is the length of the time step, and Δx is the maximum diameter of the spatial elements.) For convection-diffusion problems, the time step constraint is even stronger as the Courant number must be proportional to the diameter of the spatial elements.

In contrast, moving finite element methods are designed to track these moving regions with continuously moving nodes in order to preserve fine structures that are characteristic of solutions to convection-dominated problems, such as steep moving fronts, and this feature provides the flexibility for taking significantly longer time steps when performed properly ([57], [55], [28], [56]). Here, the word "properly" is rather ambiguous and often refers to aligning the mesh motion with the motion of the structures formed in the solution. There are many different approaches to generating this mesh motion, and each approach warrants its own moving finite element method.

1.2.1 A history of moving finite element methods

Moving finite elements were initially proposed by Miller and Miller in ([57], [55]) and have been analyzed and implemented in the context of linear and nonlinear problems. The initial schemes for computing solutions on moving meshes created a discrete problem where the PDE and the motion of the mesh nodes were computed simultaneously via residual minimization techniques. However, this approach inherently led to singularities such as node entanglement — when trajectories of the spatial nodes cross each other — and a "parallelism singularity," which arises from a non-uniqueness of the mesh parameterization [12]. In an effort to avoid these singularities, spring functions and viscosity functions were added as penalty terms to ensure shape regularity of the moving mesh and improve the conditioning of the discrete system. Numerous schemes to determine the motion of the mesh nodes have been proposed ([57], [55], [28], [56], [50]) that often add regularization terms, as mentioned above, or sophisticated variational formulations to avoid these singularities and enforce some desired behavior of the mesh motion. The success of the meshing scheme appears to depend greatly on the compatibil-

ity of the mesh motion with the behavior of the solution to the PDE, with some methods proving to be more robust than others. The book by Baines [12] serves as an excellent overview of moving finite element analysis and provides practical methods for computing moving finite element solutions, including an algorithm for finding "moving best fits" which decouples finding the solution to the PDE from determining the mesh motion by an iteratively adaptive process. One technique that provides greater flexibility for moving finite elements is the "graph massage" [50], which employs (discontinuous) re-meshing at discrete time steps to add and remove nodes in the mesh and avoid element tangling.

1.2.2 Error analysis of moving finite elements

The first error analysis of moving finite element methods is given in Dupont [37] and a symmetric error bound of the form (1.1) is proven for the semi-discrete formulation. To prove the quasi-optimality of moving finite element methods, a mesh-dependent energy (semi-) norm is typically defined, which follows the idea originally proposed by Dupont [37]. Symmetric error bounds are proven for linear moving finite elements by Bank and Santos in ([18],[65]) and by Dupont and Mogultay [40]. Some symmetric error estimates for mixed methods that use moving meshes are proven in [53].

The a priori bound for more general moving finite elements presented in this thesis are based on arguments from ([18],[65],[40]). In addition, *a posteriori* error estimates are provided in the current analysis to determine the convergence of the finite element solution to the true solution of the PDE. These a posteriori error estimates are derived from the quasi-optimality of the finite element solution and the approximation properties of the tensor product finite element space.

In this thesis, moving finite element methods are applied to linear parabolic PDEs. In the case of nonlinear equations, iterative methods using linearization techniques are often used and, thus, the error analysis may be applied to the linearized system, where additional errors coming from the nonlinear solver are bounded by some regularity conditions placed on the nonlinear terms [38]. The moving finite element space in this work is viewed as the tensor product of $\mathcal{C}^0$

finite element spaces over the spatial domain with a potentially discontinuous finite element discretization of the time domain. This allows us to analyze these space-time methods in the context of finite element error analysis and prove quasi-optimal a priori error bounds.

The elements of this discretization are the tensor product of isoparametric elements in space with an isoparametric element in time. For example, in two dimensions, this gives space-time elements that are curvilinear prisms with flat triangular faces at the beginning and end of a time step and the curved trapezoidal faces connecting them that correspond to the motion of the mesh in time. Minimal assumptions on the mesh motion are assumed to ensure that the error analysis can be readily applied to a wide variety of methods for determining mesh motion.

Time stepping methods

As moving finite elements follow a method of lines approach, the finite differencing scheme chosen to advance the solution in time plays a crucial role in determining the efficacy of the method. We restrict our attention to one-step time integration schemes, due to the nature of the finite element spaces described in this thesis. An important point that must be addressed by any discretization following the method of lines is that the semi-discrete formulation of the PDEs is often a stiff systems of ODEs [4] and, therefore, it is necessary to use *stable* time stepping methods ([68],[54]). Stable methods are necessary because they guarantee that the behavior of the finite element solution matches that of the true solution and avoid divergence and numerically-induced oscillations of the computed solution in time. Unfortunately, only time stepping methods that correspond to polynomial differentiation allow these moving mesh methods to correspond exactly to a finite element space, whereas other time stepping methods may be stable and more efficient, but only produce an approximation of the finite element solution. The tradeoff here is the symmetric quasi-optimal error bound for computational efficiency and stability. For quadratic moving finite elements, the TR-BDF2 time stepping scheme, proposed in [23], is considered as an example and an error analysis is provided. Following the error analysis of numerical ODE methods, the error of

the method must now be bounded by additional terms corresponding to truncation error of the time stepping method.

1.3 Layout of the thesis

The remainder of this thesis is organized as follows. In chapter 2, the partial differential equation that we use in this analysis is introduced, along with various formulations relevant to our study. One development in this chapter is a variational formulation that incorporates a time dependent parametrization of the spatial domain, which naturally fits into the framework of moving meshes. Chapter 3 contains a description of the tensor product moving finite element mesh, the construction of the proposed finite element space via a family of isoparametric maps, and some shape regularity assumptions for the moving mesh. This chapter describes finite element spaces that consist of piecewise tensor product polynomials of arbitrary order, whereas previous research has been restricted to the linear case. One benefit of these higher order finite element spaces is that they provide greater flexibility in the motion of the nodes in the mesh, as these nodes can now follow nonlinear trajectories and hence can align better with the motion of the structures of the solution to the PDE. Norms and notation are established in chapter 4, along with some preliminary results relating these norms to the PDE. Also in chapter 4, we present a detailed analysis of the approximation properties of the finite element space; this provides the error analysis for the finite element space as an approximating subspace of the true solution space of the differential equation. The results in section 4.2 are independent of the PDE under consideration.

In chapter 5, the content of chapters 2–4 are brought together in a proposed space-time moving finite element method. This method outlines a strategy for computing a finite element approximation to the solution of the PDE introduced in chaper 2 that employs the finite element space described and analyzed in chapters 3 and 4. This finite element method is the natural extension of those proposed in ([18],[65],[40]). The main result of this chapter is a symmetric error estimate for a certain class of moving finite element methods that is a generalization of the

error bounds found in the aforementioned trio of error analyses ([18],[65],[40]). An even broader extension of this method is described in chapter 6 that permits more general time integration schemes and, consequently, results in greater flexibility for using more stable time discretizations. This flexibility ultimately comes at the cost of sacrificing the symmetric error bound, where the error of the finite element solution must now be bounded by approximation errors introduced by the new time discretization scheme.

Chapter 7 is focused on some practical aspects of these moving finite element methods that are not necessarily addressed by the theoretical analysis provided in the preceding chapters. Namely, experiments are conducted to evaluate the benefits of using moving finite elements compared to static tensor product finite elements. Some modifications to the methods discussed in chapters 5 and 6 are explored, in addition to a scheme that incorporates adaptive meshing based on derivative recovery techniques. As a final experiment, we attempt to employ our moving finite element solver on Burger's equation, which is a simple nonlinear PDE that generates a sharp boundary layer in its solution, which sweeps through the spatial domain. This problem was also used in experiments by Miller and Miller in [57].

Chapter 1, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 2

The Differential Equation

This chapter is an introduction to the time-dependent linear convection-diffusion-reaction equation. This space-time equation contains a combination of hyperbolic and parabolic operators and is used to determine the behavior of physical systems that transport particles or quantities influenced by forces of diffusion, convection, and potentially some chemical reaction. The differential equation is given by

$$u_t - \nabla \cdot (a\nabla u) + b \cdot \nabla u + cu = f$$

and is a linear approximation of many complicated dynamical systems and is extensively used for computational experiments and numerical simulations for problems such as the heat transfer [7], wave propagation and fluid dynamics ([52],[44]), modeling meteorological phenomena [61], aerosol dynamics [41], and metal melting [27]. Due to the large variety of application, this equation and its steady state equation (when $u_t \equiv 0$) has been studied in the context of multiscale methods [49], discontinuous Galerkin and interior penalty methods [39], stabilized finite elements, the method of characteristics [48], upwind differencing [25], adaptive meshing, as well as moving finite elements ([18],[40],[65],[53]). There are several ways of formulating this problem and each formulation presented in this chapter is insightful or beneficial to our study in some way.

We begin by establishing some basic notation that is used throughout the remainder of this thesis. The spatial domain of the differential equation is denoted by Ω and is assumed to be a compact subset of $\mathbb{R}^d$, where $d = 1, 2$, or 3, and the boundary of Ω is represented by $\partial\Omega$. The time domain is a finite interval, $(0, T]$, and the space-time domain is therefore given by $\mathcal{F} \equiv \Omega \times (0, T]$. Applications of moving finite elements to problems with moving boundaries have been explored by Baines et al. [13] with some satisfying results; however, we restrict our attention to autonomous spatial domains. The space of square integrable functions on Ω is denoted by $\mathcal{L}_2(\Omega)$ (and likewise $\mathcal{L}_2(\partial\Omega)$ represents the space of functions that are square integrable on $\partial\Omega$). The space of functions in $\mathcal{L}_2(\Omega)$ that have up to k derivatives that are also square integrable is represented by $\mathcal{H}^k(\Omega)$.

2.1 The strong formulation of the linear convection-diffusion-reaction equation

The formulation given below is a physically intuitive description of the linear convection-diffusion-reaction equation and serves as a simple and accessible introduction to this initial value problem. Let u_0 be a given initial condition for the solution on $\mathcal{F}$ and let n denote the outward unit normal vector to the boundary $\partial\Omega$. This formulation is the *strong* formulation, as it has stronger regularity constraints for the solution than the other presented formulations.

Formulation 1. *Let a, b, c, and f be smooth and bounded functions defined on $\mathcal{F}$ such that there exist constants $\bar{a} > 0$ and $\bar{c} \geq 0$ with $a \geq \bar{a}$ and $c \geq \bar{c}$, and let g be piecewise continuous on $\partial\Omega$. Find u such that*

$$u_t - \nabla \cdot (a\nabla u) + b \cdot \nabla u + cu = f, \qquad \text{in } \mathcal{F}, \tag{2.1}$$

$$a\nabla u \cdot n = g, \qquad \text{on } \partial\Omega \times (0, T], \tag{2.2}$$

$$u(x, 0) = u_0(x), \qquad \text{for } x \text{ in } \Omega. \tag{2.3}$$

Formulation 1 has a direct physical interpretation. The function a determines the amount of *diffusion* acting upon the solution u, which is assumed to be nontrivial as the value of $a \geq \bar{a} > 0$. For many methods that apply to pure transport (convection) problems — cases when $a \equiv 0$ — a small amount of artificial diffusion is introduced to *stabilize* the numerical method, resulting in a singularly perturbed equation. If a numerical method is unstable, then the computed solution can potentially exhibit large oscillations that are uncharacteristic of the true solution to the differential equation. Essentially, added diffusion dampens numerically induced oscillatory behavior in the computed solution by "smearing" steep shock layers that may be present in the solution over larger areas so that the discretization scheme can detect these regions of rapid change properly ([51],[69]). Upon inspection, upwind differencing schemes and methods of characteristics also lead to an artificially enlarged diffusion term ([34],[48]), and this is also the underlying concept behind streamline diffusion methods. In the case of moving finite elements, a small amount of diffusion can prevent a phenomenon called "manifold fold over"

[28] or "element folding" [12] that leads to degenerate finite element spaces and, ultimately, singular linear systems that would otherwise describe the unique finite element solution. As such, only the case of a strictly positive diffusion coefficient is considered.

The *convection velocity*, b, is a term of significant interest in this problem; it can lead to difficulties in the analysis and serves as the primary motivation for using moving finite element methods. When the convection velocity is significantly greater than the diffusion quantity, the solution of the equation may develop steep shock layers that sweep through the spatial domain. These shock layers refer to a narrow region in the spatial domain where the solution, u, undergoes a dramatic change in value and these regions of rapid change generally propagate through the spatial domain. In the limiting case where $a \equiv 0$, this shock layer can be discontinuous ([64],[28]), leading to an ill conditioned problem and a small amount of artificial diffusion is introduced, as discussed above. In general, accurately capturing the fine structures near these shock layers can be computationally expensive [57], often requiring very short time steps. We shall see that aligning the mesh motion with the convection velocity may permit significantly larger time steps in solving the differential equation numerically, while still preserving accuracy of the solution in the shock layer. This strategy of following the convection velocity has been studied in previous research ([18], [40],[65]) and is the motivation for upwind differencing methods [35], the method of characteristics [48], flux splitting methods, and the Lagrangian approach for solving convection dominated problems [30].

The function c is the reaction term and describes how the value of the solution influences its own evolution in time. The assumption that c is nonnegative could be relaxed to where c is only required to bounded; however, this leads to unstable growth of the solution and is, therefore, not considered. The term f on the right hand side in (2.1) is the *source* function, which describes external forces that may be acting upon the solution. In the case of the heat equation, this could be some heat source that is independent of the temperature, represented by u. The *boundary condition* is determined by g in equation (2.2). In formulation

1, the function g describes a Neumann (or natural) boundary condition, as the gradient of the solution, ∇u, is constrained. If the equation (2.2) were replaced by the boundary condition

$$u = g_D \quad \text{on } \partial\Omega \times (0, T], \tag{2.4}$$

we would have a Dirichlet (or essential) boundary condition. Potentially, a Neumann boundary condition could be imposed on a subset of the boundary and a Dirichlet boundary could be imposed on the rest, giving a mixed boundary condition. For convenience, a Neumann boundary condition is assumed as the boundary condition does not play a significant role in this error analysis. Our results still hold with only minor changes to their arguments when other boundary conditions are imposed.

An unfortunate aspect of formulation 1 is that the solution u is required to be twice differentiable with respect to the spatial variables. By using techniques of variational calculus, this regularity assumption can be relaxed and the differential equation can be posed in the *variational* (or *weak*) form.

2.2 A variational formulation

In order to relax the regularity constraint that u is twice differentiable in space, the differential equations are imposed weakly in space but strongly in time. To do this, the equations in formulation 1 are multiplied by a test function, integrated over the spatial domain Ω, and the divergence theorem is applied to the diffusion term. Using the boundary condition (2.2), the differential equation (2.1) becomes

$$\int_\Omega u_t(x,t)\chi(x,t) + a(x,t)\nabla u(x,t)\cdot\nabla\chi(x) + b(x,t)\cdot\nabla u(x,t)\chi(x) + c(x,t)u(x,t)\chi(x,t)\ dx = \int_\Omega f(x,t)\chi(x)\ dx + \int_{\partial\Omega} g(s)\chi(s)\ ds, \tag{2.5}$$

for $0 < t \leq T$, where χ is an arbitrary test function living in $\mathcal{H}^1(\Omega)$. This weakly imposes equations (2.1) and (2.2) on Ω.

Define $\mathcal{V}$ to be the set of all functions, u, with domain $\mathcal{F}$ such that $u(t) \in \mathcal{H}^1(\Omega)$ and $u_t(t) \in \mathcal{L}_2(\Omega)$ for $0 \leq t \leq T$. This function space serves as the trial

space for the variational form of the differential equation and $\mathcal{H}^1(\Omega)$ shall be the test space. Since equation (2.5) is imposed at all times t in the time domain and $\mathcal{V}\big|_t = \mathcal{H}^1(\Omega)$, we see that the test space is the same as the trial space restricted to a time slice. The constraint that $u_t(t) \in \mathcal{L}_2(\Omega)$ for all t ensures that the equation (2.5) is well-posed and "ties" together this equation throughout the time-domain.

As the differential equation is imposed weakly in space and strongly in time, it is natural to analyze the differential operators of the spatial variables independently from the time derivative. Let v and χ live in $\mathcal{H}^1(\Omega)$ and define the time dependent bilinear functional

$$\mathcal{A}_0(v, \chi; t) \equiv \int_\Omega a(x,t)\nabla v(x,t)\cdot\nabla\chi(x) + b(x,t)\cdot\nabla v(x,t)\ \chi(x) + c(x,t)v(x,t)\chi(x)\ dx. \tag{2.6}$$

This bilinear functional is simply notation for the spatial terms on the left side of (2.5). When the convection velocity b is identically zero, the bilinear form $\mathcal{A}_0$ is symmetric, positive semi-definite, bounded, and coercive, which will be useful in proving the existence and uniqueness of the solution for the discrete variational formulation. On the other hand, when b is large relative to the other coefficient functions in (2.6), this no longer hold and will result in requiring smaller time steps in the discrete formulations. The method of characteristics is based on reducing the asymmetry introduced by the convection velocity via a "re-alignment" of the time derivative with the characteristics of the hyperbolic differential operator, $u_t + b\cdot\nabla u$. A similar approach can be taken with moving meshes, without some of the difficulties that arise from the misalignment of the characteristic trajectories and the spatial mesh at the distinct time steps. The usual inner-products on Ω and $\partial\Omega$ are given by

$$(f, \chi) = \int_\Omega f(x)\chi(x)\ dx \qquad \text{and} \qquad \langle g, \chi\rangle = \int_{\partial\Omega} g(s)\chi(s)\ ds.$$

The variational formulation is formally given as follows.

Formulation 2. *Let a, b, c, and f be smooth and bounded functions on $\mathcal{F}$ satisfying $a \geq \bar{a} > 0$ and $c \geq \bar{c} \geq 0$, and let $g(t)$ be integrable on the $\partial\Omega$ for $0 < t \leq T$. Find u in $\mathcal{V}$ such that for all χ in $\mathcal{H}^1(\Omega)$ and $0 < t \leq T$,*

$$\big(u_t(\cdot,t), \chi\big) + \mathcal{A}_0\big(u, \chi; t\big) = \big(f(\cdot,t), \chi\big) + \big\langle g(\cdot,t), \chi\big\rangle, \tag{2.7}$$

and when $t = 0$

$$\big(u(\cdot, 0), \chi\big) = \big(u_0, \chi\big).$$

In this formulation, it has been noted that the differential equations are imposed weakly in space, but for all times in the time domain. Another variational formulation might take the test functions to be defined on the space-time domain, $\mathcal{F}$, and then integrate over space and time. Some versions of this type of method have been studied in previous research ([65], [18]), though no symmetric error estimates have been found as far as we know. In such methods, the time domain is partitioned and the differential equation is solved on each time partition sequentially, which is an intermediate approach between the method of lines and a space-time variational formulation; penalty terms are used to enforce continuity between the time partitions as the time basis functions are restricted to their respective time partitions. Ultimately, this leads to the loss of a symmetric error estimate due to the presence of penalty terms. Formulation 2, however, is compatible with a *method of lines* approach, which has been traditionally employed for moving mesh schemes. As noted before, the method of lines approach does not correspond a variational formulation in space-time and consequently may lead to non-symmetric error estimates for some choices of time-stepping schemes, as we show in chapters 5 and 6. In section 5.2, there is a discussion comparing formulations with the equations imposed weakly and strongly in time.

2.2.1 A variational formulation with non-autonomous domain parametrization

Consider the time dependent bilinear form given in (2.6). Due to the convection term b, the bilinear form, $\mathcal{A}_0$, is not positive semi-definite, symmetric, or coercive, which presents difficulty in proving estimates for stability and consistency. Furthermore, convection dominated problems tend to have high computational cost to solve accurately ([57], [58]), as small times steps are needed to track the motion of refined structures in the solution, such as a steep moving front. Moving meshes are specifically designed to alleviate these issues in a manner similar to that to that

of the method of characteristics; the prominent distinction between moving finite elements and the methods of characteristics is that the methods of characteristics do not typically have the nodes of the spatial mesh follow the characteristic trajectories along which the time derivative is re-aligned. In this section, we describe a variational formulation with a non-autonomous parametrization of the spatial domain and study its effects on formulation 2.

Let $x : \mathcal{F} \to \Omega$ be a continuous map from the space-time domain to the spatial domain such that $x(\Omega, t) = \Omega$ for all t in $(0, T]$. By varying t, the maps $\{x(\cdot, t)\}_t$ form a set of time-dependent parametrizations of Ω. From this perspective, $x(\cdot, t)$ is a map from Ω to some manifold living on Ω. This framework has been used to analyze some of the singularities that may arise from this manifold folding over, which corresponds to a non-injective parametrization of the spatial domain. These singularities are discussed in ([28],[29], [12], [57], [57], [65], [18]), and later in this paper.

Following a Lagrangian approach — as well as the method of characteristics approach — the map x can represent a family of particle trajectories. To define these particle trajectories, fix a base point, (y, s), in the domain, $\mathcal{F}$. The trajectory passing through (y, s) is denoted by $x(y, s; t)$, where t is the parameter that is varied to trace out the path of x. It is useful to think of these trajectories as the flow lines of some ordinary differential equation: let $(y, s) \in \mathcal{F}$ and define $x(t)$ by the ordinary differential equation,

$$\frac{d}{dt}x(y, s; t) = \tau(x(y, s; t), t), \quad \text{and} \quad x(y, s; s) = y, \tag{2.8}$$

where $\tau : \mathcal{F} \to I\!R^{d+1}$ is some integrable vector-valued function on $\mathcal{F}$, though not necessarily continuous. In order to keep $x(y, s; t)$ in Ω for all t, the vector function $\tau(\cdot, t)$ must be satisfy the boundary condition

$$\tau(x, t) \cdot n = 0 \tag{2.9}$$

for all x on $\partial\Omega$ and t in $(0, T]$, where n is the outward normal on the boundary of the domain. This boundary condition ensures that the flow lines $x(y, s; t)$ stay inside the spatial domain for all times t. Given a base point (y, s) in $\mathcal{F}$, these flow lines map the line segment $(0, T]$ to a time dependent trajectory in $\mathcal{F}$. For

simplicity, this trajectory is assumed to be a *local parametrization*, meaning that the base point of $x(t)$ is assumed to be at the point (x,t) in $\mathcal{F}$, unless otherwise explicitly noted. For shorthand, the dependence on the base point is suppressed and we denote $x(t) = x(y,s;t)\big|_{\substack{y=x\\s=t}}$ and $x_t(t) = \frac{d}{dt}x(y,s;t)\big|_{\substack{y=x\\s=t}} = \tau(x,t)$. (The notation x_t and τ are used interchangeably, as they represent the same quantity.) Assuming that the convection velocity b is orthogonal to the outward normal on the boundary of the domain, taking $\tau = b$ in (2.8) recovers the particle trajectories chosen by the method of characteristics and can be an effective choice for mesh parametrization, as we show in section 4.1.2 and chapter 5. Due to this connection with the method of characteristics, the trajectories, $x(t)$, are referred to as the *characteristic trajectories*. Note that the choice $\tau \equiv 0$ implies a static parametrization and straight-line characteristic trajectories.

Let $u \in \mathcal{V}$. Using the chain rule, the derivative along the characteristic trajectories is given by

$$\frac{d}{dt}\Big[u\big(x(t),t\big)\Big] = x_t(t)\cdot\nabla u(x(t),t) + u_t(x(t),t).$$

Let ∂_τ represent the differential operator given above; that is,

$$\partial_\tau u(x,t) \equiv x_t(t)\cdot\nabla u(x,t) + u_t(x,t).$$

This differential operator is called the *characteristic derivative* and allows the differential equation (2.1) to be rewritten as

$$\partial_\tau u - \nabla\cdot(a\nabla u) + (b - x_t)\cdot\nabla u + cu = f. \tag{2.10}$$

Intuitively, this formulation flows the points in the spatial domain along the characteristic trajectories, allowing the convection velocity term to be absorbed into this flow. The motion of the spatial points is described by the time derivative of this parametrization, x_t. As mentioned before, choosing $x_t = b$ corresponds to an "ideal" method of characteristics, fully canceling the convection velocity. Fully discrete formulations only permit piecewise polynomial flow lines and, consequently, only an approximation $x_t \approx b$ can be computed for general b, to *approximately* align the characteristic trajectories with the flow of the convection term. On the

other hand, if $x_t \equiv 0$, the spatial parametrization is autonomous and the variational formulation given in formulation 2 is recovered — note that when $x_t \equiv 0$, the characteristic derivative satisfies $\partial_\tau u = u_t$.

The spatial bilinear form associated with equation (2.10) is given by

$$\mathcal{A}_\tau(u,\chi;t) \equiv \int_\Omega a(x,t)\nabla u(x,t)\cdot\nabla\chi(x) + (b(x,t)-x_t(t))\cdot\nabla u(x,t)\ \chi(x) + c(x,t)u(x,t)\chi(x)\ dx,$$

where the subscript τ is used to indicate the mesh motion. As one can see, choosing $x_t \approx b$ (or $x_t = b$) will reduce (or nullify) the effect of the convection term in $\mathcal{A}_\tau$, leading to a more symmetric functional in the sense that

$$|\mathcal{A}_\tau(u,\chi;t) - \mathcal{A}_\tau(\chi,u;t)| \le \|b(t)-x_t(t)\|_\infty \|u\|_1 \|\chi\|_1,$$

where $\|\cdot\|_1$ is taken to be the usual norm for $\mathcal{H}^1(\Omega)$ and $\|\cdot\|_\infty$ is the essential supremum over Ω.

Using the time-dependent parametrization for Ω, the variational *characteristic formulation* is given.

Formulation 3. *Let a, b, c, and f be smooth and bounded functions on $\mathcal{F}$ satisfying $a \ge \bar{a} > 0$ and $c \ge \bar{c} \ge 0$, let $g(t)$ be in $\mathcal{L}_2(\partial\Omega)$, and let $\tau(t) : \mathcal{F} \to \mathbb{R}^d$ live in $\mathcal{L}_2(\Omega)$ and be orthogonal to the outward normal vector on the domain boundary for $0 < t \le T$. Find u in $\mathcal{V}$ such that for all χ in $\mathcal{H}^1(\Omega)$ and $0 < t \le T$,*

$$\big(\partial_\tau u(t),\chi\big) + \mathcal{A}_\tau\big(u,\chi;t\big) = \big(f(t),\chi\big) + \big\langle g(t),\chi\big\rangle, \tag{2.11}$$

and when $t = 0$

$$\big(u(\cdot,0),\chi\big) = \big(u_0,\chi\big).$$

Note that the vector field that determines the spatial motion, τ in (2.8), is only required to be integrable and orthogonal to the outward normal of the domain boundary, as in (2.9), so that the mesh motion does not "flow out" of the domain. This means that there are many possible choices for how to evolve the spatial parametrization. Formulation 3 does not restrict the choice of spatial

parametrization beyond these regularity assumptions; as such, this formulation encompasses many possibilities for the choice of characteristics. Upon discretizing the spatial domain, the characteristic trajectories, x_t, determine the mesh motion, which justifies the ambiguity of the characteristic function in formulation 3. This formulation permits general characteristic functions so that the error analysis of its discretized formulation is general enough to apply to many mesh motion strategies. Often, the mesh motion is determined by imposing additional differential equations that determine the mesh motion according to some conservation law or physical phenomenon ([13],[14]), or residual minimization ([12],[18],[65]). These approaches for mesh selection conform to a discrete version of the problem formulation 3, in addition to other constraints that may be used to determine the spatial parametrization. Some mesh motion schemes require the use of penalty functions or stability terms to ensure shape regularity and desired behavior of the moving mesh ([57],[57], [56]); some of these penalty functions can be "hidden" in the source terms for our purposes. In such a case, the source function in equation (2.11) may depend on the mesh motion as well, $f(x,t) = f(x, x_t, t)$. Some schemes for mesh moving inherently introduce nonlinear terms into the differential equation ([28],[29],[50]), which means that our analysis does not directly apply to these methods, though one of our numerical experiments in chapter 7 show satisfactory performance on a simple nonlinear equation. Shape regularity will be discussed in depth in section 3.3, but mesh selection will be assumed to satisfy the constraints of formulation 3 a priori without modification.

Formulation 3 follows the semi-variational form of formulation 2 as it imposes the differential equation weakly in space, though strongly in time. The key difference between formulations 2 and 3 is that the new formulation allows the convection term to be absorbed into the characteristic derivative. The result is a formulation that can effectively improve the conditioning of convection dominated problems. Discretized versions of formulation 3 are given in chapters 5 and 6, where sufficient conditions for the well-posedness of these discretized problems are provided.

Chapter 2, in part, is in preparation for submission for publication of this

material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 3

A Space-Time Tensor Finite Element Space

Finite element methods fall under the category of Galerkin methods, where the solution space, $\mathcal{V}$, is approximated by some finite dimensional subspace. This finite dimensional subspace is the mechanism that discretizes formulations 2 and 3, and consequently plays a large role in determining properties of the finite element method used to define the finite element solution to a differential equation. The description of a finite element space is essentially an explanation of where degrees of freedom are allocated in the discretized problem. In many cases, degrees of freedom are used to determine the mesh on which the finite element basis functions are defined and to define the basis functions of the finite element space on the reference element(s).

For elliptic equations, there is a vast amount of research describing and analyzing finite element spaces used to discretize differential equations. For problems of a single dimension, the finite element mesh typically consists of a partition of the domain; for two-dimensional problems, the finite element mesh is often a triangulation of the domain; and for three-dimensional problems, a basic mesh could be a union of tetrahedral elements. Another common meshing strategy for problems with more than a single dimension employs tensor-product elements, where the elements in this mesh are rectangles or rectangular prisms. A benefit of tensor-product finite elements is that each dimension of the domain is discretized independently and the cross-terms of the tensor-product provide additional degrees of freedom that allow for more flexible finite element spaces [7]. Other types of elements that are commonly used are serendipity elements, which are tensor-product elements without degrees of freedom on the interior of the mesh elements to improve the computation efficiency of the finite element method while maintaining inter-element continuity of the finite element functions [6]. Other meshing schemes involve more general polygon elements than triangles and boxes, as in ([42],[70]).

In addition to describing the mesh and its associated elements, the finite element space cannot be fully constructed without understanding the basis functions. In most applications, piecewise polynomial basis functions are defined over the mesh to generate the finite element space, though general polygon elements lead to examples of finite element spaces that have non-polynomial finite element

basis functions [70]. Variations in the different types of the basis functions often correspond to the inter-element continuity of the finite element functions — discontinuous Galerkin methods, for example, do not require finite element functions to be continuous [8] and benefit from the resulting efficiency of adaptive meshing (these methods are commonly applied to parabolic and hyperbolic equations too, as in [63] and [39]), whereas some discretization schemes involve a finite element space where derivatives of the finite element functions are required to be continuous up to some order [26]. Other types of basis functions differ in the degree of the polynomial basis functions used to generate the mesh ([67],[11]). Typically, these spaces are constructed by defining basis functions on a reference element and using isoparametric maps to send these basis functions onto the degrees of freedom in finite element mesh. Other alternatives to constructing finite element spaces in this way exist: for example, the partition of unity finite element method [10], element-free Galerkin methods [24], and other methods ([71],[46]) are *meshless*, meaning that only information regarding the location of the nodes in the domain is required. These methods permit more general finite element spaces and are well-suited to adaptive methods; however, the use of isoparametric maps can simplify the construction of the finite element basis functions and the linear system corresponding to the discrete formulation of the partial differential equation.

So far, the finite element spaces discussed pertain only to elliptic equations, which means that these discretizations do not consider the dimension of time. The topic of this dissertation is a class of finite element methods for space-time differential equations and we accordingly must consider meshes for space-time domains. The most straightforward meshes for space-time domains can be attained by treating the time variable as an additional spatial variable, leading to discrete problems where the approximate solution is computed over the entire space-time domain at once. This permits solvers and software designed for elliptic partial differential equations, like the software package PLTMG [16], to compute solutions for time-dependent problems without special treatment of the time variable. One difficulty with this approach is that adaptive time-stepping methods cannot be employed, as it becomes a complicated task to adjust the resolution of the time discretization

without affecting the spatial discretization. More specific to our purposes, this approach is not readily compatible with formulations 2 and 3, as it imposes the differential equation *weakly* in time. Since equation (2.11) in formulaton 3 (and equation (2.7) for formulaton 2, respectively) is required to hold for all t, the time discretization must be independent of the spatial discretization. Thus, a tensor-product finite element space is used, where the discretization is a tensor-product of spatial discretizations and a time discretization. As discussed in chapter 5, this space-time tensor finite element space is easily adapted to fit a method of lines approach to solving the discretized problem. Many finite element methods employing a method of lines approach can be interpreted in terms of this sort of finite element space; the description of the linear case of this finite element space is outlined in ([18],[65]) for space-time moving finite elements.

In this chapter, we follow ([18],[65]) in describing a tensor-product finite element space that is built on the space-time domain, $\mathcal{F}$, where the finite element space in this dissertation permits higher order basis functions and more flexible mesh motion. Furthermore, we discuss the construction of the finite element space via the isoparametric and its potential degeneracies. Discussion regarding the approximation properties of such finite element spaces are provided in the next chapter; discussion of the proposed methods for solving the discrete equation are given in chapters 5 and 6. Discussion for finding appropriate finite element spaces for a given problem are presented in chapter 7, along with the numerical experiments.

3.1 The description of a moving finite element mesh

We begin by describing a moving mesh that is built on the space-time domain. As mentioned above, this mesh is the tensor-product of spatial meshes with a time partition. The result is an isoparametric finite element space comprised of continuous piecewise polynomials in space tensored with piecewise polynomials in time on the reference element. This provides a simple means to allow for mesh motion, as the time discretization will be chosen to be independent of the spatial

discretization. Another important feature of the finite element functions is that they are permitted to have discontinuities at discrete time nodes, which was first introduced to moving finite element methods in [50].

This section is concerned with the description of the finite element mesh; the isoparametric maps and finite element functions are described in the next section. The finite element space is denoted by $\mathcal{V}_h^p$, where the superscript p describes the order of the tensored piecewise polynomial finite element functions and the subscript h denotes the relative size of the maximum space-time diameter of an element in the discretization to the size of $\mathcal{F}$.

3.1.1 The time and space discretizations

To construct the finite element space $\mathcal{V}_h^p$, it is helpful to first describe the associated mesh, which is denoted by $\mathcal{T}_h^p$. Partition the time domain into m disjoint intervals, where the endpoints of the partitions are given by $\{t_i\}$ and satisfy

$$0 = t_0 < t_1 < \ldots < t_m = T,$$

with $\Delta t_i \equiv t_i - t_{i-1}$ for $i = 1, \ldots, m$. Throughout this document, the index i shall be assumed to run over the time partitions, $1 \leq i \leq m$, unless otherwise noted. This organizes the space-time domain $\mathcal{F}$ into space-time partitions, denoted by $\mathcal{F}_i \equiv \Omega \times (t_{i-1}, t_i]$, and the corresponding finite element space on $\mathcal{F}_i$ is $\mathcal{V}_{h,i}^p$, with mesh $\mathcal{T}_{h,i}^p$. Within each of these partitions, the finite element functions will be required to be continuous; however, continuity is *not* assumed between partitions. This allows for discontinuous changes in the mesh at the discrete nodes of the time partitions. This idea was originally investigated by Kuprat [50] and referred to as *graph massaging*. These discontinuous changes in the mesh have been implemented and are proven to be a very efficient tool for maintaining shape regular meshes ([28],[29],[18]).

What is special about the methods studied in this thesis is that the nodes of the finite element mesh are permitted continuously shift in time throughout the time partitions. Such methods have been analyzed in ([18], [65], [40]), though attention was restricted to the case of linear finite element spaces in these papers.

Now, this is generalized to higher-order polynomial mesh motion. To represent this mathematically, fix t in $(t_{i-1}, t_i]$ and choose some triangulation of the spatial domain Ω with nodes at $\{x_k^{(i)}(t)\}$, with $1 \leq k \leq N_i$. Since the finite element space is the tensor product of degree p polynomials, each node $x_k^{(i)}(t)$ is allowed to be a polynomial of at most degree p, mapping from $(t_{i-1}, t_i]$ to Ω. By tracking the time variable $(x_k^{(i)}(t), t) : (t_{i-1}, t_i] \to \mathcal{F}_i$, the nodes $x_k^{(i)}(t)$ trace out the characteristic trajectories of the finite element mesh within the time partition. At any time t in the partition $(t_{i-1}, t_i]$, the spatial nodes $x_k^{(i)}(t)$ are required to define a triangulation of the spatial domain and this triangulation is denoted by $\mathcal{T}_h^p(t)$.

3.1.2 Example in one spatial dimension (d=1)

To visualize a partition of $\mathcal{T}_h^p$, consider the case of $d = 1$ and $p = 2$. This corresponds to a problem with one spatial dimension and choosing a finite element space that is the tensor product of piecewise quadratic polynomials. Let $\Omega = [0, 1]$ and fix t in $(t_{i-1}, t_i]$. Let $\mathcal{T}_h^2(t)$ have nodes $\{x_k^{(i)}(t)\}$ such that

$$0 \equiv x_0^{(i)}(t) < x_1^{(i)}(t) < \ldots < x_{N_i}^{(i)}(t) \equiv 1$$

for $t_{i-1} < t \leq t_i$. Denote $\Delta x_k^{(i)}(t) \equiv x_k^{(i)}(t) - x_{k-1}^{(i)}(t)$ and suppose that $0 < \Delta x_{\min} \leq \Delta x_k^{(i)}(t) \leq \Delta x$ for $k = 1, \ldots, N_i$. The resulting mesh is a tessellation of $\mathcal{F}_i$ with quadratically moving space nodes, and figure 3.1 depicts an example of a time partition. In this figure, the solid and dashed horizontal lines correspond to the spatial discretizations at a particular *time slice*, where a time slice refers to the spatial domain at some fixed point in time. Other time partitions are discretized analogously and the complete mesh on $\mathcal{F}$ would look like meshes similar to figure 3.1 stacked upon one another, potentially with node discontinuities between the time partitions.

3.1.3 Mesh discontinuities

At this point, the mesh associated with a partition of the finite element space, $\mathcal{V}_{h,i}^p$, has been discussed in detail. Now, consider what happens between the time partitions. Recall that there is no continuity requirement of $\mathcal{T}_h^p(t)$ at

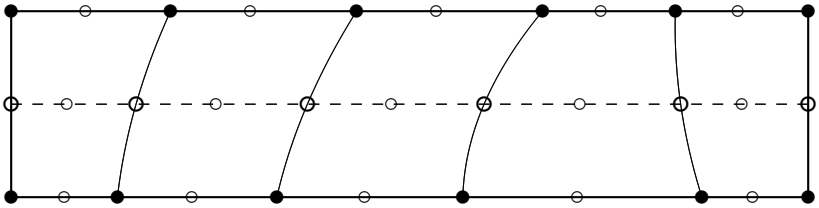

Figure 3.1: An example space-time mesh $\mathcal{T}_{h,i}^p$ on a partition with $d = 1$ and $p = 2$. The filled circles represent the space-time "hat" basis functions; hollow circles correspond to basis functions that are the product of a "bump" basis function.

the discrete time nodes. These discontinuities in the mesh provide for the periodic addition, deletion, and relocation of the spatial nodes at the beginning of each time partition [50]. Furthermore, the motion of the space nodes can lead to elements that tangle or degenerate in time and, thus, these periodic reconfigurations of the mesh allow for more robust finite element spaces ([50],[28],[29]). This can occur when a set of spatial nodes converge or cross each other — even if the nodes move too close together, the shape regularity assumptions may be violated. Discontinuous mesh adaptation serves as a means of avoiding such degeneration of elements. A discussion on the tradeoff between moving the mesh to offset the convection term and constraining the motion to maintain shape regularity is given in section 3.3, where these reconfigurations of the mesh are a rather invaluable tool.

3.1.4 The reference elements

The mesh $\mathcal{T}_h^p$ is constructed by a family of isoparametric maps from a reference element to the domain $\mathcal{F}$. We now describe the reference elements for cases $d = 1, 2, 3$. The most striking feature of the reference element stems from the fact that $\mathcal{V}_h^p$ is the tensor product of finite element spaces on Ω and a finite element space on $(0, T]$. As a result, the space-time reference element is the Cartesian product of the spatial reference element and the time reference element. Since the shape of the spatial reference element depends on the dimension, d, of the spatial domain Ω, the cases $d = 1, 2$, and 3 are considered separately.

For $d = 1$, the elements in $\mathcal{T}_h^p$ are curvilinear trapezoids with flat parallel edges corresponding to the beginning and end of the time partitions, and curved edges representing the moving space nodes — recall that these edges are polynomial curves of degree p. The associated space-time reference element is the Cartesian product of the reference element in space with the reference element in time: $e_{\text{ref}} = [0,1] \times [0,1]$. Moreover, the space-time basis functions are given by the tensor products of the degree p polynomial spatial basis functions on $[0,1]$ and the degree p polynomial temporal basis functions on $[0,1]$. This implies that there are $(p+1)^2$ degrees of freedom associated with the reference element. Furthermore, since a tensor product is used to define the space-time basis functions, the degrees of freedom on the reference element are aligned into time slices that are orthogonal to the time direction (the degrees of freedom are represented by the filled and empty circles in figure 3.1 and their alignment into time slices is emphasized by the dashed line).

When $d = 2$, the elements are curvilinear triangular prisms. The flat triangular parallel edges mark the beginning and end of a time partition, and the curved edges correspond to the space nodes moving along polynomial trajectories of degree p. The reference element for the spatial discretization is the unit triangle with $(p+1)(p+2)/2$ degrees of freedom and the reference element in time is still the unit interval $[0,1]$ with $p+1$ degrees of freedom; therefore, the space-time reference element is a wedge-like shape, which is defined by the Cartesian product of the spatial and temporal reference elements,

$$e_{\text{ref}} = \{(x_1, x_2) \in \mathbb{R}^2 \,|\, x_1, x_2 \geq 0,\ x_1 + x_2 \leq 1\} \times [0,1],$$

and has $(p+1)\times(p+1)(p+2)/2$ degrees of freedom. The space-time basis functions are the tensor product of the degree p polynomial basis functions in space, on the unit triangle, and the degree p polynomial basis functions in time on the unit interval. As in the case when $d = 1$, the degrees of freedom are organized into time slices on the reference element. Figure 3.2 illustrates how e_{ref} looks for linear elements, $p = 1$, with two spatial dimensions, $d = 2$.

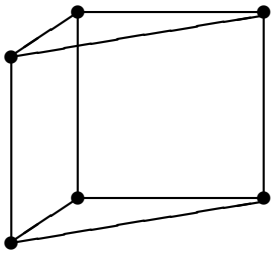

Figure 3.2: An illustration of e_{ref} with degrees of freedom shown when $d = 2$ and $p = 1$.

The final case is when $d = 3$ and the reference element is four-dimensional. Accordingly, we enjoy the opportunity to creatively describe its shape. A suitable description is that the elements in this space are curvilinear tetrahedral hyper-prisms, which one could imagine as a tetrahedron that morphs continuously in time. Remember that the nodes' movements in time are described by polynomial paths of degree p. For linear elements, the evolution of an element would would appear as a tetrahedron steadily morphing into a new tetrahedron, whereas higher-order elements are allowed to have several intermediate phases. The space-time reference element is the Cartesian product of the unit tetrahedron — the spatial reference element — with the unit interval reference element for time. In set notation, the reference element is represented by $e_{\text{ref}} = \{(x_1, x_2, x_3) \in \mathbb{R}^3 \,|\, x_1, x_2, x_3 \geq 0,\ x_1+x_2+x_3 \leq 1\} \times [0, 1]$. The spatial reference element has $(p+1)(p+2)(p+3)/6$ degrees of freedom and the time reference element has $p + 1$ degrees of freedom; consequently, the space-time element e_{ref} has $(p+1) \times (p+1)(p+2)(p+3)/6$ degrees of freedom. The space-time basis functions are the tensor product of the degree p polynomial basis functions in space and time, just as in the previous cases. In the case when $d = 3$, it remains true that the nodes are organized into time slices, although this property is not as obvious in four dimensions. To recover this idea, project one spatial dimension onto the other two and observe that this transforms the unit tetrahedron into the unit triangle, which is the spatial reference element when $d = 2$, where the nodes are organized into time slices. Repeating this process for each spatial dimension shows that the degrees of freedom occur in time slices as in the previous cases. This property is also be reflected in the discussion of the

isoparametric maps from e_{ref} to an element in the mesh.

3.2 The isoparametric map

An isoparametric map in the context of this dissertation is a map from the reference element, e_{ref}, to an element e in the mesh. Analyzing the isoparametric maps is a crucial component to understanding finite element spaces built on moving meshes since isoparametric maps are used both in the construction of the finite element functions and the mesh $\mathcal{T}_h^p$. They contain information regarding properties of the finite element functions, the shape regularity of the mesh, and, more specific to the purposes of this paper, the motion of the mesh in time. The term 'isoparametric' is a reference to the fact that the map is constructed by mapping the degrees of freedom in e_{ref} to the degrees of freedom of e in $\mathcal{T}_h^p$.

Isoparametric maps are tools that we use to parameterize the space-time variables on a given element in the mesh by a set of variables on a reference element, which is common to all of the elements in the mesh. This is convenient for defining and performing calculations involving finite element functions in an element-wise fashion, which is ultimately the approach taken to solve for the finite element solution of the partial differential equation. To start, an example isoparametric map is constructed in the case of a one-dimensional spatial domain, $d = 1$, and the extension to higher dimensional cases follows.

3.2.1 Example in one spatial dimension (d=1)

To define functions in the finite element space $\mathcal{V}_h^p$, the degrees of freedom and the associated basis functions on the reference element must be discussed. Since the finite element space is a tensor product of finite element spaces on the spatial domain and the time domain, the degrees of freedom in space and time can be discussed independent of one another. Let $e_{\text{ref}} = s_{\text{ref}} \times [0, 1]$ denote the reference element as the Cartesian product of the spatial reference element, s_{ref}, and the temporal reference element, $[0, 1]$ and let element $e \in \mathcal{T}_{h,i}^p$ for some fixed i.

Let $\{\hat{t}_j\}$ for $j = 0, 1, \ldots, p$ denote the degrees of freedom on the time

reference element such that

$$0 = \hat{t}_0 < \hat{t}_1 < \ldots < \hat{t}_p \leq 1.$$

For the spatial degrees of freedom, analogous notation is used: the degrees of freedom in s_{ref} are typically chosen to be uniformly distributed as $\hat{x}_k = k/p$, for $k = 0, 1, \ldots, p$. As a result, the degrees of freedom are uniformly distributed in parallel time slices on the reference element.

To parameterize the variables (x, t) on element e in the i^{th} mesh partition, $\mathcal{T}_{h,i}^p$, the isoparametric maps are used. Let $\left\{(x_k(t_{i,j}), t_{i,j})\right\}$ represent the degrees of freedom on an element e that have the associated isoparametric maps $\{\iota^{(j,k)}\}$, with $0 \leq j, k \leq p$. Note that the degrees of freedom on the element e must be given in parallel time slices, as on the reference element. This ensures that the finite element space for fixed t is a standard finite element space on Ω. The parameterization using the isoparametric maps then takes the form

$$\begin{bmatrix} x \\ t \end{bmatrix} = \sum_{j,k=0}^{p} \iota^{(j,k)}(\hat{x}, \hat{t}) \begin{bmatrix} x_k(t_{i,j}) \\ t_{i,j} \end{bmatrix}, \tag{3.1}$$

which parameterizes the (x, t)-space in the mesh by the $(\hat{x}, \hat{t})$-space on the reference element using the isoparametric maps as basis functions.

We now construct the isoparametric maps. Let $\hat{\beta}_j(\hat{t})$ be the temporal basis function associated with the j^{th} degree of freedom on the time reference element [0,1] and $\hat{\sigma}_k(\hat{x})$ be the spatial basis function associated with the k^{th} degree of freedom on s_{ref} such that

$$\hat{\beta}_j(\hat{t}) = \begin{cases} \prod_{\substack{\ell=1 \\ \ell \neq j}}^{p} \frac{\hat{t} - \hat{t}_\ell}{\hat{t}_j - \hat{t}_\ell} & \text{when } \hat{t} \in [0, 1], \\ 0 & \text{when } \hat{t} \notin [0, 1], \end{cases}$$

and

$$\hat{\sigma}_k(\hat{x}) = \begin{cases} \prod_{\substack{j=1 \\ j \neq k}}^{p} \frac{\hat{x} - \hat{x}_j}{\hat{x}_k - \hat{x}_j} & \text{when } \hat{x} \in s_{\text{ref}}, \\ 0 & \text{when } \hat{x} \notin s_{\text{ref}}. \end{cases}$$

As the finite element space $\mathcal{V}_h^p$ is a tensor product of spatial and temporal finite element spaces, the isoparametric basis functions on the reference element are given

by

$$\iota^{(j,k)}(\hat{x},\hat{t}) = \hat{\beta}_j(\hat{t})\hat{\sigma}_k(\hat{x}),$$

for $j,k = 0,\ldots,p$. Accordingly, the polynomial $\hat{\beta}_j(\hat{t})\hat{\sigma}_k(\hat{x})$ is of degree p in $\hat{t}$ and in $\hat{x}$, and takes a value of one at $(\hat{x}_k,\hat{t}_j)$ and is zero at the other degrees of freedom. If $p = 1$ and $d = 1$, these are just the usual bilinear element basis functions that are used on rectangular grids for two-dimensional elliptic problems.

Writing the parameterization (3.1) in terms of the basis functions on the reference element, we have

$$\begin{aligned}\begin{bmatrix} x \\ t \end{bmatrix} &= \sum_{j,k=0}^{p} \hat{\beta}_j(\hat{t})\hat{\sigma}_k(\hat{x}) \begin{bmatrix} x_k(t_{i,j}) \\ t_{i,j} \end{bmatrix} \\ &= \sum_{j=0}^{p} \hat{\beta}_j(\hat{t}) \begin{bmatrix} (1-\hat{x})x_0(t_{i,j}) + \hat{x}\, x_p(t_{i,j}) \\ t_{i,j} \end{bmatrix},\end{aligned}$$

where we used $\sum_{k=0}^{p} \hat{\sigma}_k(\hat{x}_\ell)x_k(t_{i,j}) = x_\ell(t_{i,j})$ for the spatial component and $\sum_{k=0}^{p} \hat{\sigma}_k(\hat{x}) = 1$ for the time component. We define the polynomial trajectories of the spatial nodes by the Lagrangian time basis functions

$$x_k(\hat{t}) \equiv \sum_{j=0}^{p} \hat{\beta}_j(\hat{t})x_k(\hat{t}),$$

for $t_{i-1} < t \le t_i$. Thus, the k^{th} spatial node in element e is parametrized on the reference element by $x_k(\hat{t})$. Summing over $j = 0,\ldots,p$, this gives

$$\begin{bmatrix} x \\ t \end{bmatrix} = \begin{bmatrix} (1-\hat{x})x_0(\hat{t}) + \hat{x}\, x_p(\hat{t}) \\ (1-\hat{t})t_{i-1} + \hat{t}\, t_i \end{bmatrix}. \tag{3.2}$$

As a result, the time component of the isoparametric map is given by the affine map

$$t_e(\hat{t}) \equiv (1-\hat{t})t_{i-1} + \hat{t}\, t_i = t_{i-1} + \hat{t}\Delta t_i \tag{3.3}$$

for $0 \le \hat{t} \le 1$. Note that this map is completely independent of the spatial variables; the subscript e in the notation of the map is only needed to provide information of the time partition in which e is located. Since the spatial nodes move in time, the spatial component of the isoparametric map does in fact depend on time. Fix $\hat{t}$ in $[0,1]$, then the spatial component of the isoparametric map is

$$x_e(\hat{x},\hat{t}) \equiv (1-\hat{x})x_0(\hat{t}) + \hat{x}\, x_p(\hat{t}) = x_0(\hat{t}) + \hat{x}\Delta x_e(\hat{t}). \tag{3.4}$$

with $\hat{x}$ in s_{ref}. For $t = t_{i-1} + \hat{t}\Delta t_i$, this is an affine map from s_{ref} to $e(t)$ in the mesh at time t.

Mesh motion

Fixing $\hat{x}$ in s_{ref} and varying $\hat{t}$ in $[0, 1]$, the space map $x_e(\hat{x}, \hat{t})$ traces out the characteristic trajectories of the mesh. Since the spatial nodes live in the (x, t)-space corresponding to $\mathcal{F}$, their trajectories are typically parameterized by t instead of $\hat{t}$: we write $x_k(t)$ as shorthand for $x_k(t_e^{-1}(t))$. Furthermore, the spatial nodes $x_k(t)$ are polynomials of degree at most p, which implies that all characteristic trajectories are polynomial paths of degree at most p in the space-time partition $\mathcal{F}_i$. These trajectories are denoted by $x(t)$ and can be aligned with the characteristics of the solution to the hyperbolic operator $u_t + b \cdot \nabla u$ by choosing $dx/dt \approx b$. Accordingly, they can be chosen to mitigate the effects of large convection velocity.

Invertibility of the isoparametric map

The inverse of the isoparametric map from e_{ref} to e in the partition $\mathcal{T}_{h,i}^p$ is given by

$$t_e^{-1}(t) = \frac{t - t_{i-1}}{\Delta t_i}$$

and

$$x_e^{-1}(x, t) = \frac{x - x_{k-1}(t)}{\Delta x_k(t)},$$

for t in $[t_{i-1}, t_i]$ and x in $e(t) = [x_{k-1}(t), x_k(t)]$. Since the inverse of the spatial component, x_e, is an affine transformation in space, it holds for fixed t that the finite element space $\mathcal{V}_h^p(t)$ is a standard finite element space of piecewise polynomials of degree at most p defined over Ω. This is in contrast to other discretization schemes where the space and time dimensions are discretized indiscriminately. That sort of scheme leads to the inverse of the spatial map, x_e^{-1}, to be a rational function in its spatial variable and is not naturally compatible with the method of lines approach used for moving finite elements. This property of the proposed finite element space justifies analyzing finite element functions on a time slice — for example, $\phi(t)$ in

$\mathcal{V}_h^p(t)$ — in a way that is consistent with the study of finite element methods for elliptic problems. Note that $\Delta t_i > 0$ and $\Delta x_k(t) > 0$ is always required for the finite element space to be well-defined.

The degrees of freedom and basis functions on the mesh

To define the basis functions of the finite element space, the basis functions on the reference element are composed with the inverse of the isoparametric maps. Fix $1 \leq i \leq m$. For $0 \leq j \leq p$, $0 \leq k \leq N_i$, and e in $\mathcal{T}_{h,i}^p$, the basis functions for node $(x_k(t_{i,j}), t_{i,j})$ are given by

$$\beta_{i,j}(t) \equiv \hat{\beta}_j \left(\frac{t - t_{i-1}}{\Delta t_i} \right),$$

and

$$\sigma_k(x,t) \equiv \sum_{(\hat{e},\ell) \in \Xi(x_k(t))} \hat{\sigma}_\ell \circ x_{\hat{e}}^{-1}(x,t), \tag{3.5}$$

where $\Xi(x_k(t))$ is the set of all elements and indices, $(\hat{e}, \ell)$, such that the node $x_k(t) = x_{\hat{e}}(\hat{x}_\ell, t)$. The set $\Xi(x_k(t))$ is a singleton when $x_k(t)$ is an interior node of the element $e(t)$. When $p = 1$, there are no interior nodes and each $x_k(t_{i,j})$, therefore, corresponds to a *hat* function with support on two neighboring elements. Note that the discontinuities in the time finite element space simplify the basis functions $\{\beta_{i,j}\}$, whereas the inter-element continuity in the spatial variable requires a summation to define the appropriate basis functions when $|\Xi(x_k(t)| \geq 1$. Also, for fixed t, we see that the basis functions $\{\sigma_k(t)\}_{k=0}^{N_i}$ form the basis for a finite element discretization of $\mathcal{H}^1(\Omega)$. As usual, a finite element function ϕ is determined by its values at the degrees of freedom in the mesh. Suppose at the degrees of freedom that ϕ takes the value $\phi\big(x_k(t_{i,j}), t_{i,j}\big) = \phi_{i,j}^{(k)}$ for $0 \leq j \leq p$, $0 \leq k \leq N_i$, and $1 \leq i \leq m$. Then, the function ϕ is given by

$$\phi(x,t) = \sum_{i=1}^{m} \sum_{j=0}^{p} \sum_{k=0}^{N_i} \beta_{i,j}(t) \sigma_k(x,t) \phi_{i,j}^{(k)}.$$

Since the time discretization is independent of the spatial discretization, the summation can be reordered as

$$\phi(x,t) = \sum_{\substack{1\le i\le m\\ 0\le j\le p}} \beta_{i,j}(t) \left(\sum_{k=0}^{N_i} \sigma_k(x,t)\phi_{i,j}^{(k)} \right) \equiv \sum_{\substack{1\le i\le m\\ 0\le j\le p}} \beta_{i,j}(t)\phi_{i,j}(x,t),$$

which shows directly that $\phi(t)$ belongs to $\mathcal{V}_h^p(t)$, which is a typical finite element space on Ω.

Due to the time dependence of the spatial basis function, the time derivative of ϕ is not generally a polynomial of degree $p-1$. Instead of using the time derivative in these finite element functions, the characteristic trajectories, $x(t)$, are used to define the characteristic derivative. Set $\hat{x}$ to be the spatial coordinate in the reference element that corresponds to the characteristic trajectory $x(t)$, passing through the point (x,t) in $\mathcal{F}$; then, we have $x(t) = x_e(\hat{x},t)$ when $x \in e(t)$. From the definition of the basis functions (3.5), we have $\sigma_k(x(t),t) = \sum_{(\hat{e},\ell)\in\Xi(x_k(t))} \hat{\sigma}_\ell(\hat{x})$, which gives

$$\frac{d}{dt}\sigma_k(x(t),t) = \frac{d}{dt} \sum_{(\hat{e},\ell)\in\Xi(x_k(t))} \hat{\sigma}_\ell(\hat{x}) = 0.$$

Thus, we define the characteristic derivative of ϕ to be

$$\partial_\tau \phi(x,t) \equiv \frac{d}{dt}\phi(x(t),t) = \sum_{\substack{1\le i\le m\\ 0\le j\le p}} \left[\frac{d}{dt}\beta_{i,j}(t) \right] \phi_{i,j}(x(t),t). \tag{3.6}$$

This corresponds to taking the time derivative on the reference element e_{ref} instead of on the domain $\mathcal{F}$. In terms of the space and time derivative, the characteristic derivative is given by the chain rule as

$$\partial_\tau \phi(x,t) = x_t(t)\phi_x(x,t) + \phi_t(x,t).$$

Note that $\partial_\tau\phi(t) \in \mathcal{V}_h^p(t)$ and recall that the characteristic derivative is also denoted by a subscript as $\phi_\tau = \partial_\tau \phi$.

Mesh discontinuities

Since there is no continuity requirement of the mesh between time partitions, the spatial elements between partitions do not necessarily align, which leads

to discontinuous finite element functions at the discrete time partitions t_i. For a discontinuous function ϕ in $\mathcal{V}_h^p$, denote the jump at $t = t_i$ by

$$[\phi](t_i) = \lim_{\delta \to 0^+} \big(\phi(t_i + \delta) - \phi(t_i - \delta)\big) \equiv \phi(t_{i^+}) - \phi(t_{i^-}).$$

In order to uniquely define ϕ in $\mathcal{V}_h^p$, take $\phi(t_i)$ to be $\phi(t_{i^-})$. Moreover, require for all χ in $\mathcal{V}_h^p(t_{i^+})$ that

$$([\phi](t_i), \chi) = 0. \tag{3.7}$$

This enforces $\phi(t_{i^+})$ to be the $\mathcal{L}_2$-projection of $\phi(t_{i^-})$ onto $\mathcal{T}_h^p(t_{i^+})$, which is weakly imposing the continuity of the finite element functions at the mesh discontinuities. As a practical note, the requirement of $[\phi](t_i)$ to be orthogonal to the mesh is convenient for the purposes of analysis. In chapter 7, interpolation is tested as an alternative for defining $\phi(t_{i^+})$.

The Jacobian matrix of the isoparametric map

Combining (3.2)–(3.4), denote the space-time isoparametric map from e_{ref} to e by

$$\iota_e(\hat{x}, \hat{t}) \equiv \begin{bmatrix} x_e(\hat{x}, \hat{t}) \\ t_e(\hat{t}) \end{bmatrix} = \begin{bmatrix} x \\ t \end{bmatrix}.$$

Denote the space-time derivative by $D_{x,t}$ so that the Jacobian matrix of the isoparametric map is given by

$$D_{x,t}\iota_e(\hat{x}, \hat{t}) = \begin{bmatrix} \Delta x_e(t) & \Delta t_i \frac{d}{dt} x_e(t) \\ 0 & \Delta t_i \end{bmatrix},$$

where $t = t_e(\hat{t})$. As noted in ([18], [65]), the Jacobian matrix describes the element e in the mesh. The diagonal of $D_{x,t}\iota_e(t)$ gives information regarding the spatial and time discretization independently: the term $\Delta x_e(t)$ gives the size of spatial element $e(t)$ in $\mathcal{T}_h^p(t)$ and Δt_i is the length of the time step. The off-diagonal terms in the Jacobian matrix describe the space-time interaction of the isoparametric map. The value of 0 in the lower left component demonstrates that the time discretization is independent of the spatial variables, which implies that the spatial nodes are aligned into time slices. As discussed above, the term $\frac{d}{dt} x_e(t)$ describes the motion

of the spatial nodes in time. Accordingly, the shape regularity of the mesh in space and in time can be maintained separately by constraining the diagonal of the Jacobian matrix $D_{x,t}\iota_e$, and the mesh motion can be controlled by the non-zero off-diagonal term.

3.2.2 The isoparametric map in higher dimensions

In the case of higher dimensions, when $d = 2$ or 3, the time component of the isoparametric map remains the same, though the spatial component becomes an affine map:

$$x_e(\hat{x}, t) = \mathcal{J}_e(t)\hat{x} + \mathcal{S}_e(t), \tag{3.8}$$

where $\mathcal{J}_e(t)$ is now a $d \times d$ matrix and $\mathcal{S}_e(t)$ is a d-vector. Note that this map is defined for $\hat{x}$ in s_{ref}, which is the unit triangle when $d = 2$ and the unit tetrahedron when $d = 3$. Both terms $\mathcal{J}_e(t)$ and $\mathcal{S}_e(t)$ are polynomials of degree p in time and are respectively analogous to $\Delta x_k(t)$ and $x_{k-1}(t)$ in the $d = 1$ case.

The finite element functions are defined as when $d = 1$, with the only difference being that there are more degrees of freedom in space; there are now $s_{d,p} = \frac{1}{d!}\prod_{i=1}^{d}(p+i)$ spatial degrees of freedom on the reference element, denoted by $\{\hat{x}_k\}_{k=1}^{s_{d,p}}$. The degrees of freedom on the time reference element are again defined by and order partitioning of the unit interval, $0 = \hat{t}_0 < \ldots < \hat{t}_p \leq 1$, and the spatial degrees of freedom are uniformly distributed on the spatial reference element. The degrees of freedom are mapped to the mesh via the isoparametric maps: for $0 \leq j \leq p$ and $0 \leq k \leq N_i$,

$$\big(x_k(t_{i,j}), t_{i,j}\big) = \iota_e(\hat{x}_\ell, \hat{t}_j),$$

for some $0 \leq \ell \leq p$ and e in $\mathcal{T}_{h,i}^p$, $i = 1, \ldots, m$.

The basis functions in time are unchanged for $d > 1$, but the spatial basis functions are constructed on a higher dimensional reference element. As a result, the spatial basis function at the point $(x_k(t_{i,j}), t_{i,j})$ is defined as in (3.5), except the set $\Xi(x_k(t_{i,j}))$ is now more complex to define explicitly in the presence of shared element edges and faces. Details of the basis functions and space-time elements for the linear case ($p = 1$) are given in detail in ([18], [65]) and generalize to higher-order finite element spaces in a straightforward manner.

As in the $d = 1$ case, the finite element functions are taken to be linear combinations of the tensor products of the spatial and temporal basis functions. For a function ϕ in $\mathcal{V}_h^p$, suppose $\phi(x_k(t_{i,j}), t_{i,j}) = \phi_{i,j}^{(k)}$. Then, the space-time basis representation of ϕ is

$$\phi(x,t) = \sum_{i=1}^{m}\sum_{j=0}^{p}\sum_{k=0}^{N_i} \beta_{i,j}(t)\sigma_k(x,t)\phi_{i,j}^{(k)}$$
$$= \sum_{\substack{1\le i\le m\\ 0\le j\le p}} \beta_{i,j}(t)\left(\sum_{k=0}^{N_i}\sigma_k(x,t)\phi_{i,j}^{(k)}\right) = \sum_{\substack{1\le i\le m\\ 0\le j\le p}} \beta_{i,j}(t)\phi(x,t).$$

As before, the finite element function $\phi(x,t)$ is generally a polynomial of degree $2p$, due to cross-terms in the tensor product; however, $\phi(x(t),t)$ is a polynomial of degree p with respect to t, as this is moving along the characteristics of the finite element mesh. For $d = 2$ and 3, the characteristic derivative of ϕ is given by

$$\partial_\tau \phi = x_t(t)\cdot\nabla\phi + \phi_t. \tag{3.9}$$

The Jacobian matrix $D_{x,t}\iota_e$ is a $(d+1)\times(d+1)$ block triangular matrix given by

$$D_{x,t}\iota_e(\hat{x},\hat{t}) = \begin{bmatrix} \mathcal{J}_e(t) & \Delta t_i \frac{d}{dt}x_e(t) \\ 0 & \Delta t_i \end{bmatrix},$$

where again, we write $t = t_e(\hat{t})$. Now, the time derivative of the spatial map, $\frac{d}{dt}x_e(t)$, is a vector function of d components describing the motion of the mesh nodes in $\Omega \subset \mathbb{R}^d$. The spatial Jacobian matrix, $\mathcal{J}_e(t)$, still describes the spatial element, $e(t)$ in $\mathcal{T}_h^p(t)$, at time t and the absolute value of the determinant $\mathcal{D}_e(t) \equiv |\det \mathcal{J}_e(t)|$ is proportional to the size of the element $e(t)$. In order to avoid colliding spatial nodes and collapsing elements, it is required that $\mathcal{D}_e(t) > 0$ for all t in the time partition of element e. Finite element meshes that have positive $\mathcal{D}_e(t)$ for all elements at all times are called *non-degenerate* and are the only types of meshes considered in this analysis.

3.3 Shape regularity of the finite element space

The shape regularity assumptions for moving meshes can be sorted into two parts: the space and time shape regularity, which take the usual form for a d-dimensional and a 1-dimensional domain, respectively, and the space-time shape regularity. The space-time shape regularity is often more evasive to describe and enforce; however, the constraints set into place in this section fit quite naturally into the error analysis, validating our choice. Additionally, the space-time shape regularity assumptions in this dissertation imply the space-time shape regularity assumptions used in the error analyses for the linear finite elements ([18],[65],[40]). The space and time shape regularity are briefly defined and then the space-time regularity, corresponding to the mesh motion, is introduced.

3.3.1 Shape regularity in space and time

Since the finite element space is a tensor product of spatial finite element spaces and a discontinuous finite element space in time, the space and time meshes must have a set of separate shape regularity assumptions. The shape regularity constraints in space are analogous to those of a standard finite element space on a d-dimensional domain and the shape regularity in the time dimension follows from the standard requirements for a 1-dimensional finite element space.

We emphasize that for all t in $(0, T]$, the spatial mesh $\mathcal{T}_h^p(t)$ is a typical finite element triangulation of Ω. Thus, for all t in the time domain, we assume that $\mathcal{T}_h^p(t)$ is a quasi-uniform mesh and that it satisfies a prescribed set of shape regularity assumptions. The assumptions we use come directly from Brenner and Scott [7]. Let $B_e(t)$ represent the largest possible ball inscribed inside of an element $e(t)$ in $\mathcal{T}_h^p(t)$. Then, we assume that there exist positive constants $\Delta x < 1$ and δ such that

$$\text{diam } e(t) \leq \Delta x \, \text{diam } \Omega, \tag{3.10}$$

and

$$\text{diam } B_e(t) \geq \delta \, \text{diam } e(t), \tag{3.11}$$

for all e in the mesh $\mathcal{T}_h^p$. The inequality (3.10) bounds the maximum element diameter in space and (3.11) is equivalent to a least angle criterion, which is needed to ensure that approximation properties on the element level can be uniformly estimated. Note that (3.11) automatically implies that the discretization of Ω is non-degenerate, meaning $\mathcal{D}_e(t) > 0$ for all $e(t)$ in $\mathcal{T}_h^p(t)$. (In a degenerate element $e(t)$, the ball $B_e(t)$ has a null diameter.) For quasi-uniformity, a minimum element size is also assumed; there must exist a positive constant $\Delta x_{\min}$ such that

$$\text{diam } e(t) \geq \Delta x_{\min} \,\text{diam } \Omega$$

for each $e(t)$ in $\mathcal{T}_h^p(t)$. These shape regularity assumptions can be imposed by controlling the eigenvalues of the Jacobian of the spatial component of the isoparametric maps.

For the quasi-uniformity of the time discretization, assume there exist a maximum time step size, Δt, and minimum step size, $\Delta t_{\min}$, such that the bound

$$0 < \Delta t_{\min} T \leq \Delta t_i \leq \Delta t T, \tag{3.12}$$

is satisfied for each time partition, $i = 1, \ldots, m$. Note that controlling Δt_i enforces constraints on the time component of the Jacobian of the isoparametric map. These restrictions ensure that the mesh is locally quasi-uniform in space directions and the time direction, although quasi-uniformity is not implied globally nor in general space-time directions.

3.3.2 Restrictions on the mesh motion (shape regularity in space-time)

In order to understand the approximation properties of the finite element space, further assumptions must be made beyond quasi-uniformity in space and quasi-uniformity in time. Specifically, the element deformation in time must be controlled. This is essentially imposing a *space-time* shape regularity constraint for these moving tensor elements. It is important, however that the mesh motion x_t is not directly bounded, as this would limit the extent to which the mesh could align with the convection term ([57],[55],[18]). As in subsection 3.3.1, the Jacobian

matrix of the isoparametric map, $D_{x,t}\iota_e$, is used to impose shape regularity; in this case, the mesh regularity assumptions can be interpreted as constraining the off-diagonal block of the Jacobian matrix.

Fix e to be an element in $\mathcal{T}_{h,i}^p$ and $t_{i-1} \leq t \leq t_i$. Then, the Jacobian matrix at time t can be represented as

$$\mathcal{J}_e(t) = \big(\mathcal{R}_e(t) + \Delta t_i \mathcal{H}_e(t)\big)\mathcal{J}_e(t_{i-1^+}) \tag{3.13}$$

for some orthogonal *rotation* matrix, $\mathcal{R}_e(t)$, and *evolution* matrix, $\mathcal{H}_e(t)$. The matrix $\mathcal{R}_e + \Delta t_i \mathcal{H}_e$ is constrained to have polynomial entries of degree at most p throughout the time partition. As the name suggests, the matrix $\mathcal{R}_e(t)$ describes the element rotation in time, and the evolution matrix describes the deformation of the shape of the element. If an element is merely translated in time, without rotation or changing shape, then the Jacobian matrix, $\mathcal{J}_e(t)$, remains unchanged and the shift vector $\mathcal{S}_e(t)$ in (3.8) varies in time.

Let $\rho(\cdot)$ represent the spectral radius norm for $d \times d$ matrices. For space-time regularity, it is assumed that the evolution matrix $\mathcal{H}_e$ has a uniformly bounded spectral radius throughout the time step; namely, there exists some positive constant μ that does not depend on e or t such that

$$\rho\big(\mathcal{H}_e(t)\big) \leq \mu. \tag{3.14}$$

This can be interpreted as bounding the relative change in shape and size of the element over time.

Notice that if the mesh motion, x_t, is large due solely to rapid spatial translation of the element, then the shift vector, $\mathcal{S}_e(t)$ in (3.8), is nonzero and the Jacobian matrix of the spatial map is constant, $\mathcal{J}_e(t) = \mathcal{J}_e(t_{i-1^+})$. If the mesh velocity is large due to a perfect rotation of the element, then $\mathcal{J}_e(t) = \mathcal{R}_e(t)\mathcal{J}_e(t_{i-1^+})$. In both cases, the shape and size of the element does not change and, hence, the evolution matrix is zero. This indicates that elements may translate or rotate and still maintain perfect shape regularity, whereas elements that deform throughout the time partition do not possess such regularity. An example of an ideally evolving mesh could take place on a circular domain with a uniform clockwise or counterclockwise flow. In this case, the elements could flow perfectly with the

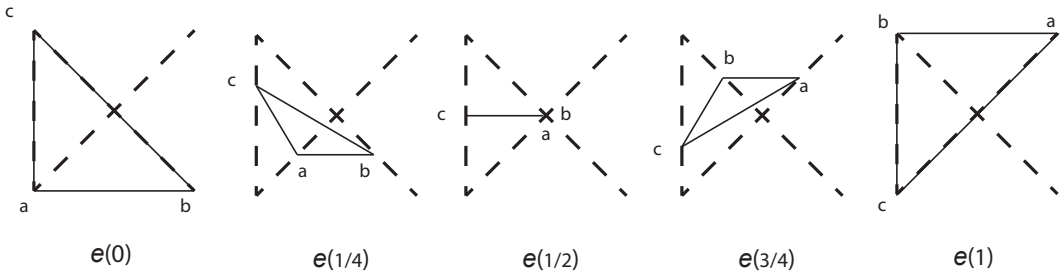

Figure 3.3: A twisting element with $d = 2$ and $p = 1$ plotted at discrete time slices. The spatial nodes are labeled and their linear trajectories are represented by the dashed lines and superimposed on each time slice. From the time $t = 0$ to $t = 1$, the element seems to merely rotate; however, due to the linear paths traced out by the spatial nodes, the element degenerates when $t = 1/2$.

convection term, nullifying the hyperbolic effects of the differential equation, with $\rho(\mathcal{H}_e(t)) = 0$ for all elements e in the mesh and t in the time domain.

Unfortunately, in the case of rotating elements, this is an oversimplification since the rotation matrix $\mathcal{R}_e(t)$ cannot perfectly preserve orthogonality when its entries are polynomials; this implies that there is inherently some deformation that must take place ($\mathcal{H}_e \neq 0$) to allow the element to rotate while preserving degree p polynomial trajectories of the spatial nodes. This deformation of the element corresponds to twisting the element rather that rotating it. An example of this twisting leading to a nearly degenerate element is depicted in figure 3.3. Note that higher order mesh motion — larger p — is very useful in offsetting this "twist" effect.

The shape regularity assumption (3.14), therefore, serves as a constraint on the off-diagonal term of the Jacobian matrix, $D_{x,t}\iota_e$, though this constraint does not penalize mesh motion; it only constrains the amount of shape and size deformation that an element can undergo within a time partition. In Carlson

and Miller [28], is was noted that adaptive time stepping schemes always chose smaller time steps when spatial nodes slip into shock layers or exit the domain at a boundary. In both cases, the relative size of the element shrinks rapidly. The smaller time steps indicate that the error in the computed solution is larger at these times, which agrees with the conjecture that the mesh regularity may be compromised when the spatial nodes shrink or grow quickly (relative to their sizes at the beginning of the time partition).

Assuming a non-degenerate finite element space and the space-time shape regularity bound (3.14), it follows that

$$\rho\big(\mathcal{J}_e(t)\mathcal{J}_e^{-1}(t_{i-1^+})\big) = \rho\big(\mathcal{R}_e(t) + \Delta t_i \mathcal{H}_e(t)\big) \leq 1 + \mu \Delta t_i \tag{3.15}$$

and, for $\tilde{c}_{\mu,d} = [(1+\mu\Delta t_i)^d - 1]/\Delta t_i = \mathcal{O}(1)$ and $\Delta t_i \leq 1/2\tilde{c}_{\mu,d}$,

$$\begin{aligned}(1 - \tilde{c}_{\mu,d}\Delta t_i) \leq (1-\mu\Delta t_i)^d \leq \frac{\mathcal{D}_e(t)}{\mathcal{D}_e(t_{i-1^+})} \\ = \det\left(\mathcal{J}_e(t)\mathcal{J}_e^{-1}(t_{i-1^+})\right) \leq (1+\mu\Delta t_i)^d \leq (1+\tilde{c}_{\mu,d}\Delta t_i),\end{aligned} \tag{3.16}$$

since $\mathcal{R}_e$ is an orthogonal matrix. Inequalities (3.15) and (3.16) will provide the necessary bounds for proving the results of chapter 4, which establish the subspace approximation properties of finite element spaces built on moving meshes.

Consequences of shape regularity

The restriction that $\mathcal{T}_h^p$ is non-degenerate is a minimal shape regularity assumption and additional constraints on the blocks of $D_{x,t}\iota_e$ can be used to enforce stronger shape regularity assumptions for the mesh. Selectively imposing these additional shape regularity assumptions outlined above leads to better approximation properties of the finite element space and improves error estimates for the finite element solutions found by the methods in chapters 5 and 6. The efficacy of these regularity assumptions is validated in the numerical results of chapter 7.

Chapter 3, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 4

Preliminary Results and Approximation Properties of the Finite Element Space

This chapter establishes the relationships of shape regularity of the mesh and the approximation properties of the finite element space. Since we are considering tensor product finite elements, these approximation properties are well understood in the absence of moving nodes [7]; some preliminary results must be proven to handle the motion of the mesh before providing analysis of the moving finite elements. We begin by introducing the relevant notation for the space-time domain and the finite element space. Then, *shift functions* are introduced to facilitate the analysis of the moving finite element space. These functions serve as an isomorphism between finite element spaces with moving meshes and static meshes, meaning the meshes do not move in time. Finally, some approximation properties of the space-time moving finite element space are proven. It is important to note that these approximation properties solely depend on the mesh and basis functions of the finite element space; they are independent of the differential equation and the methods used to solve the differential equation. Ultimately, these approximation properties can be used to provide *a posteriori* error estimates, which are useful for understanding the convergence rate of a finite element method and defining adaptive meshing schemes.

4.1 Norms and Notation

In this section, norms and semi-norms over the spatial domain and time domain are defined. These norms are then employed together to define norms over the space-time domain. Some of the semi-norms defined in this section depend on the discretizations of the spatial domain. These dependencies signify that the error of the finite element solution can be reduced by carefully selecting the finite element space to be an appropriate fit for the given differential equation.

4.1.1 Space Norms

Multi-index notation is used to represent spatial derivatives, although time and characteristic derivatives do not follow this convention in order to emphasize the property that the finite element functions are the tensor products of polyno-

mials in space and polynomials time. Thus, for $\alpha = (\alpha_1, \ldots, \alpha_d) \geq 0$, the α spatial derivative of some function v is denoted

$$D_x^\alpha v(x_1, \ldots, x_d) \equiv \frac{\partial^{\alpha_1}}{\partial {x_1}^{\alpha_1}} \cdots \frac{\partial^{\alpha_d}}{\partial {x_d}^{\alpha_d}} v(x_1, \ldots, x_d).$$

In this notation, denote $|\alpha| = \sum_{i=1}^d \alpha_i$ and $\alpha! = \prod_{i=1}^d \alpha_i!$.

Let v be a function in $\mathcal{H}^k(\Omega)$, where k is some non-negative integer. Then, the $\mathcal{H}^k(\Omega)$ semi-norm and norm are given by

$$|v|_k = \left[\sum_{|\alpha|=k} (D_x^\alpha v, D_x^\alpha v) \right]^{1/2}$$

and

$$\|v\|_k = \left[\sum_{|\alpha|\leq k} (D_x^\alpha v, D_x^\alpha v) \right]^{1/2}.$$

Note that $\mathcal{H}^0(\Omega) = \mathcal{L}_2(\Omega)$ and, accordingly, $\|\cdot\|_0$ is referred to as the $\mathcal{L}_2$-norm.

The dual to the $\mathcal{H}^1(\Omega)$-norm is indirectly used in this analysis as well; a mesh-dependent version of this norm is introduced, as originally proposed by Dupont in [37]. For v in $\mathcal{L}_2(\Omega)$, define the semi-norm

$$\|v\|_{(-1,\mathcal{V}_h^p(t))} = \sup_{\substack{\chi \in \mathcal{V}_h^p(t) \\ \chi \neq 0}} \frac{|(v,\chi)|}{\|\chi\|_1}.$$

While this is a semi-norm on $\mathcal{V}$, its restriction to the finite element space is truly a norm. Furthermore, since $\mathcal{V}_h^p(t)$ is a subset of $\mathcal{H}^1(\Omega)$, note that $\|v\|_{(-1,\mathcal{V}_h^p(t))} \leq \|v\|_{-1}$ for all v in $\mathcal{L}_2(\Omega)$ and t in $(0,T]$. Here $\|\cdot\|_{-1}$ takes on the standard definition of the dual $\mathcal{H}^1(\Omega)$-norm:

$$\|v\|_{-1} = \sup_{\substack{\chi \in \mathcal{H}^1(\Omega) \\ \chi \neq 0}} \frac{|(v,\chi)|}{\|\chi\|_1}.$$

We also take $\|\cdot\|_\infty$ to measure the maximum value of a function over Ω. For a bounded function v on Ω,

$$\|v\|_\infty \equiv \max_{x \in \Omega} |v|.$$

4.1.2 Bounding the bilinear form

Using these norms, we have lemma 1 for bounding the bilinear form. An important note here is that improved alignment of the convection and mesh velocities lead to sharper bounds. This suggests that aligning the mesh motion with the convection velocity reduces the bounding constant — ultimately improving the stiffness of the differential equation and allowing for larger time steps. Hence, the method of characteristics, where $x_t = b$, can be an effective choice for achieving nearly optimal error bounds.

Lemma 1. *Suppose there exists a constant $\kappa > 0$ such that the mesh motion satisfies*

$$\|b - x_t\|_\infty \leq \kappa. \tag{4.1}$$

Then, given functions $u, v \in \mathcal{H}^1(\Omega)$, there exist positive constants $C_{\mathcal{A}}$ and $C'_{\mathcal{A},\kappa}$ such that

$$\mathcal{A}_\tau(u, u; t) \geq C_{\mathcal{A}}\|u\|_1^2 - C'_{\mathcal{A},\kappa}\|u\|_0^2 \tag{4.2}$$

and, furthermore, the bilinear form is bounded,

$$\mathcal{A}_\tau(u, v; t) \leq C_{\mathcal{A},\kappa}\|u\|_1\|v\|_1. \tag{4.3}$$

Proof. Since $a \geq \bar{a} > 0$ and $c \geq \bar{c} \geq 0$, we can use (4.1) to get

$$\begin{aligned}\mathcal{A}_\tau(u, u; t) &= \big(a(t)\nabla u, \nabla u\big) + \big((b(t) - x_t(t)) \cdot \nabla u, u\big) + \big(c(t)u, u\big) \\ &\geq \bar{a}\|\nabla u\|_0^2 - \|b - x_t\|_\infty\|\nabla u\|_0\|u\|_0 + \bar{c}\|u\|_0^2 \\ &\geq (\bar{a} - \delta)|u|_1^2 + \left(\bar{c} - \frac{\kappa}{4\delta}\right)\|u\|_0^2, \end{aligned} \tag{4.4}$$

where δ is an arbitrary positive constant. Choose δ to be small, say $\delta = \bar{a}/2$, to show that there exist constants $C_{\mathcal{A}} = \bar{a}/2 > 0$ and $C'_{\mathcal{A},\kappa} = C_{\mathcal{A}} + \kappa/2\bar{a} - \bar{c}$ such that

$$\mathcal{A}_\tau(u, u; t) \geq C_{\mathcal{A}}\|u\|_1^2 - C'_{\mathcal{A},\kappa}\|u\|_0^2. \tag{4.5}$$

For the upper bound, we use (4.1) to show

$$\mathcal{A}_\tau(u, v; t) \leq \|a\|_\infty\|\nabla u\|_0\|\nabla v\|_0 + \kappa\|\nabla u\|_0\|v\|_0 + \|c\|_\infty\|u\|_0\|v\|_0 \leq C_{\mathcal{A},\kappa}\|u\|_1\|v\|_1, \tag{4.6}$$

where $C_{\mathcal{A},\kappa} = 2\left(\max\left\{\|a\|_\infty, \|c\|_\infty\right\} + \kappa\right)$. This proves the lemma. □

Assuming that the mesh motion is sufficiently aligned with the convection velocity, coercivity of $\mathcal{A}$ follows: from (4.5) and $\delta = \bar{a}/2$, if $\kappa < 2\bar{a}\bar{c}$, then

$$\mathcal{A}_\tau(u,u) \geq \min\{\bar{a}/2, \bar{c} - \kappa/2\bar{a}\}\|u\|_1^2. \tag{4.7}$$

While this result is not directly used in the error analysis, it indicates that choosing mesh motion where $b - x_t$ is small leads to a more well-conditioned stiffness matrix, providing more flexibility in the maximum permissible time steps.

Recall that the shape regularity constraint (3.14) bounds the amount of shape and size distortion that an element undergoes within a time partition. This is important as it affects the condition number of the finite element mass matrix, as discussed in section 5.3. However, the results in this section demonstrate that the choice for small $b - x_t$ will lead to a better conditioned stiffness matrix in the linear system. In many practical cases where the convection velocity is "well behaved," aligning the mesh velocity with the convection does not compromise the shape regularity of the mesh and the most care must be taken to ensure mesh regularity when elements flow into or out of the domain boundary or slip into shock layers. The former case is typically not an issue when the convection velocity is divergence free or orthogonal to the outward normal of the spatial boundary, as in (2.9). The effects of the tradeoff between aligning the convection and mesh velocities and preserving space-time shape regularity in the mesh are examined in the numerical results section.

4.1.3 Time norms

We now focus on the time norms used in this analysis. For a function v defined on the time interval $(0,T]$, the $\mathcal{L}_2(0,T]$-norm is

$$\|v\|_{\mathcal{L}_2(0,T]} = \left[\int_0^T v^2(t)\ dt\right]^{1/2}.$$

Ultimately, moving finite element methods in this dissertation use time stepping methods to propagate the solution in time. Accordingly, it is useful to define a discretized version of $\|\cdot\|_{\mathcal{L}_2(0,T]}$; this is analogous to how $\|\cdot\|_{(-1,\mathcal{V}_h^p(t))}$ is a discretized version of $\|\cdot\|_{-1}$. To discretize the norm over the time domain, we first note that

time stepping methods have a natural correspondence to quadrature rules, where the collocation points of the time stepping method are chosen to coincide with the knots of some corresponding quadrature rule. Let $\hat{\mathcal{Q}}$ represent a *reference quadrature rule*, defined over the time reference element $[0,1]$, with knots given by $\{\hat{t}_j\}_{j=1}^p$. It is required that the knots of the $\hat{\mathcal{Q}}$ satisfy

$$0 < \hat{t}_1 < \ldots < \hat{t}_p \leq 1, \tag{4.8}$$

and that the weights $\{w_j\}$ are all positive for $j = 1, \ldots, p$. For convenience, let $\hat{t}_0 = 0$ be coincident with a degree of freedom on the reference element. Note that this knot may not be used in the reference quadrature rule.

From $\hat{\mathcal{Q}}$, it is possible to define a composite quadrature rule over the time partitions; let v be in $\mathcal{L}_2(0,T]$ and set

$$\mathcal{Q}_i(v) \equiv \sum_{j=1}^{p} w_j v(t_{i-1} + \hat{t}_j \Delta t_i) \approx \frac{1}{\Delta t_i} \int_{t_{i-1}}^{t_i} v(t)\ dt$$

so that, summing over i, for $1 \leq i \leq m$, gives

$$\mathcal{Q}(v) \equiv \sum_{i=1}^{m} \Delta t_i \mathcal{Q}_i(v) \approx \int_0^T v(t)\ dt. \tag{4.9}$$

A reference quadrature rule has *order of exactness* q when any polynomial of degree at most q is integrated exactly: let $r(t)$ be an arbitrary polynomial of degree q, then

$$\hat{\mathcal{Q}}(r) = \int_0^1 r(\hat{t})\ d\hat{t},$$

if $\hat{\mathcal{Q}}$ has order of exactness q. It is obvious that any interpolatory quadrature rule has order of exactness at least p, where an interpolatory quadrature rule is defined to satisfy $\hat{\mathcal{Q}}(f) = \int_0^1 \mathcal{I}f(\hat{t})\ d\hat{t}$ for the interpolant, $\mathcal{I}$, of degree p with nodes at $\{\hat{t}_j\}_{j=0}^p$. From the order of exactness, we use Taylor's theorem to find a representation of the quadrature error. Taylor's theorem states that any f in $\mathcal{C}^{q+1}[0,1]$ satisfies

$$f(\hat{t}) = \sum_{k=0}^{q} \frac{1}{k!} f^{(k)}(0)\hat{t}^k + \frac{1}{q!} \int_0^1 f^{(q+1)}(\zeta)(\hat{t} - \zeta)_+^q\ d\zeta$$

for $\hat{t}$ in $[0,1]$, where $(\hat{t}-\zeta)_+ \equiv \max\{0, \hat{t}-\zeta\}$ and $f^{(k)}$ represents the k^{th} derivative of f. As a result, we integrate with respect to $\hat{t}$ and subtract $\hat{\mathcal{Q}}(f)$ to get

$$\int_0^1 f(\hat{t})\ d\hat{t} - \hat{\mathcal{Q}}(f) = \int_0^1 f^{(q+1)}(\zeta) K_{\hat{\mathcal{Q}},q}(\hat{t},\zeta)\ d\zeta, \tag{4.10}$$

where

$$K_{\hat{\mathcal{Q}},q}(\zeta) \equiv \int_0^1 (\hat{t}-\zeta)_+^q\ d\hat{t} - \hat{\mathcal{Q}}\big((\cdot-\zeta)_+^q\big).$$

This is a restatement of the Peano kernel theorem and shows that the quadrature error can be bounded by the norm $f^{(q+1)}$ multiplied by the kernel function, $K_{\hat{\mathcal{Q}},q}$, that can be bounded independently of f. We use the Cauchy-Schwarz inequality to show

$$\int_0^1 f(\hat{t})\ d\hat{t} - \hat{\mathcal{Q}}(f) \le C_{\hat{\mathcal{Q}},q} \left\{ \int_0^1 \left[f^{(q+1)}(\zeta)\right]^2\ d\zeta \right\}^{1/2}. \tag{4.11}$$

In this bound, the constant $C_{\hat{\mathcal{Q}},q} = \sqrt{\int_0^1 K_{\hat{\mathcal{Q}},q}(\zeta)\ d\zeta}$.

The order of exactness of a reference quadrature rule can be used to determine how well integration is approximated by using the composite rule $\mathcal{Q}$. To see this, suppose $\hat{\mathcal{Q}}$ has order of exactness q and denote $\hat{f}_i(\hat{t}) = f(t_{i-1} + \hat{t}\Delta t_i)$

$$\begin{aligned}\int_{t_{i-1}}^{t_i} f(t)\ dt - \Delta t_i \mathcal{Q}_i(f) &= \Delta t_i \left[\int_0^1 \hat{f}_i(\hat{t})\ d\hat{t} - \hat{\mathcal{Q}}(\hat{f}_i) \right] \\ &\le C_{\hat{\mathcal{Q}},q} \Delta t_i \left\{ \int_0^1 \left[\hat{f}_i^{(q+1)}(\hat{t})\right]^2\ d\hat{t} \right\}^{1/2} = C_{\hat{\mathcal{Q}},q} \Delta t_i^q \left\{ \int_{t_{i-1}}^{t_i} \left[f^{(q+1)}(t)\right]^2\ dt \right\}^{1/2}.\end{aligned}$$

Summing over the time partitions, and assuming quasi-uniformity of the time partition (3.12), we have for any f in $\mathcal{C}^{q+1}(0,T]$

$$\int_0^T f(t)\ dt - \mathcal{Q}(f) \le C_{T,\hat{\mathcal{Q}},q} \Delta t^q \left\{ \int_0^T \left[f^{(q+1)}(t)\right]^2\ dt \right\}^{1/2} = C_{T,\hat{\mathcal{Q}},q} \Delta t^q \|f^{(q+1)}\|_{\mathcal{L}_2(0,T]}. \tag{4.12}$$

4.1.4 Collocation nodes

As mentioned before, the knots of the composite quadrature rule correspond to the collocation points of some time-stepping method; these collocation nodes

will be used to determine the space-time finite element method and are denoted as

$$t_{i,j} \equiv t_{i-1} + \hat{t}_j \Delta t_i,$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, p$. It is also convenient to denote the limiting value $t_{i,0} \equiv t_{i-1^+}$, in the sense that $v(t_{i,0}) = v(t_{i-1^+}) = \lim_{\delta \to 0^+} v(t_{i-1} + \delta)$. This will be useful in referencing the "initial value" of v on the i^{th} time partition.

4.1.5 The energy norm

For measuring functions in $\mathcal{V}$, we use a space-time semi-norm that depends on both the mesh and quadrature rule associated with the finite element method. The *energy* semi-norm that is used for functions v in $\mathcal{V}$ is defined as

$$|\!|\!| v |\!|\!|^2_{(\mathcal{V}_h^p, \mathcal{Q})} \equiv \max_{\substack{1 \le i \le m \\ 1 \le j \le p}} \|v(t_{i,j})\|_0^2 + \mathcal{Q}\left(\|v_\tau(\cdot)\|^2_{(-1, \mathcal{V}_h^p(\cdot))} + \|v(\cdot)\|_1^2 \right). \tag{4.13}$$

This energy semi-norm will be used to measure the error of finite element approximations to functions in $\mathcal{V}$. Additionally, the semi-norm is defined continuously over the spatial domain, Ω, and discretely over the time domain, $(0, T]$, which corresponds to the fact that the finite element solution is determined by a method of lines approach. As with the semi-norm $\|\cdot\|_{(-1, \mathcal{V}_h^p(t))}$, this energy semi-norm restricted to the finite element space is again a true norm. Furthermore, this energy semi-norm can be thought of as a discretized version of the space-time norm given by

$$\max_{0 < t \le T} \|v(t)\|_0^2 + \int_0^T \left(\|v_\tau(t)\|_{-1}^2 + \|v(t)\|_1^2 \right) \, dt > 0,$$

for nonzero v in $\mathcal{V}$. When the mesh and quadrature rule are assumed to be known, the subscript is dropped, and we write $|\!|\!| \cdot |\!|\!|_{(\mathcal{V}_h^p, \mathcal{Q})} = |\!|\!| \cdot |\!|\!|$.

4.2 Preliminary results for moving finite element spaces

The purpose of this section is to demonstrate the approximation properties of the moving finite element space $\mathcal{V}_h^p$. Due to the moving nodes in the discretiza-

tion, it is tricky to prove these approximation properties directly. Therefore, we set up a framework for *shifting* the moving finite element space to a static finite element space, which is just a standard tensor product finite element space. We first prove some technical lemmas so that our task is reduced to demonstrating approximation properties of these standard tensor product finite element spaces, which are well understood from elementary finite element analysis [7].

4.2.1 Shifting finite element functions

Let ϕ be a function in the finite element space $\mathcal{V}_h^p(t)$ for some t in the time partition $(t_{i-1}, t_i]$. We *shift* ϕ onto the mesh of $\mathcal{V}_h^p(t_{i-1^+})$, at the beginning of the time partition, by replacing the basis functions for $\mathcal{V}_h^p(t)$ with with their corresponding basis functions in $\mathcal{V}_h^p(t_{i-1^+})$, while preserving the basis coefficients. Formally, one can define the *shift* of ϕ element-wise to be function composition

$$\tilde{\phi}(\tilde{x}) \equiv \phi \circ x_e\big(x_e^{-1}(\tilde{x}, t_{i-1^+}), t\big) \tag{4.14}$$

for $\tilde{x}$ in element $e(t_{i-1^+})$. It follows from the definition of the spatial map (3.8) that a characteristic trajectory,

$$x(t) = x_e\big(x_e^{-1}(\tilde{x}, t_{i-1^+}), t\big) = \mathcal{J}_e(t)\mathcal{J}_e^{-1}(t_{i-1^+})\big[\tilde{x} - \mathcal{S}_e(t_{i-1^+})\big] + \mathcal{S}_e(t),$$

is affine in space and a polynomial of degree p in time. Moreover, these trajectories are injective when the mesh is non-degenerate, indicating that these mesh trajectories do not tangle. Using the mesh characteristics as given above, it is useful to conceptualize $\tilde{\phi}$ as

$$\tilde{\phi}(x(t_{i-1^+}), t) = \phi(x(t), t), \tag{4.15}$$

so that the shift of a function is merely re-parametrization of ϕ onto a non-moving mesh.

Using the basis functions defined in (3.5), it is straightforward to verify that $\tilde{\phi} \in \mathcal{V}_h^p(t_{i-1^+})$. Let $\tilde{x} \in e(t_{i-1^+})$. Then, we see that $\tilde{\phi}$ is a linear combination of the

basis functions of $\mathcal{V}_h^p(t_{i-1^+})$,

$$\begin{aligned}
\tilde{\phi}(\tilde{x}) &= \phi \circ x_e\big(x_e^{-1}(\tilde{x}, t_{i-1^+}), t\big) \\
&= \sum_{k=0}^{N_i} \phi_k \sigma_k\Big(x_e\big(x_e^{-1}(\tilde{x}, t_{i-1^+}), t\big), t\Big) \\
&= \sum_{k=0}^{N_i} \phi_k \sum_{(\hat{e},\ell)\in\Xi(x_k(t))} \hat{\sigma}_\ell \circ x_{\hat{e}}^{-1}\Big(x_e\big(x_e^{-1}(\tilde{x}, t_{i-1^+}), t\big), t\Big) \\
&= \sum_{k=0}^{N_i} \phi_k \sum_{(\hat{e},\ell)\in\Xi(x_k(t_{i-1^+}))} \hat{\sigma}_\ell \circ x_{\hat{e}}^{-1}(\tilde{x}, t_{i-1^+}) \\
&= \sum_{k=0}^{N_i} \phi_k \sigma_k(\tilde{x}, t_{i-1^+}), \qquad (4.16)
\end{aligned}$$

where we used the fact that $\Xi(x_k(t)) = \Xi(x_k(t_{i-1^+}))$, since $x_k(t)$ is just a time parametrization of the k^{th} spatial node.

Hence, if we shift $\phi(t)$ onto the mesh $\mathcal{T}_h^p(t_{i-1^+})$ by

$$\tilde{\phi}(\tilde{x}, t) = \phi(x(t), t)$$

for all t in $(t_{i-1}, t_i]$, then the resulting shift will live in $\mathcal{V}_h^p(t_{i-1^+})$ for all t in the time partition, $i = 1, \ldots, m$. Thus, the shift actually lives in a finite element space with static nodes that do not move and we see that non-degenerate finite element spaces are isomorphic to basic space-time tensor elements. Using the shape regularity assumption (3.14), the following technical lemma is proven. Loosely speaking, this lemma shows that the shift of a function is an $\mathcal{O}(\Delta t_i)$ perturbation over the spatial domain in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms.

Lemma 2 (Shift Lemma)**.** *Let $\phi, \chi \in \mathcal{V}_h^p(t)$ and $\tilde{\phi}, \tilde{\chi} \in \mathcal{V}_h^p(t_{i-1^+})$ represent a pair of finite element functions and their shifts, respectively, on a non-degenerate mesh partition $\mathcal{T}_{h,i}^p$ that satisfies (3.14) for each element in the partition. If $\Delta t_i \leq 1/2\tilde{c}_{\mu,d}$, as defined in (3.16), then*

$$\Big|\big(\phi, \chi\big) - \big(\tilde{\phi}, \tilde{\chi}\big)\Big| \leq \tilde{c}_{\mu,d}\Delta t_i \frac{\big\|\tilde{\phi}\big\|_0^2 + \big\|\tilde{\chi}\big\|_0^2}{2}, \qquad (4.17)$$

$$\Big|\big(\phi, \chi\big) - \big(\tilde{\phi}, \tilde{\chi}\big)\Big| \leq \tilde{c}_{\mu,d} \frac{\big\|\Delta t_i\tilde{\phi}\big\|_0^2 + \big\|\tilde{\chi}\big\|_0^2}{2}, \qquad (4.18)$$

and there exists a positive constant $C_{\mu,d}$ such that

$$\left| \|\phi\|_0^2 - \|\tilde{\phi}\|_0^2 \right| \leq C_{\mu,d} \Delta t_i \|\tilde{\phi}\|_0^2, \tag{4.19}$$

$$\left| \|\phi\|_1^2 - \|\tilde{\phi}\|_1^2 \right| \leq C_{\mu,d} \Delta t_i \|\tilde{\phi}\|_1^2. \tag{4.20}$$

The restriction $\Delta t_i \leq 1/2\tilde{c}_{\mu,d}$ is slightly stricter than necessary in this case because we only require $1 - \tilde{c}_{\mu,d}\Delta t_i > 0$. Nevertheless, choosing $\Delta t_i \leq 1/2\tilde{c}_{\mu,d}$ provides a practical means for avoiding "over-stepping" in time, which could lead to poor regularity of the mesh — or even degenerate meshes.

Proof. The basic mechanism that drives this proof is an element-wise change of variables

$$x = x_e\big(x_e^{-1}(\tilde{x}, t_{i-1^+}), t\big) \tag{4.21}$$

and the fact that the shape regularity (3.14) implies inequalities (3.15) and (3.16), as shown above. This change of variables is well-defined since the mesh is non-degenerate and gives

$$\int_{e(t)} \phi(x)\chi(x)\ dx = \int_{e(t_{i-1^+})} \tilde{\phi}(\tilde{x})\tilde{\chi}(\tilde{x})\ \frac{\mathcal{D}_e(t)}{\mathcal{D}_e(t_{i-1^+})} d\tilde{x},$$

which leads to

$$\begin{aligned}
&\left| \int_{e(t)} \phi(x)\chi(x)\ dx - \int_{e(t_{i-1^+})} \tilde{\phi}(\tilde{x})\tilde{\chi}(\tilde{x})\ d\tilde{x} \right| \\
&\qquad = \left| \int_{e(t_{i-1^+})} \tilde{\phi}(\tilde{x})\tilde{\chi}(\tilde{x})\ \left(1 - \frac{\mathcal{D}_e(t)}{\mathcal{D}_e(t_{i-1^+})}\right) d\tilde{x} \right| \\
&\qquad \leq \left| 1 - \frac{\mathcal{D}_e(t)}{\mathcal{D}_e(t_{i-1^+})} \right| \sqrt{\int_{e(t_{i-1^+})} \tilde{\phi}^2(\tilde{x})\ d\tilde{x} \int_{e(t_{i-1^+})} \tilde{\chi}^2(\tilde{x})\ d\tilde{x}}.
\end{aligned} \tag{4.22}$$

Since $\Delta t_i \leq 1/2\tilde{c}_{\mu,d}$, we use bound (3.16) to show

$$0 < 1 - \tilde{c}_{\mu,d}\Delta t_i \leq (1 - \mu\Delta t_i)^d \leq \frac{\mathcal{D}_e(t)}{\mathcal{D}_e(t_{i-1^+})} \leq (1 + \mu\Delta t_i)^d \leq 1 + \tilde{c}_{\mu,d}\Delta t_i, \tag{4.23}$$

for each element e in the mesh partition. Combining (4.22), (4.23), and summing

of the elements in the partition, one recovers

$$\left|(\phi,\chi)-(\tilde{\phi},\tilde{\chi})\right| \leq \sum_{e\in\mathcal{T}_{h,i}^{p}} \left|1-\frac{\mathcal{D}_e(t)}{\mathcal{D}_e(t_{i-1^+})}\right| \sqrt{\int_{e(t_{i-1^+})} \tilde{\phi}^2(\tilde{x})\ d\tilde{x} \int_{e(t_{i-1^+})} \tilde{\chi}^2(\tilde{x})\ d\tilde{x}}$$
$$\leq \tilde{c}_{\mu,d}\Delta t_i \sum_{e\in\mathcal{T}_{h,i}^{p}} \sqrt{\int_{e(t_{i-1^+})} \tilde{\phi}^2(\tilde{x})\ d\tilde{x} \int_{e(t_{i-1^+})} \tilde{\chi}^2(\tilde{x})\ d\tilde{x}}.$$

Now, we make use of the identity that for scalars a and b,

$$2ab \leq a^2+b^2.$$

By setting $\alpha=1$ and $\alpha=\Delta t_i$,

$$\alpha\sqrt{\int_{e(t_{i-1^+})} \tilde{\phi}^2(\tilde{x})\ d\tilde{x} \int_{e(t_{i-1^+})} \tilde{\chi}^2(\tilde{x})\ d\tilde{x}} \leq \frac{\int_{e(t_{i-1^+})} \left(\alpha\tilde{\phi}(\tilde{x})\right)^2\ d\tilde{x} + \int_{e(t_{i-1^+})} \tilde{\chi}^2(\tilde{x})\ d\tilde{x}}{2},$$

and we get bounds (4.17) and (4.18), respectively. From the bound (4.17), the bound on the $\mathcal{L}_2(\Omega)$ norm difference (4.19) is an immediate consequence of choosing $\chi=\phi$.

For bound (4.20), the difference is split into a few easily bounded terms. We have

$$\left|\|\phi\|_1^2-\|\tilde{\phi}\|_1^2\right| \leq \left|\|\nabla\phi\|_0^2-\|\nabla\tilde{\phi}\|_0^2\right| + \left|\|\phi\|_0^2-\|\tilde{\phi}\|_0^2\right|, \tag{4.24}$$

where the second term is bounded immediately by (4.19). The first term can be broken down even further by shifting the gradient $\nabla\phi$, as follows:

$$\left|\|\nabla\phi\|_0^2-\|\nabla\tilde{\phi}\|_0^2\right| \leq \left|\|\nabla\phi\|_0^2-\|\widetilde{\nabla\phi}\|_0^2\right| + \left|\|\nabla\tilde{\phi}\|_0^2-\|\widetilde{\nabla\phi}\|_0^2\right|, \tag{4.25}$$

for which the first term is bounded by applying (4.19) to the function $\nabla\phi$. Therefore, all that remains is to bound the difference $\left|\|\nabla\tilde{\phi}\|_0^2-\|\widetilde{\nabla\phi}\|_0^2\right|$. Note that

$$\widetilde{\nabla\phi} = (\nabla\phi)\circ x_e\big(x_e^{-1}(x,t_{i-1^+}),t\big),$$

whereas

$$\nabla\tilde{\phi} = \nabla\big(\phi\circ x_e\big(x_e^{-1}(\cdot,t_{i-1^+}),t\big)\big) = \big[\mathcal{J}_e(t)\mathcal{J}_e^{-1}(t_{i-1^+})\big]^T\widetilde{\nabla\phi}.$$

Since bound (3.14) holds on every element, it follows that

$$\left|\|\nabla\tilde{\phi}\|_0^2-\|\widetilde{\nabla\phi}\|_0^2\right| \leq \mu\Delta t_i\|\widetilde{\nabla\phi}\|_0^2.$$

Applying this bound to (4.24) yields

$$\left| \|\nabla\phi\|_0^2 - \|\nabla\tilde{\phi}\|_0^2 \right| \leq (\mu + \tilde{c}_{\mu,d})\Delta t_i \|\widetilde{\nabla\phi}\|_0^2 \leq 2(\mu + \tilde{c}_{\mu,d})\Delta t_i \|\nabla\tilde{\phi}\|_0^2, \tag{4.26}$$

where $\Delta t_i \leq 1/2\tilde{c}_{\mu,d}$ was used to get

$$\|\widetilde{\nabla\phi}\|_0^2 \leq \frac{1}{1 - \tilde{c}_{\mu,d}\Delta t_i} \|\nabla\tilde{\phi}\| \leq 2\|\nabla\tilde{\phi}\|.$$

From (4.24)–(4.26), we recover

$$\left| \|\phi\|_1^2 - \|\tilde{\phi}\|_1^2 \right| \leq 2(\mu + \tilde{c}_{\mu,d})\Delta t_i \|\nabla\tilde{\phi}\|_0^2 + \tilde{c}_{\mu,d}\Delta t_i \|\tilde{\phi}\|_0^2 \leq 2(\mu + \tilde{c}_{\mu,d})\Delta t_i \|\tilde{\phi}\|_1^2.$$

This provides the desired result, and concludes the proof by taking $C_{\mu,d} = 2(\mu + \tilde{c}_{\mu,d})$. □

4.2.2 The shifted characteristic derivative

For static finite element spaces, the characteristic derivative is simply the derivative with respect to time, as $x_t \equiv 0$. Consequently, it follows that shifting the characteristic derivative of a moving finite element function results in the time derivative of the shifted finite element function. This notion is summarized by the following lemma, which is a result concerning the commutativity of the characteristic derivative and the shift operator.

Lemma 3. *Let $u \in \mathcal{V}$. Then, it holds for all $(\tilde{x}, t)$ in $\mathcal{F}$ that*

$$(\widetilde{\partial_\tau u})(\tilde{x}, t) = \frac{d}{dt}\tilde{u}(\tilde{x}, t).$$

Proof. This is shown directly from the definitions of the characteristic derivative (3.6) and shift operation (4.14). Let $x(t)$ denote the characteristic trajectory starting at $\tilde{x} = x(t_{i-1^+})$. Hence,

$$\begin{aligned}(\widetilde{\partial_\tau u})(\tilde{x}, t) = (\widetilde{\partial_\tau u})(x(t_{i-1^+}), t) &= (\partial_\tau u)(x(t), t) \\ &= \frac{d}{dt}\big[u(x(t), t)\big] = \frac{d}{dt}\big(\tilde{u}(x(t_{i-1^+}), t) = \frac{d}{dt}\tilde{u}(\tilde{x}, t).\end{aligned}$$

□

4.3 Approximation properties of the moving finite element space

In this section, the finite element interpolant is defined and used to find bounds for the best finite element approximation for a given function with domain $\mathcal{F}$. Namely, given a function u on $\mathcal{F}$ that possesses sufficient regularity and a shape regular finite element mesh, it is proven that

$$\inf_{\chi \in \mathcal{V}_h^p} |\!|\!| u - \chi |\!|\!| \leq C(\Delta x^p + \Delta t^p)|u|_*,$$

where the semi-norm, $|\cdot|_*$, is defined below. In effect, this shows that the order of a finite element space determines the convergence rate of the best finite element approximation with respect to mesh refinement.

4.3.1 Defining the finite element interpolant

For an element e in the mesh partition $\mathcal{T}_{h,i}^p$ and time t in $(t_{i-1}, t_i]$, let $\mathcal{I}_{e(t)}^p$ denote the spatial element interpolant on $e(t)$ with nodes at the degrees of freedom. The interpolant of a function v on $e(t)$ is then given by

$$\mathcal{I}_{e(t)}^p v(x) \equiv \sum_{k=0}^{s_{d,p}} \hat{\sigma}_k(x_e^{-1}(x,t)) v(x_k),$$

where $\hat{\sigma}_k$ represents the basis function corresponding to the k^{th} degree of freedom, $\hat{x}_k$, on the reference element, with $k = 0, \ldots, s_{d,p}$. Using these element interpolants, the finite element interpolant for $\mathcal{V}_h^p(t)$ is given by

$$\mathcal{I}_{\mathcal{V}_h(t)}^p v(x) \equiv \sum_{e \in \mathcal{T}_{h,i}^p} \mathcal{I}_{e(t)}^p v(x)\big|_{e(t)},$$

for $v : \Omega \to I\!R$.

Due to the potential discontinuities between the time partitions, we must be careful in defining the space-time interpolant as the interpolant must satisfy the jump orthogonality condition,

$$\big([\mathcal{I}_h^p v(t_i)], \chi\big) = 0,$$

for all χ in $\mathcal{V}_h^p(t_{i^+})$. Define the $\mathcal{L}_2$-projection $\mathcal{P}_i : \mathcal{L}_2(\Omega) \to \mathcal{V}_h^p(t_{i^+})$ such that

$$\big(\mathcal{P}_i v(t_i), \chi\big) = (v(t_{i^-}), \chi),$$

for all χ in $\mathcal{V}_h^p(t_{i^+})$, where $v \in \mathcal{L}_2(\Omega)$. Then for v on $\mathcal{F}$, we use the spatial interpolants, the collocation nodes of the quadrature rule described in (4.9), and the projection, $\mathcal{P}_i$, to define the space-time finite element interpolant. Let $\beta_{i,j}(t)$ represent the time basis function corresponding to collocation node $t_{i,j}$. Then, the finite element interpolant is defined as

$$\mathcal{I}_h^p v(x,t) \equiv \sum_{i=1}^{m} \left\{ \beta_{i,0}(t)\mathcal{P}_{i-1}\big[\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^-})} v(x,t_{i-1})\big] + \sum_{j=1}^{p} \beta_{i,j}(t)\mathcal{I}^p_{\mathcal{V}_h(t_{i,j})} v(x,t_{i,j}) \right\}.$$

While the projection at the discrete time partition nodes has no effect on functions that are independent of spatial variables, we incorporate it into a pseudo-interpolant over the time partition for convenience:

$$\mathcal{K}^p_{\mathcal{Q},i} v(t) = \beta_{i,0}(t)\mathcal{P}_{i-1} v(t_{i-1^-}) + \sum_{j=1}^{p} \beta_{i,j}(t) v(t_{i,j}),$$

whereas the true interpolant over the time partition, which does not use the projection operator, is denoted by

$$\mathcal{I}^p_{\mathcal{Q},i} v(t) = \sum_{j=0}^{p} \beta_{i,j}(t) v(t_{i,j}).$$

Note that the difference between the pseudo-interpolant and the true interpolant is limited to the first time basis function on the time partition:

$$(\mathcal{I}^p_{\mathcal{Q},i} - \mathcal{K}^p_{\mathcal{Q},i}) v(t) = \beta_{i,0}(t)\left[\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})} v(t_{i-1}) - \mathcal{P}_{i-1}\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})} v(t_{i-1})\right]. \tag{4.27}$$

Thus, the finite element interpolant can be represented by

$$\mathcal{I}_h^p v(x,t) = \sum_{i=1}^{m} \mathcal{K}^p_{\mathcal{Q},i}\big(\mathcal{I}^p_{\mathcal{V}_h(t)} v(x,t)\big)\Big|_{[t_{i-1},t_i]}.$$

It is the goal of this section to bound $\left|\!\left|\!\left| u - \mathcal{I}_h^p u \right|\!\right|\!\right|$, where the energy semi-norm is defined in (4.13). Note for moving finite elements, the time and space interpolants do not commute; we have the interpolation composition $\mathcal{K}^p_{\mathcal{Q},i}\left(\mathcal{I}^p_{\mathcal{V}_h(t)} v\right)(x,t) \in \mathcal{V}^p_{h,i}$,

whereas $\mathcal{I}^p_{\mathcal{V}_h(\tilde{t})}\left(\mathcal{K}^p_{\mathcal{Q},i}v\right)(x,t)$ is in the non-moving tensor space $\mathcal{V}^p_h(\tilde{t})\otimes(t_{i-1},t_i]$ for all t in the time partition.

Since the interpolants do not commute, we employ the shift functions to remove the time dependence of the spatial interpolant. From (4.16), the shift of the spatial interpolant satisfies

$$\widetilde{\left(\mathcal{I}^p_{\mathcal{V}_h(t)}v\right)}(\tilde{x},t)=\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\tilde{v}(\tilde{x},t)\in\mathcal{V}^p_h(t_{i-1^+})$$

for $t_{i-1}\le t\le t_i$, which gives

$$\begin{aligned}\widetilde{\left(\mathcal{I}^p_h v\right)}(\tilde{x},t)=\mathcal{K}^p_{\mathcal{Q},i}\widetilde{\left(\mathcal{I}^p_{\mathcal{V}_h(t)}v\right)}(\tilde{x},t)&=\mathcal{K}^p_{\mathcal{Q},i}\left(\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\tilde{v}\right)(\tilde{x},t)\\&=\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\left(\mathcal{K}^p_{\mathcal{Q},i}\tilde{v}\right)(\tilde{x},t),\quad(4.28)\end{aligned}$$

where we use the fact that the shifted spatial interpolation operator $\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}$ does not depend on time and, accordingly, commutes with the time interpolation operator.

4.3.2 Interpolation error bounds for the finite element spaces

The interpolation error bounds proven in Brenner and Scott [7] are used in our interpolation error analysis to provide standard results for the finite element spaces $\mathcal{V}^p_h(t)$ on Ω. Constraints (3.10) and (3.11) are used to bound the interpolation error, $(1-\mathcal{I}^p_{\mathcal{V}_h(t)})u$ for all t in $(0,T]$, in lemma 4 below.

Lemma 4. *Suppose $\mathcal{V}^p_h(t)$ corresponds to a mesh such that (3.10) and (3.11) are satisfied for some $1\ge\Delta x>0$, $\delta>0$, and some t in $(t_{i-1},t_i]$. Then for $p\ge1$, there exists a positive constant $C_{\delta,d,p}$ such that, for all u in $\mathcal{H}^{p+1}(\Omega)$,*

$$\left\|u-\mathcal{I}^p_{\mathcal{V}_h(t)}u\right\|_k\le C_{\delta,d,p}\Delta x^{p-k+1}|u|_{p+1}.$$

Proof. As described in section 3.2, the finite element space $\mathcal{V}^p_h(t)$ is constructed using affine equivalent elements and the shape regularity constraints (3.10) and (3.11) demonstrate that the element interpolation operators $\mathcal{I}^p_{e(t)}$ are uniformly bounded for all $e(t)$ in $\mathcal{T}^p_h(t)$, c.f. Brenner and Scott [7]. Since we are using Lagrange basis

functions on the reference element to define the finite element functions, we need $2(p+1) > d$. This is satisfied by $p \geq 1$ for $d = 1, 2, 3$, and so the spatial interpolant $\mathcal{I}^p_{\mathcal{V}_h(t)}$ and the finite element space $\mathcal{V}^p_h(t)$ satisfy the hypothesis of theorem 4.4.20 in [7], which implies the desired result. □

Finite element interpolation error

As a result of the shape regularity estimates (3.10)–(3.12), (3.14), and lemma 4, the approximation error of the interpolant can be bounded in the energy semi-norm defined in (4.13). Define the non-negative mesh dependent functional

$$\mathcal{E}_{(\mathcal{Q},\tau,p+1)}(u) \equiv \Bigg\{ \Delta x^2 \max_{0<t\leq T} |u(t)|^2_{p+1} + \mathcal{Q}\left(|u(\cdot)|^2_{p+1} + \Delta x^2 |u_\tau(\cdot)|^2_{p+1}\right) + \sum_{i=1}^{m} \Delta t_i |u(t_{i-1})|^2_{p+1} + \int_0^T \left(\|\partial^{p+1}_\tau u(t)\|^2_0 + \Delta x^{2(p+1)} |\partial^{p+1}_\tau u(t)|^2_{p+1} \right) dt \Bigg\}^{1/2}. \quad (4.29)$$

Note that $\mathcal{E}_{(\mathcal{Q},\tau,p+1)}(\chi) = 0$ for any χ in $\mathcal{V}^p_h$.

For a given mesh and quadrature rule, the functional $\mathcal{E}_{(\mathcal{Q},\tau,p+1)}$ acts as a semi-norm on $\mathcal{V}$; however, this quantity has terms that depend on the maximum element diameter of the spatial meshes $\mathcal{V}^p_h(t)$ that are cross terms arising from the tensor product elements. (Recall that $\Delta x \leq 1$ can alleviate this dependence.) Note that this quantity also depends on the mesh motion, due to the characteristic derivative term. In convection dominated problems, aligning the mesh with the convection velocity can keep u_τ small. This demonstrates that the quantity $\mathcal{E}_{(\mathcal{Q},\tau,p+1)}$ can be reduced by following characteristics of the differential equation.

Furthermore, the time discretization is a necessary dependence that cannot be avoided due to the method of lines approach in the proposed finite element method. The definition of this functional shows that when u has large spatial derivatives of order p or the spatial derivative of u_τ is large, small time steps, Δt_i, should be chosen to get a smaller quantity, and larger time steps should be chosen when the spatial derivatives of u and u_τ are smaller. This is consistent with previous results regarding adaptive time-stepping methods [44], and is revisited in chapter 7.

Lemma 5. *Suppose $\mathcal{V}_h^p$ is a finite element space with a non-degenerate mesh that satisfies (3.10)–(3.12) and (3.14) with $\Delta t \leq 1/2\tilde{c}_{\mu,d}$ and $\Delta x/\Delta t_i \leq \gamma$, for some positive constant γ, $i = 1, \ldots, m$. Then, the error of the interpolant using the collocation nodes of quadrature rule $\mathcal{Q}$ is bounded by*

$$\left|\!\left|\!\left| u - \mathcal{I}_h^p u \right|\!\right|\!\right|_{(\mathcal{V}_h^p, \mathcal{Q})} \leq C(\Delta x^p + \Delta t^p)\mathcal{E}_{(\mathcal{Q},\tau,p+1)}(u),$$

for some positive constant $C = C_{\hat{\mathcal{Q}},\delta,\gamma,\mu,d,p}$, where $\mathcal{E}_{(\mathcal{Q},\tau,p+1)}$ is defined in (4.29).

To comment on the constraint $\Delta x/\Delta t_i \leq \gamma$, this requirement is a necessary bound to control the effect of the projection error at the beginning of each time partition. The quantity $\Delta x/\Delta t_i$ naturally helps bound this projection error, as it improves upon spatial refinement (small Δx) and prefers longer time steps (large Δt_i), which counteracts using many time steps, which implies many $\mathcal{L}_2(\Omega)$-projections at the mesh discontinuities. If the meshes at the time partition align quite closely, the constant γ plays a less significant role, as can be seen in the proof.

Proof. This proof uses the definition of the energy semi-norm (4.13) to split the approximation error into several terms that can be bounded independently. We have

$$\begin{aligned}\left|\!\left|\!\left| u - \mathcal{I}_h^p u \right|\!\right|\!\right|^2_{(\mathcal{V}_h^p, \mathcal{Q})} = \max_{\substack{1\leq i\leq m \\ 1\leq j\leq p}} \left\| \big(u - \mathcal{I}_h^p u\big)(t_{i,j}) \right\|_0^2 \\ + \mathcal{Q}\left(\left\| \partial_\tau \big(u - \mathcal{I}_h^p u\big)(\cdot) \right\|^2_{(-1,\mathcal{V}_h^p(\cdot))} + \left\| \big(u - \mathcal{I}_h^p u\big)(\cdot) \right\|_1^2 \right). \quad (4.30)\end{aligned}$$

For any collocation node $t_{i,j}$ with $j > 0$, the shape regularity assumptions (3.10)–(3.11) show that $\mathcal{I}^p_{\mathcal{V}_h(t_{i,j})}$ satisfies the hypothesis of lemma 4 to provide

$$\left\| \big(u - \mathcal{I}_h^p u\big)(t_{i,j}) \right\|_k \leq C_{\delta,d,p} \Delta x^{p+1-k} |u(t_{i,j})|_{p+1}, \quad (4.31)$$

for $k = 0$ and 1.

Bounding the terms involving the characteristic derivatives is more complicated due to the moving nodes in the mesh. To circumvent the complications that arise from the characteristic derivative, we use the results proven in section 4.2.1

to shift the functions onto a static mesh. First and foremost, we remove the time dependence of the negative norm, $\|\cdot\|_{(-1,\mathcal{V}_h^p(t))}$, by noting

$$\big\|\partial_\tau\big(u-\mathcal{I}_h^p u\big)(t_{i,j})\big\|_{(-1,\mathcal{V}_h^p(t_{i,j}))} \le \big\|\partial_\tau\big(u-\mathcal{I}_h^p u\big)(t_{i,j})\big\|_0. \tag{4.32}$$

From (4.28), the time derivative of the shifted interpolant satisfies

$$\frac{d}{dt}\big(\widetilde{\mathcal{I}_h^p u}\big)(\tilde{x},t) = \mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\Big(\frac{d}{dt}\mathcal{K}^p_{\mathcal{Q},i}\tilde{u}\Big)(\tilde{x},t),$$

which we combine with the shift lemma 2, lemma 3, and (4.27) to show

$$\begin{aligned}
\big\|\partial_\tau\big(u-\mathcal{I}_h^p u\big)(t_{i,j})\big\|_{(-1,\mathcal{V}_h^p(t_{i,j}))} &\le (1+C_{\mu,d}\Delta t_i)\Big\|\tilde{u}_t(t_{i,j})-\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\Big(\frac{d}{dt}\mathcal{K}^p_{\mathcal{Q},i}\tilde{u}\Big)(t_{i,j})\Big\|_0 \\
&\le (1+C_{\mu,d}\Delta t_i)\Big\{\Big\|\tilde{u}_t(t_{i,j})-\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\tilde{u}_t(t_{i,j})\Big\|_0 \\
&\qquad + \Big\|\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\Big(\tilde{u}_t(t_{i,j})-\frac{d}{dt}\mathcal{K}^p_{\mathcal{Q},i}\tilde{u}(t_{i,j})\Big)\Big\|_0\Big\} \\
&\le (1+C_{\mu,d}\Delta t_i)\Big\{\Big\|\tilde{u}_t(t_{i,j})-\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\tilde{u}_t(t_{i,j})\Big\|_0+\Big\|\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\Big(\tilde{u}_t(t_{i,j})-\frac{d}{dt}\mathcal{I}^p_{\mathcal{Q},i}\tilde{u}(t_{i,j})\Big)\Big\|_0 \\
&\qquad + \big|\beta'_{i,0}(t_{i,j})\big|\big\|\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\tilde{u}(t_{i-1}) - \mathcal{P}_{i-1}\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^-})}\tilde{u}(t_{i-1})\big\|_0\Big\}.
\end{aligned} \tag{4.33}$$

Lemmas 2 and 4 give

$$\begin{aligned}
\Big\|\tilde{u}_t(t_{i,j}) - \mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\tilde{u}_t(t_{i,j})\Big\|_0 &\le C_{\delta,d,p}\Delta x^{p+1}|\tilde{u}_t|_{p+1} \\
&\le (1+C_{\mu,d}\Delta t_i)C_{\delta,d,p}\Delta x^{p+1}|u_\tau|_{p+1}
\end{aligned} \tag{4.34}$$

and, since $\tilde{u}(t_{i-1}) = u(t_{i-1})$,

$$\begin{aligned}
&\Big\|\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}\tilde{u}(t_{i-1}) - \mathcal{P}_{i-1}\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^-})}\tilde{u}(t_{i-1})\Big\|_0 \\
&\qquad = \big\|\mathcal{P}_{i-1}\big(\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}u(t_{i-1}) - \mathcal{I}^p_{\mathcal{V}_h(t_{i-1^-})}u(t_{i-1})\big)\big\|_0 \\
&\qquad \le \big\|u(t_{i-1}) - \mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}u(t_{i-1})\big\|_0 + \big\|u(t_{i-1}) - \mathcal{I}^p_{\mathcal{V}_h(t_{i-1^-})}u(t_{i-1})\big\|_0 \\
&\qquad \le 2C_{\delta,d,p}\Delta x^{p+1}|u(t_{i-1})|_{p+1}.
\end{aligned} \tag{4.35}$$

To complete the proof, it is only needed to bound the second term on the right in (4.33). Denoting $v = \tilde{u}_t(t_{i,j}) - \frac{d}{dt}\mathcal{I}^p_{\mathcal{Q},i}\tilde{u}(t_{i,j})$, we again invoke lemma 4 to bound

$$\big\|\mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}v\big\|_0 \le \|v\|_0 + \big\|v - \mathcal{I}^p_{\mathcal{V}_h(t_{i-1^+})}v\big\|_0 \le \|v\|_0 + C_{\delta,d,p}\Delta x^{p+1}|v|_{p+1}. \tag{4.36}$$

Using the Peano kernel theorem [66] and $\Delta t_i \leq 1/2\tilde{c}_{\mu,d}$, we have

$$\begin{aligned}\left\|\tilde{u}_t(t_{i,j}) - \frac{d}{dt}\mathcal{I}^p_{\mathcal{Q},i}\tilde{u}(t_{i,j})\right\|_0 &\leq C_{\hat{\mathcal{Q}},p}\Delta t^p \left\| \left(\int_{t_{i-1}}^{t_i} \left[\left(\frac{d}{dt}\right)^{p+1} \tilde{u}(t) \right]^2 dt \right)^{1/2} \right\|_0 \\ &\leq C_{\hat{\mathcal{Q}},\mu,d,p}\Delta t^p \left\{ \int_{t_{i-1}}^{t_i} \|\partial_\tau^{p+1} u(t)\|_0^2 \, dt \right\}^{1/2} \end{aligned} \tag{4.37}$$

and

$$\begin{aligned}&\left|\tilde{u}_t(t_{i,j}) - \frac{d}{dt}\mathcal{I}^p_{\mathcal{Q},i}\tilde{u}(t_{i,j})\right|_{p+1} \\ &\qquad = \left\| \frac{d}{dt}\Big[\sum_{|\alpha|=p+1} D_x^\alpha \tilde{u}(t_{i,j}) \Big] - \frac{d}{dt}\mathcal{I}^p_{\mathcal{Q},i}\Big[\sum_{|\alpha|=p+1} D_x^\alpha \tilde{u}(t_{i,j}) \Big] \right\|_0 \\ &\qquad \leq C_{\hat{\mathcal{Q}},p}\Delta t^p \left\| \left(\int_{t_{i-1}}^{t_i} \left[\left(\frac{d}{dt}\right)^{p+1} \sum_{|\alpha|=p+1} D_x^\alpha \tilde{u}(t) \right]^2 dt \right)^{1/2} \right\|_0 \\ &\qquad \leq C'_{\hat{\mathcal{Q}},p}\Delta t^p \left\| \left(\int_{t_{i-1}}^{t_i} \sum_{|\alpha|=p+1} \left[\left(\frac{d}{dt}\right)^{p+1} D_x^\alpha \tilde{u}(t) \right]^2 dt \right)^{1/2} \right\|_0 \\ &\qquad \leq C_{\hat{\mathcal{Q}},\mu,d,p}\Delta t^p \left\{ \int_{t_{i-1}}^{t_i} |\partial_\tau^{p+1} u(t)|_{p+1}^2 \, dt \right\}^{1/2}, \end{aligned} \tag{4.38}$$

where $C_{\hat{\mathcal{Q}},p}$ is a bounding constant for the function and kernel, $\hat{K}_{\hat{\mathcal{Q}},p}(t,\zeta) = \int_0^1 \int_0^1 (t-\zeta)_+^p - \mathcal{I}^p_{\hat{\mathcal{Q}}}[(\cdot-\zeta)_+^p] \, d\zeta \, dt$, which is attained analogously to (4.10). Combining (4.32)–(4.38) gives the bound for the time discretization error:

$$\begin{aligned}\left\|\partial_\tau\big(u - \mathcal{I}_h^p u\big)(t_{i,j})\right\|_{(-1,\mathcal{V}_h^p(t_{i,j}))} &\leq C_{\hat{\mathcal{Q}},\delta,\gamma,\mu,d,p}\Bigg\{ \Delta x^{p+1}|u_\tau|_{p+1} \\ &+ \Delta x^{p+1}\Delta t_i^{-1}|u(t_{i-1})|_{p+1} + \Delta x^{p+1}\Delta t^p \left(\int_{t_{i-1}}^{t_i} |\partial_\tau^{p+1} u(t)|_{p+1}^2 \, dt \right)^{1/2} \\ &+ \Delta t^p \left(\int_{t_{i-1}}^{t_i} \|\partial_\tau^{p+1} u(t)\|_0^2 \, dt \right)^{1/2} \Bigg\}. \end{aligned} \tag{4.39}$$

Due to the coefficient $|\beta'_{i,0}(t_{i,j})| = \Delta t_i^{-1}|\hat{\beta}'_0(\hat{t})|$ in (4.33), we have a term multiplied by $\Delta x^{p+1}/\Delta t_i \leq \gamma \Delta x^p$, which follows from the space-time quasi-uniformity.

From (4.30), (4.31), and (4.39), we obtain

$$\begin{aligned}\|\!|u - \mathcal{I}_h^p u|\!\|^2 \leq C^2_{\hat{\mathcal{Q}},\delta,\gamma,\mu,d,p}(\Delta x^p + \Delta t^p)^2 \times \\ \left\{ \left[\Delta x \max_{0<t\leq T} |u(t)|_{p+1}\right]^2 + \mathcal{Q}\left(|u(\cdot)|^2_{p+1} + \left[\Delta x |u_\tau(\cdot)|_{p+1}\right]^2\right) \right. \\ \left. + \sum_{i=1}^{m} \Delta t_i |u(t_{i-1})|^2_{p+1} + \int_0^T \|\partial_\tau^{p+1} u(t)\|_0^2 \, dt + \left[\Delta x \int_0^T \|\partial_\tau^{p+1} u(t)\|_0^2 \, dt\right]^2 \right\},\end{aligned}$$

as desired. □

As an immediate consequence of this lemma, we have the following theorem regarding the best-approximation error for $\mathcal{V}_h^p$.

Theorem 1. *Suppose $\mathcal{V}_h^p$ is a finite element space with a non-degenerate mesh that satisfies (3.10)–(3.12) and (3.14) with $\Delta t \leq 1/2\tilde{c}_{\mu,d}$ and $\Delta x/\Delta t_i \leq \gamma$, for $i = 1, \ldots, m$. Then, the best approximation error of the finite element space using the collocation nodes of a composite quadrature rule, $\mathcal{Q}$, is bounded by*

$$\inf_{\chi \in \mathcal{V}_h^p} |\!|\!| u - \chi |\!|\!|_{(\mathcal{V}_h^p, \mathcal{Q})} \leq C(\Delta x^p + \Delta t^p) \mathcal{E}_{(\mathcal{Q},\tau,p+1)}(u),$$

for some positive constant $C = C_{\hat{\mathcal{Q}},\delta,\gamma,\mu,d,p}$, where $\mathcal{E}_{(\mathcal{Q},\tau,p+1)}$ is defined in (4.29).

Proof. The proof is an immediate result because $\mathcal{I}_h^p u \in \mathcal{V}_h^p$. Thus, we see

$$\inf_{\chi \in \mathcal{V}_h^p} |\!|\!| u - \chi |\!|\!|_{(\mathcal{V}_h^p, \mathcal{Q})} \leq \left|\!\left|\!\left| u - \mathcal{I}_h^p u \right|\!\right|\!\right|_{(\mathcal{V}_h^p, \mathcal{Q})} \leq C(\Delta x^p + \Delta t^p) \mathcal{E}_{(\mathcal{Q},\tau,p+1)}(u).$$

□

Lemma 5 and theorem 1 are results that describe approximation properties of the finite element space. They are independent of the differential equation and the finite element methods discussed in this dissertation. However, in providing bounds for the error of the finite element solution, we separate the error introduced by the finite element method and the finite element discretization. Thus, proving a quasi-optimal error estimate of the finite element solution, as discussed in the introduction, and using theorem 1 shows that

$$|\!|\!| u - u_h |\!|\!| \leq C \inf_{\chi \in \mathcal{V}_h^p} |\!|\!| u - \chi |\!|\!| \leq C'(\Delta x^p + \Delta t^p) \mathcal{E}_{(\mathcal{Q},\tau,p+1)}(u). \tag{4.40}$$

The results in this section form the basis for the analysis in the following chapters. Insofar, the differential equation and the finite element mesh have not been considered together, except for some brief discussion regarding choices for mesh motion and adaptive time stepping. We now discuss and analyze two finite element methods that employ these space-time moving finite element spaces for discretizing the differential equation and solving this discrete formulation of the problem.

Chapter 4, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 5

A Space-Time Moving Finite Element Method

In this chapter, a space-time moving finite element method is described. An important feature of this method is that the mesh motion is not prescribed by the method and many strategies for determining the mesh motion of the finite element space fall within the framework of this method. The efficacy of moving finite elements is largely predicated on how well the mesh motion suits the given differential equation. Accordingly, mesh selection is a critical component of finding an accurate numerical solution using moving finite elements. Many techniques have been proposed and tested for determining the mesh motion for a wide range of differential equations, though as far as we have seen, there is no single strategy that works well for all types of problems.

By not fixing the mesh motion, we provide a general error analysis that applies to a broad class of moving finite element methods. For many moving finite element methods, the finite element solution is computed by residual minimization; this approach is effective for finding the finite element solution and mesh motion simultaneously via an extended set of constraint equations, beyond those of the usual Galerkin approach. The mesh motion is often determined by additional equations relating the motion of the spatial nodes to the computed solution, its gradient, or some penalty terms that enforce shape regularity (and, consequently, ensure well-posedness of the moving finite element method) ([57],[55]). Another scheme is the Gradient-Weighted Moving Finite Element method, where the integrals of the variational equation are weighted by $(1 + |u|_1^2)^{1/2}$, which leads to a nonlinear system of equations that must be solved to find the finite element solution ([28],[29]). Other methods correspond to solving some modified version of the residual minimization problem to ensure some other desirable properties of the mesh motion, such as stabilizing meshes for steady state problems ([**?**],[56]). Many of these methods also lead to a nonlinear set of equations, as in the case of gradient weighted moving finite element methods.

In an effort to avoid nonlinear equations, we only consider methods where the motion of the spatial nodes does not implicitly depend on the solution to the PDE. For our finite element method, the mesh motion can be determined by the method of characteristics, by satisfying some conservation law [13], by previously

computed values of the finite element solution, by some monitor function like an a posteriori error estimators [65], or by some iterative method [12]. For simplicity, we assume that the mesh motion is pre-determined so that we may use a standard Galerkin approach and avoid solving nonlinear systems to find the finite element solution. To maintain a level of generality, the only assumptions that the mesh must satisfy in this analysis are the shape regularity constraints (3.10)–(3.12), (3.14), and

$$\|b(t) - x_t(t)\|_\infty < \infty,$$

for $0 < t \leq T$. These relaxed constraints should be satisfied for any reasonable choice of mesh. Notice that both standard space-time tensor elements (when $x_t \equiv 0$) and the method of characteristics (when $x_t = b$) can satisfy these mesh constraints, assuming that b does not lead to degenerate finite element spaces on $\mathcal{F}$.

In this chapter, a space-time moving finite element formulation of the PDE is given, followed by a brief discussion of a discrete Galerkin orthogonality property of the proposed method. Then, the effects of the method of lines approach are analyzed along with certain constraints required to attain a symmetric error bound. The well-posedness of the method is proven by analyzing the linear system that results from the discrete formulation, along with two Grönwall-like arguments that are used to prove a quasi-optimal, symmetric a priori error estimate. To conclude the chapter, the results in chapter 4 are used to prove convergence rates of the finite element solution, with respect to mesh refinement.

5.1 The finite element formulation

The space-time finite element formulation that is used to determine the finite element solution is a discretization of formulation 3. As usual with finite element methods, the space of trial and test functions is restricted to the finite element space. The result is a system of ordinary differential equations, where each equation describes the evolution of the solution in time along the trajectory of a spatial node. This method is, therefore, a method of lines. In the case of

moving meshes, this generalizes to a "method of trajectories," which encompasses the method of lines, when $x_t = 0$, and the method of characteristics, when $x_t = b$. Hence, we have a semi-discrete formulation of the differential equation, as the time dimension remains undiscretized. To discretize this problem in time, we only require the constraint (2.11) to hold at *discrete time steps* that correspond to some composite quadrature rule to numerically solve the resulting system of ordinary differential equations. Consequently, the fully discrete formulation depends on the choice of mesh motion, x_t, and the quadrature rule, $\mathcal{Q}$, that is used to determine the location of the discrete collocation nodes. As before, the bilinear form corresponding to the spatial components of the differential equation depends on the mesh motion and is given by

$$\mathcal{A}_\tau(u, \chi; t) \equiv$$
$$\int_\Omega a(x,t)\nabla u(x,t)\cdot\nabla\chi(x) + (b(x,t) - x_t(t))\cdot\nabla u(x,t)\ \chi(x) + c(x,t)u(x,t)\chi(x)\ dx.$$

Using this notation, the fully discretized moving finite element formulation of the convection-diffusion-reaction equation is given in the following.

Formulation 4. *Let a, b, c, and f be smooth and bounded functions on $\mathcal{F}$ satisfying $a \geq \bar{a} > 0$ and $c \geq \bar{c} \geq 0$, and let $g(t)$ be in $\mathcal{L}_2(\partial\Omega)$ for $0 < t \leq T$. Given the mesh velocity, x_t, and collocation nodes, $\{t_{i,j}\}$, find u_h in $\mathcal{V}_h^p$ such that for each $t_{i,j}$ and all χ in $\mathcal{V}_h^p(t_{i,j})$, the finite element solution satisfies*

$$\big(\partial_\tau u_h(t_{i,j}), \chi\big) + \mathcal{A}_\tau\big(u_h, \chi; t_{i,j}\big) = \big(f(t_{i,j}), \chi\big) + \big\langle g(t_{i,j}), \chi\big\rangle \tag{5.1}$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, p$, and when $t = 0$,

$$\big(u_h(\cdot, 0), \chi\big) = \big(u_0, \chi\big).$$

Although not explicitly stated in the finite element formulation, the computed solution at the beginning of each time step is the $\mathcal{L}_2$-projection of the solution at the end of the previous time partition:

$$(u_h(t_{i^+}), \chi) = (u_h(t_{i^-}), \chi), \tag{5.2}$$

for all χ in $\mathcal{V}_h^p(t_{i^+})$ and $i = 1, \ldots, m$. This is a requirement for u_h to belong to $\mathcal{V}_h^p$, as stated in (3.7). In truth, this is necessary only to simplify the error analysis. As a practical note, interpolation is a faster option for transferring the computed solution across the mesh discontinuities between time partitions. The effects of using interpolation instead of projection onto a new mesh is discussed in the numerical experiments section 7.4.

5.2 A discrete Galerkin orthogonality

Inherent to formulation 4, the error of the finite element solution will satisfy a *Galerkin orthogonality* condition at the discrete collocation nodes; this is an immediate result of subtracting (6.5) from (2.11). Hence, at each collocation node in the time discretization,

$$\big(\partial_\tau(u - u_h)(t_{i,j}), \chi\big) + \mathcal{A}_\tau\big(u - u_h, \chi; t_{i,j}\big) = 0, \tag{5.3}$$

for all χ in $\mathcal{V}_h^p(t_{i,j})$, when $1 \le i \le m$ and $1 \le j \le p$. This is a standard property of finite element methods for elliptic equations; however, this Galerkin orthogonality property does not necessarily hold throughout the entire time domain, as it is only imposed at the discrete collocation nodes. The degree to which this orthogonality holds between these time nodes depends on the placement of collocation nodes.

As discussed in section 4.1.3, the collocation nodes $\{t_{i,j}\}$ correspond to a composite quadrature rule, $\mathcal{Q}$. Using weights w_j from definition of the composite quadrature rule (4.9) in conjunction to (5.3), one recovers

$$\begin{aligned}\int_0^T \big(\partial_\tau(u-u_h)(t), \chi\big) + \mathcal{A}_\tau\big(u-u_h, \chi; t\big)\, dt &\approx \mathcal{Q}\Big(\big(\partial_\tau(u-u_h), \chi\big) + \mathcal{A}_\tau\big(u-u_h, \chi\big)\Big) \\ &= \sum_{i=1}^{m}\sum_{j=1}^{p} w_j \Delta t_i \big\{\big(\partial_\tau(u-u_h)(t_{i,j}), \chi\big) + \mathcal{A}_\tau\big(u-u_h, \chi; t_{i,j}\big)\big\} = 0, \end{aligned} \tag{5.4}$$

since the finite element space $\mathcal{V}_h^p(t_{i,j}) \subset \mathcal{H}^1(\Omega)$. Thus, a discrete Galerkin orthogonality property over the space-time domain is satisfied. By refining the time discretization (smaller Δt) or carefully choosing the collocation nodes (optimizing

$\{t_{i,j}\}$), the discrete Galerkin orthogonality better resembles a true Galerkin orthogonality on the space-time domain, and the rate of convergence to this orthogonality property is determined by the order of the quadrature rule.

5.2.1 A closer look at the time discretization

In formulation 3, the differential equation (2.11) is imposed for every time t in the domain $(0, T]$. Since the finite element solution only satisfies this requirement at the discrete time collocation nodes in the mesh, it is important to understand the effects on the error introduced by the relative position of the collocation nodes within a time partition. Any interpolatory quadrature rule with p collocation nodes is of order p, as mentioned in section 4.1.3, and we consequently assume an interpolatory quadrature rule.

The space-time variational formulation given by

$$\int_{t_{i-1}}^{t_i} (\partial_\tau u_h(t), \chi(t)) + \mathcal{A}_\tau(u_h, \chi; t)\ dt = \int_{t_{i-1}}^{t_i} (f(t), \chi(t)) + \langle g(t), \chi(t)\rangle\ dt$$

has a characteristic derivative term that is a polynomial of degree $4p-1$. This is because $\partial_\tau u_h(t)$ is, in general, a polynomial of degree $2p-1$ with respect to t, since it is the composition of the static polynomial $\frac{d}{dt}\tilde{u}_h$, of degree $p-1$, and the degree p polynomial characteristic trajectory $x(t)$. Similarly, the function $\chi(t) = \tilde{\chi} \circ x(t)$ is of degree $2p$ with respect to t. By shifting this term and using (4.18) of lemma 2,

$$\big(\partial_\tau u_h(t), \chi(t)\big) = \Big(\frac{d}{dt}\tilde{u}_h(t), \tilde{\chi}(t)\Big) + \mathcal{O}(\|\Delta t_i \partial_\tau u_h\|_0 \|\chi\|_0),$$

the resulting time derivative term is of order $2p-1$ with respect to the time variable. Thus, to integrate this shifted term exactly, the collocation nodes must correspond to the nodes of Gaussian quadrature, defined by the roots of the p^{th} orthogonal polynomial; this is the only quadrature rule that has order of exactness $2p-1$ with only p nodes.

This does not restrict the only plausible choice of quadrature rule to be Gaussian, though. The error analysis provides a symmetric error bound as long as the quadrature rule satisfies a *non-truncating* condition, which we describe here.

Let $\hat{\mathcal{Q}}$ be a given quadrature rule over the unit interval, $[0,1]$. Then, the quadrature rule is *non-truncating* if it satisfies

$$\hat{\mathcal{Q}}\Big(\frac{d}{dt}(v^2)\Big) \geq \int_0^1 \frac{d}{dt}(v^2(t))\ dt = v^2(1) - v^2(0), \tag{5.5}$$

for all polynomials v of degree at most p with positive leading coefficients.

5.2.2 A one parameter family of non-truncating quadrature rules

Clearly, the Gaussian quadrature rule satisfies (5.5), as it is exact for all polynomials of degree at most $2p$, meaning $\hat{\mathcal{Q}}\big(\frac{d}{dt}(v^2)\big) = \int_0^1 \frac{d}{dt}(v^2(t))\,dt$ in the case of Gaussian quadrature. We describe a one parameter family of quadrature rules that are non-truncating for a given p. Consider quadrature rules that have degree of exactness $2p-2$. Then, for a monic polynomial $v(t) = t^p + v_{p-1}t^{p-1} + v_{p-2}t^{p-2} + \cdots + v_0$, it holds that

$$\int_0^1 \frac{d}{dt}(v^2(t))\ dt - \hat{\mathcal{Q}}\Big(\frac{d}{dt}(v^2)\Big) = 2p\left\{\int_0^1 t^{2p-1}\ dt - \hat{\mathcal{Q}}\big(t^{2p-1}\big)\right\} = 1 - 2p\sum_{j=1}^{p} w_j \hat{t}_j^{2p-1},$$

since $\hat{\mathcal{Q}}$ has degree of exactness $2p-2$. Thus, we need to show that

$$\sum_{j=1}^{p} w_j \hat{t}_j^{2p-1} \geq 1/2p. \tag{5.6}$$

From [62], it is true that any quadrature rule with degree of exactness $2p-2$ must have collocation nodes that coincide to the roots of a linear combination of the orthogonal polynomials of degree p and $p-1$ (these polynomials are orthogonal on $[0,1]$ with the unit weight function $w \equiv 1$). Namely, if $\{\pi_q\}$ represent these orthogonal polynomials, assumed to be normalized to be monic, then there exists some α such that

$$(t - \hat{t}_1)\cdots(t - \hat{t}_p) = \pi_p(t) + \alpha\pi_{p-1}(t). \tag{5.7}$$

Note that for $\alpha = 0$, we have Gaussian quadrature. Let $\alpha = \alpha_{\mathrm{GR}} < 0$ correspond to the right Gauss-Radau quadrature, defined by $\hat{t}_p = 1$. It is known that right Gauss-Radau quadrature rule has degree of exactness $2p-2$ and that (5.5) holds

with strict inequality [59], which implies that (5.6) holds with strict inequality when $\alpha = \alpha_{\mathrm{GR}}$. Thus, using Gauss or Gauss-Radau quadrature provides a non-truncating quadrature rule.

Let $\hat{t}_j(\alpha)$ denote the collocation nodes as continuously parametrized by α in (5.7). Then, the weights $w_j(\alpha)$ also depend continuously on α since

$$w_j(\alpha) = \int_0^1 \prod_{\substack{k=1 \\ k \neq j}}^{p} \frac{\hat{t} - \hat{t}_k(\alpha)}{\hat{t}_j(\alpha) - \hat{t}_k(\alpha)} \, d\hat{t}.$$

Note that $\hat{t}_j(\alpha) \neq \hat{t}_k(\alpha)$ for $j \neq k$, which keep the w_j well-defined and positive. Hence, the quadrature of t^{2p-1} is a quantity that is continuously parametrized by α when we assume order of exactness $2p-2$. The choice $\alpha = 0$ satisfies (5.6) with equality, since it corresponds to the higher order Gauss rule, whereas the choice $\alpha = \alpha_{\mathrm{GR}}$ satisfies this requirement with a strict inequality. Suppose there exists $\bar{\alpha}$ in the interval $(\alpha_{\mathrm{GR}}, 0)$ such that (5.6) does not hold; then, the intermediate value theorem states that there must be some other rule, beside the Gauss rule, that satisfies (5.6) with equality holding since the quadrature of t^{2p-1} is continuous with respect to the parameter α. If this were true, then there would exist another quadrature rule with p collocation nodes, distinct from the Gauss rule, with order of exactness $2p-1$. This is a contradiction since this property is known to be unique to the Gauss rule. As a result, if the quadrature rule has collocation nodes defined to be the roots of some polynomial as given in (5.7) with α in $[\alpha_{\mathrm{GR}}, 0]$, then it is non-truncating.

Examples of families of non-truncating quadrature rules

For the case where $p = 1$, using the decomposition (5.7), we find $t - \hat{t}_1 = (t - 1/2) + \alpha$, which implies $\alpha_{\mathrm{GR}} = -1/2$ since this gives $\hat{t}_1 = 1$. Thus, for α in $[-1/2, 0]$, we have a non-truncating quadrature rule; this corresponds to choosing the collocation node to be in the range $[1/2, 1]$, and an a priori error estimate has been proven in this case by Dupont and Mogultay [40].

Once the order $p \geq 1$, the parameterization of the quadrature rule and its collocation nodes becomes nonlinear and more complicated. For example, when

$p = 2$, we have

$$(t - \hat{t}_1)(t - \hat{t}_2) = (t^2 - t/7 - 17/42) + \alpha(t - 1/2)$$
$$= \left(t - \frac{\frac{1}{7} - \alpha - \sqrt{\alpha^2 + \frac{12}{7}\alpha + \frac{241}{147}}}{2} \right) \left(t - \frac{\frac{1}{7} - \alpha + \sqrt{\alpha^2 + \frac{12}{7}\alpha + \frac{241}{147}}}{2} \right),$$

which gives $\alpha_{\text{GR}} = -\frac{19}{21}$. Thus, we may choose

$$\hat{t}_1 = \frac{\frac{1}{7} - \alpha - \sqrt{\alpha^2 + \frac{12}{7}\alpha + \frac{241}{147}}}{2}$$

with

$$\hat{t}_2 = \frac{\frac{1}{7} - \alpha + \sqrt{\alpha^2 + \frac{12}{7}\alpha + \frac{241}{147}}}{2}$$

for $-\frac{19}{21} \leq \alpha \leq 0$. The non-truncating family for higher order quadrature rules can be found analogously to the $p = 1$ and $p = 2$ cases.

5.3 Well-posedness of the finite element formulation

Formulation 4 corresponds to a linear system that can be solved to compute the finite element solution, u_h, by using a basis expansion of the finite element functions. For finite element methods that use a method of lines approach for space-time problems, the linear system is derived by expanding the finite element solution by its spatial basis functions to get a semi-discrete linear system involving the vector representation of the finite element solution and its time derivative. Subsequently, the time domain is discretized so that the existence and uniqueness of a solution to the fully discrete formulation is established.

5.3.1 The semi-discrete linear system

Using the basis functions of $\mathcal{V}_h^p(t_{i,j})$, the finite element solution u_h of formulation 4 must satifsy

$$\big(\partial_\tau u_h(t_{i,j}), \sigma_\ell(t_{i,j})\big) + \mathcal{A}_\tau\big(u_h, \sigma_\ell; t_{i,j}\big) = \big(f(t_{i,j}), \sigma_\ell(t_{i,j})\big) + \big\langle g(t_{i,j}), \sigma_\ell(t_{i,j})\big\rangle \quad (5.8)$$

for all spatial basis functions $\sigma_\ell(t_{i,j})$, $1 \le \ell \le N_i$, corresponding to the degrees of freedom in the mesh, $x_k(t_{i,j})$, and for all $t_{i,j}$, $j = 1, \ldots, p$ and $i = 1, \ldots, p$, where N_i denotes the number of degrees of freedom on the mesh at time $t_{i,j}$. Expanding $\mathcal{V}_h^p(t_{i,j})$ by the spatial basis functions gives

$$\big(\partial_\tau u_h(t_{i,j}), \sigma_\ell(t_{i,j})\big) = \sum_{k=1}^{N_i} \partial_\tau u_h(x_k(t_{i,j}), t_{i,j})\big(\sigma_k(t_{i,j}), \sigma_\ell(t_{i,j})\big).$$

Define the *mass matrix* at time $t_{i,j}$ by the pairwise inner-products of the spatial basis functions,

$$\mathcal{M}(t_{i,j}) \equiv \Big[\big(\sigma_k(t_{i,j}), \sigma_\ell(t_{i,j})\big)\Big]_{(\ell,k)}$$

and the vector of basis coordinates for $\partial_\tau u_h(t_{i,j})$ by

$$U_\tau(t_{i,j}) \equiv \big[\partial_\tau u_h(x_k(t_{i,j}), t_{i,j})\big]_k,$$

for $1 \le k, \ell \le N_i$. Thus, if there are N_i degrees of freedom in the mesh at time $t = t_{i,j}$, then the mass matrix $\mathcal{M}(t_{i,j})$ is a $N_i \times N_i$ matrix and $U_\tau(t_{i,j})$ is a vector of length N_i. It is easy to see that the mass matrix, $\mathcal{M}(t)$, is positive definite. Let the vector $\vec{v}$ in $\mathbb{R}^{N_i}$ be the vector containing the basis coefficients of a nonzero finite element function $v(t_{i,j})$ in $\mathcal{V}_h^p(t_{i,j})$. Then, the positive definiteness follows from the identity $\vec{v}^{\,T}\mathcal{M}(t_{i,j})\vec{v} = \|v(t_{i,j})\|_0^2 > 0$. Note that the shape regularity assumptions improve the condition number of the mass matrix.

For the bilinear form $\mathcal{A}_\tau\big(u_h, \sigma_\ell; t_{i,j}\big)$, we expand $u_h(t_{i,j})$ to get

$$\mathcal{A}_\tau\big(\partial_\tau u_h(t_{i,j}), \sigma_\ell(t_{i,j}); t_{i,j}\big) = \sum_{k=1}^{N_i} u_h(x_k(t_{i,j}), t_{i,j})\mathcal{A}_\tau\big(\sigma_k(t_{i,j}), \sigma_\ell(t_{i,j}); t_{i,j}\big).$$

Similar to the characteristic derivative term, define the stiffness matrix at time $t_{i,j}$, now using the bilinear form,

$$\mathcal{S}(t_{i,j}) \equiv \Big[\mathcal{A}_\tau\big(\sigma_k(t_{i,j}), \sigma_\ell(t_{i,j}); t_{i,j}\big)\Big]_{(\ell,k)}$$

and the vector of basis coefficients for $u_h(t_{i,j})$,

$$U(t_{i,j}) \equiv \big[u_h(x_k(t_{i,j}), t_{i,j})\big]_k,$$

for $1 \leq k, \ell \leq N_i$. In the case when $b - x_t \equiv 0$, it holds true that the stiffness matrix is positive definite as well since $a > 0$ and $c \geq 0$, following from the same argument as for the mass matrix. However, the asymmetry brought on by the convection term can break this positive definite property when $|b - x_t|$ is sufficiently large. If the bilinear form is coercive, as discussed in subsection 4.1.2, this matrix is guaranteed to be nonsingular by (4.7). Methods that are designed from an Eulerian approach seek to improve this asymmetry by means of upwind differencing, streamline diffusion, the method of characteristics, or, as in our case, moving meshes. Of course, this list of methodologies is not exhaustive, but it covers some of the most common methods that are designed for non-symmetric stiffness matrices.

For the right side of equation (5.8), define the vectors

$$F(t_{i,j}) \equiv \big[(f(t_{i,j}), \sigma_\ell(t_{i,j}))\big]_\ell \quad \text{and} \quad G(t_{i,j}) \equiv \big[\langle g(t_{i,j}), \sigma_\ell(t_{i,j})\rangle\big]_\ell,$$

where $1 \leq \ell \leq N_i$. Then, the system of ODEs corresponding to equation (5.8) is given by

$$\mathcal{M}(t_{i,j})U_\tau(t_{i,j}) + \mathcal{S}(t_{i,j})U(t_{i,j}) = F(t_{i,j}) + G(t_{i,j}). \tag{5.9}$$

Thus, the vectorized finite element solution, $U(t)$, and its characteristic derivative, $U_\tau(t)$, must solve the linear system (5.9) for each collocation node, $t = t_{i,j}$. It is important to remember that this equality only holds at the collocation nodes, and not for all t in the time domain.

5.3.2 The fully discrete linear system

The linear system (5.9) is not quite ready to be solved as there are two unknowns, $U(t_{i,j})$ and $U_\tau(t_{i,j})$, which must be related for $j = 1, \ldots, p$ and $i = 1, \ldots, m$. Thus, we expand the finite element solution and its characteristic derivative using the time basis functions, as in (3.6). For the characteristic derivative, this gives

$$\partial_\tau u_h\big(x_k(t_{i,j}), t_{i,j}\big) = \frac{1}{\Delta t_i} \sum_{\ell=0}^{p} \hat{\beta}'_\ell\big(\hat{t}_j\big) u_h(x(t_{i,\ell}), t_{i,\ell})$$

where $\hat{\beta}_j$ and $\hat{t}_j = (t_{i,j} - t_{i-1})/\Delta t_i$ are the basis functions and collocation nodes on the reference element, respectively. In truth, using the collocation nodes as the time basis nodes is only a special case of the basis expansion of the finite element formulation. In general, we do not require the nodes of the time basis to coincide with the collocation nodes of the quadrature rule. This assumption is made in chapter 4 for simplicity; we lift this assumption in this section to attain a more general linear system, which corresponds to a wider variety of time stepping methods for propagating the finite element solution in time.

Let $\{\hat{\zeta}_j\}_{j=0}^p$ be an ordered partition of $[0,1]$ such that

$$0 = \hat{\zeta}_0 < \hat{\zeta}_1 < \ldots < \hat{\zeta}_p \leq 1.$$

We require $\hat{\zeta}_0 = 0$, as this basis node is mapped to the beginning of each time partition by the isoparametric map. Consequently, this ensures that the $\mathcal{L}_2$-projection of the computed solution, $u_h(t_{i^+})$, is incorporated into the fully discrete system. As before, we use the time component of the isoparametric map to distribute these basis nodes through the time domain: $\zeta_{i,j} = t_{i-1} + \hat{\zeta}_j \Delta t_i$. The basis Lagrange basis functions are defined on the reference element as

$$\hat{\lambda}_j(\hat{\zeta}) = \begin{cases} \prod_{\substack{k=0 \\ k \neq j}}^p \frac{\hat{\zeta} - \hat{\zeta}_k}{\hat{\zeta}_j - \hat{\zeta}_k} & \hat{\zeta} \in [0,1], \\ 0 & \text{else.} \end{cases}$$

At collocation node $t_{i,j}$, we have

$$u_h(x, t_{i,j}) = \sum_{\ell=0}^p \hat{\lambda}_\ell(\hat{t}_j) u_h\big(x(\zeta_{i,\ell}), \zeta_{i,\ell}\big)$$

and

$$\partial_\tau u_h(x, t_{i,j}) = \frac{1}{\Delta t_i} \sum_{\ell=0}^p \hat{\lambda}'_\ell(\hat{t}_j) u_h\big(x(\zeta_{i,\ell}), \zeta_{i,\ell}\big),$$

where we assume $x = x(t_{i,j})$ for the characteristic trajectory.

Distinguishing the collocation nodes from the time basis nodes proves to be useful in the next chapter, where Runge-Kutta methods are used to propagate the semi-discrete system in time. In section 5.2, we show that the quadrature rule

determines the order of approximation of the time stepping method; however, the basis nodes can be used to improve the *stability* of the time stepping method.

Using the finite element solution at the time basis nodes, define the pN_i-vector of space-time basis coefficients of u_h by

$$U_i \equiv \begin{bmatrix} U(\zeta_{i,1}) \\ \vdots \\ U(\zeta_{i,p}) \end{bmatrix}.$$

The vectorized finite element solution and its characteristic derivative can be written in terms of U_i. Define the $(pN_i) \times (pN_i)$ block matrices Λ and Λ' by

$$\Lambda \equiv \left[\hat{\lambda}_\ell(\hat{t}_j) I_{N_i}\right]_{(j,\ell)} \quad \text{and} \quad \Lambda' \equiv \left[\hat{\lambda}'_\ell(\hat{t}_j) I_{N_i}\right]_{(j,\ell)}, \tag{5.10}$$

for $1 \leq j, \ell \leq p$, and the $(pN_i) \times N_i$ block matrices by

$$\Lambda_0 \equiv \left[\hat{\lambda}_0(\hat{t}_j) I_{N_i}\right]_j \quad \text{and} \quad \Lambda'_0 \equiv \left[\hat{\lambda}'_0(\hat{t}_j) I_{N_i}\right]_j,$$

for $1 \leq j \leq p$. Then, we have $U(t_{i,j}) = \Lambda U_i + \Lambda_0 U(t_{i-1^+})$ and $U_\tau(t_{i,j}) = \Delta t_i^{-1} \Lambda' U_i + \Delta t_i^{-1} \Lambda'_0 U(t_{i-1^+})$, where we have separated the $U(t_{i-1^+})$ term as it corresponds to the initial condition of the time step. This initial condition is moved to the right side of the linear system since it is known from the $\mathcal{L}_2$-projection (5.2) of the solution in the previous time partition.

Denote the time and space block diagonal matrices by

$$M_i \equiv \begin{bmatrix} \mathcal{M}(t_{i,1}) & 0 & \cdots & 0 \\ 0 & \mathcal{M}(t_{i,2}) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathcal{M}(t_{i,p}) \end{bmatrix}$$

and

$$S_i \equiv \begin{bmatrix} \mathcal{S}(t_{i,1}) & 0 & \cdots & 0 \\ 0 & \mathcal{S}(t_{i,2}) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathcal{S}(t_{i,p}) \end{bmatrix},$$

respectively, and

$$F_i \equiv \begin{bmatrix} F(t_{i,1}) \\ \vdots \\ F(t_{i,p}) \end{bmatrix} \quad \text{and} \quad G_i \equiv \begin{bmatrix} G(t_{i,1}) \\ \vdots \\ G(t_{i,p}) \end{bmatrix}.$$

Then, the fully discrete linear system that determines the finite element solution u_h on $\mathcal{F}_i$ is given by

$$\Big[\frac{1}{\Delta t_i} M_i \Lambda' + S_i \Lambda\Big] U_i = F_i + G_i - \Big[\frac{1}{\Delta t_i} M_i \Lambda_0' + S_i \Lambda_0\Big] U(t_{i-1^+}). \tag{5.11}$$

Hence, a linear system of size pN_i must be solved to find the basis coefficients for the finite element solution on the i^{th} time partition.

5.3.3 Invertibility of the linear system

Denoting the coefficient matrix $A_i = \frac{1}{\Delta t_i} M_i \Lambda' + S_i \Lambda$ and the right side vector $RHS_i = F_i + G_i - \big[\frac{1}{\Delta t_i} M_i \Lambda_0' + S_i \Lambda_0\big] U(t_{i-1^+})$ in (5.11), we have

$$A_i U_i = RHS_i.$$

Since the basis coefficients in the vector U_i uniquely determine the finite element solution of formulation 4 on the i^{th} partition, the existence and uniqueness of the finite element solution is guaranteed when A_i is nonsingular for $i = 1, \ldots, m$.

To determine whether A_i is nonsingular, recall that the mass matrices $\mathcal{M}(t)$ are invertible whenever $\mathcal{T}_h^p(t)$ is a non-degenerate mesh, whereas the invertibility of the stiffness matrices $\mathcal{S}(t)$ depend on the alignment of the mesh and convection velocities, $b(t) - x_t(t)$. Since we do not want to constrain the mesh motion more than necessary, we rely on the invertibility of the mass matrices to prove that A_i is nonsingular. For sufficiently small $\Delta t_i > 0$, the coefficient matrix A_i is dominated by $M_i \Lambda'$ and, accordingly, is invertible whenever $M_i \Lambda'$ is invertible. If we assume that $\mathcal{T}_h^p(t_{i,j})$ satisfies (3.10) and (3.11), $\mathcal{M}(t)$ is invertible and all that is needed is to show that Λ' is invertible. This is accomplished by showing that the $p \times p$ reduced time matrix

$$L' = \big[\hat{\lambda}_\ell(\hat{t}_j)\big]_{(j,\ell)}, \quad 1 \le \ell, j \le p, \tag{5.12}$$

is invertible, since the invertibility of Λ' follows directly by row reduction.

Lemma 6. *Let $\{\hat{\lambda}_j\}_{j=0,\ldots,p}$ be the Lagrange basis for polynomials of degree p with nodes*

$$0 = \hat{\zeta}_0 < \hat{\zeta}_1 < \cdots < \hat{\zeta}_p \leq 1.$$

Then, the matrix L', as defined in (5.12), is invertible.

Proof. Suppose that there exists a p-vector, v, such that $L'v = 0$. Then, the polynomial of degree p defined by

$$v(\hat{t}) \equiv \sum_{j=1}^{p} \hat{\lambda}_j(\hat{t}) v_j$$

satisfies $v'(\hat{t}_j) = 0$ for $j = 1, \ldots, p$. Hence, the derivative $v'(\hat{t})$ has p distinct roots, by the hypothesis, meaning that $v'(\hat{t}) \equiv 0$ because it has degree at most $p-1$. As a result, the polynomial $v(\hat{t})$ is constant and $\hat{\lambda}_j(0) = 0$ for $j = 1, \ldots, p$ implies that $v(\hat{t}) \equiv 0$. Hence, the vector $v = 0$, which shows that the matrix L' is invertible. □

From lemma 6, the linear system corresponding to the finite element formulation is nonsingular, which implies that the finite element solution computed in formulation 4 exists and is unique when the discrete time basis nodes are distinct and the spatial meshes at the time collocation nodes satisfy (3.10) and (3.11).

Furthermore, note that the condition number of the coefficient matrix is influenced by the shape regularity and $b - x_t$ via the block diagonal mass and stiffness matrices, M_i and S_i. Hence, the finite element mesh influences the conditioning of the linear system through these block diagonal matrices. The other block matrices, Λ' and Λ, can also be used to improve the condition number of the linear system for appropriate choices of time basis nodes; this corresponds to the stability of the time discretization, which is determined by the collocation and time basis nodes. The connections between the time collocation and basis nodes with Runge-Kutta methods are the topic of chapter 6. This is useful for introducing the notion of stable time stepping schemes, which correspond to "A-stable" or "L-stable" Runge-Kutta methods.

Basis representation of finite element functions

For analytical purposes, the choice of basis functions used to represent the finite element functions do not affect the solution of finite element formulation; however, as a practical matter, it has been pointed out that the choice of basis is very important for computational stability and efficiency of the time stepping. In the analysis of this chapter, we choose collocation and basis nodes to coincide, $\hat{\zeta}_j = \hat{t}_j$, as the theoretical properties finite element functions are independent of their basis representations. As such, let

$$B = L' = \begin{bmatrix} \hat{\beta}_1'(\hat{t}_1) & \hat{\beta}_2'(\hat{t}_1) & \cdots & \hat{\beta}_p'(\hat{t}_1) \\ \hat{\beta}_1'(\hat{t}_2) & \hat{\beta}_2'(\hat{t}_2) & \cdots & \hat{\beta}_p'(\hat{t}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{\beta}_1'(\hat{t}_p) & \hat{\beta}_2'(\hat{t}_p) & \cdots & \hat{\beta}_p'(\hat{t}_p) \end{bmatrix}.$$

5.4 Two discrete Grönwall inequalities

Since we are analyzing methods for solving time-dependent problems, a Grönwall inequality provides the tools necessary for bounding the error of the computed solution over the time domain by the approximation error of the initial condition and the source terms. For our purposes, two discrete Grönwall estimates are required; the first corresponds to bounding the local error accumulated within each time partition and the second, which comes from [18], corresponds to bounding the error that aggregates over all time partitions.

We begin with the local Grönwall inequality that can bound the error within the time partitions. An important property that follows from the invertibility of matrix $B = L'$, proven in lemma 6, is that there exists a p-vector, $\vec{\alpha}_k = [\alpha_j^{(k)}]_{j=1}^p$, such that

$$B^T \vec{\alpha}_k = \begin{bmatrix} \sum_{j=1}^p \alpha_j^{(k)} \hat{\beta}_1'(\hat{t}_j) \\ \vdots \\ \sum_{j=1}^p \alpha_j^{(k)} \hat{\beta}_p'(\hat{t}_j) \end{bmatrix} = \vec{e}_k = \begin{bmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \leftarrow k^{\text{th}} \text{ element} \tag{5.13}$$

where $\vec{\alpha}_k = B^{-T}\vec{e}_k$ only depends on the collocation nodes, $\{\hat{t}_j\}$, on the reference element.

Lemma 7 (Local Grönwall Inequality)**.** *Let $\hat{\mathcal{Q}}$ be the reference quadrature rule for $\mathcal{Q}_i$ with positive weights and p distinct nodes on the unit interval and suppose that the mesh partition $\mathcal{T}_{h,i}^p$ satisfies the space-time regularity constraint (3.14) at the collocation nodes and there exists a positive constant κ such that*

$$\|b - x_t\|_\infty \leq \kappa. \tag{5.14}$$

If functions ϕ in $\mathcal{V}_{h,i}^p$ and η in $\mathcal{V}|_{\mathcal{F}_i}$ satisfy

$$\big(\phi_\tau(t_{i,j}), \chi\big) + \mathcal{A}_\tau\big(\phi, \chi; t_{i,j}\big) = \big(\eta_\tau(t_{i,j}), \chi\big) + \mathcal{A}_\tau\big(\eta, \chi; t_{i,j}\big) \tag{5.15}$$

for all χ in $\mathcal{V}_h^p(t_{i,j})$ at time $t_{i,j}$ and $\Delta t_i \leq 1/2\tilde{c}_{\mu,d}$, as defined in (3.16), then, there exists a constant such that

$$\max_{1\leq j\leq p} \left\|\phi(t_{i,j})\right\|_0^2 \leq C\left\{\left\|\phi(t_{i-1^+})\right\|_0^2 + \Delta t_i \mathcal{Q}_i\left(\|\eta_\tau\|_{(-1,\mathcal{V}_h^p(\cdot))}^2 + \|\eta\|_1^2 + \|\phi\|_1^2\right)\right\},$$

where C depends on κ, μ, d, p, the reference collocation nodes $\{\hat{t}_j\}_{j=1}^p$, and the differential equation.

Proof. To simplify notation, the time partition index, i, is assumed to be fixed and we let $\phi_j \equiv \phi(t_{i,j})$. Define $\tilde{\phi}_\ell^{(j)}$ in $\mathcal{V}_h^p(t_{i,j})$ to be the finite element function with the same basis coefficients as ϕ_ℓ multiplying the basis functions for time $t_{i,j}$. The function $\tilde{\phi}_\ell^{(j)}$ is, therefore, the shift of $\phi(t_{i,\ell})$ onto the mesh at time $t_{i,j}$, and $\tilde{\phi}_j^{(j)} = \phi_j$.

Let k index the collocation node where ϕ attains its maximimal $\mathcal{L}_2$ norm: $\|\phi_k\|_0 = \max_{1\leq j\leq p}\|\phi_j\|_0$. The crux of this proof is to use the linear combination given in (5.13) to combine the equations (5.15) such that the characteristic derivative terms cancel out and simplify to isolate $\|\phi_k\|_0$ on the left side of the equation, which is in turn bounded to get the desired result.

Choose $\chi = \tilde{\phi}_k^{(j)}$ in $\mathcal{V}_h^p(t_{i,j})$ for equation (5.15), $j = 1, \ldots, p$. Use the time basis expansion of the characteristic derivative (3.6) to get

$$\frac{1}{\Delta t_i}\sum_{\ell=0}^{p} \hat{\beta}_\ell'(\hat{t}_j)\big(\tilde{\phi}_\ell^{(j)}, \tilde{\phi}_k^{(j)}\big) + \mathcal{A}_\tau(\phi_j, \tilde{\phi}_k^{(j)}; t_{i,j}) = \big(\eta_{j,\tau}, \tilde{\phi}_k^{(j)}\big) + \mathcal{A}_\tau(\eta_j, \tilde{\phi}_k^{(j)}; t_{i,j}),$$

for $j = 1, \ldots, p$. Equivalently,

$$\sum_{\ell=1}^{p} \hat{\beta}'_\ell(\hat{t}_j)\Big(\tilde{\phi}_\ell^{(j)}, \tilde{\phi}_k^{(j)}\Big) = -\hat{\beta}'_0(\hat{t}_j)\Big(\tilde{\phi}_0^{(j)}, \tilde{\phi}_k^{(j)}\Big) + \Delta t_i \left\{ \Big(\eta_{j,\tau}, \tilde{\phi}_k^{(j)}\Big) + \mathcal{A}_\tau\Big(\eta_j - \phi_j, \tilde{\phi}_k^{(j)}; t_{i,j}\Big)\right\}. \quad (5.16)$$

Taking the linear combination defined in (5.13), the left side can be simplified and bounded using the shift lemma 2:

$$\begin{aligned}
&\sum_{j=1}^{p} \alpha_j^{(k)} \sum_{\ell=1}^{p} \hat{\beta}'_\ell(\hat{t}_j)\Big(\tilde{\phi}_\ell^{(j)}, \tilde{\phi}_k^{(j)}\Big) \\
&\geq \sum_{j=1}^{p} \alpha_j^{(k)} \sum_{\ell=1}^{p} \hat{\beta}'_\ell(\hat{t}_j)\Big(\tilde{\phi}_\ell^{(0)}, \tilde{\phi}_k^{(0)}\Big) - \frac{1}{2} C_\mu \Delta t_i \sum_{j=1}^{p} \Big|\alpha_j^{(k)}\Big| \sum_{\ell=1}^{p} \Big|\hat{\beta}'_\ell(\hat{t}_j)\Big| \left(\big\|\tilde{\phi}_\ell^{(0)}\big\|_0^2 + \big\|\tilde{\phi}_k^{(0)}\big\|_0^2 \right) \\
&\geq \big\|\tilde{\phi}_k^{(0)}\big\|_0^2 - \hat{C}_{\hat{\mathcal{Q}},\mu,d,p} \Delta t_i \sum_{j=1}^{p} \big\|\tilde{\phi}_j^{(0)}\big\|_0^2 \\
&\geq \big\|\tilde{\phi}_k^{(0)}\big\|_0^2 - \hat{C}'_{\hat{\mathcal{Q}},\mu,d,p} \Delta t_i \sum_{j=1}^{p} \|\phi_j\|_0^2, \quad (5.17)
\end{aligned}$$

where the dependence of the constant on the quadrature rule $\hat{\mathcal{Q}}$ comes from the fact that the basis functions $\hat{\beta}_\ell(\hat{t}_j)$ and the linear combination $\vec{\alpha}_k$ depend on the reference collocation nodes, $\{\hat{t}_j\}_{j=1}^p$.

To bound the right side, choose $\delta > 0$ to be sufficiently small, say $\delta < 1/2$, and use the shift lemma (lemma 2) to get

$$\begin{aligned}
-\sum_{j=1}^{p} \alpha_j^{(k)} \hat{\beta}'_0(\hat{t}_j)\Big(\tilde{\phi}_0^{(j)}, \tilde{\phi}_k^{(j)}\Big) &\leq \frac{1}{2} \sum_{j=1}^{p} \big|\alpha_j^{(k)} \hat{\beta}'_0(\hat{t}_j)\big| \big\|\tilde{\phi}_0^{(j)}\big\|_0 \big\|\tilde{\phi}_k^{(j)}\big\|_0 \\
&\leq \hat{C}''_{\hat{\mathcal{Q}},\mu,d,p} \big\|\phi_0\big\|_0^2 + \delta \big\|\tilde{\phi}_k^{(0)}\big\|_0^2. \quad (5.18)
\end{aligned}$$

Furthermore, by the definition of the mesh dependent negative norm and the shift lemma,

$$\Big(\eta_{j,\tau}, \tilde{\phi}_k^{(j)}\Big) \leq \frac{1}{2} \left(\|\eta_{j,\tau}\|^2_{(-1,\mathcal{V}_h^p(t_{i,j}))} + C_{\mu,d,p} \|\phi_k\|_1^2 \right). \quad (5.19)$$

From lemmas 1 and 2,

$$\begin{aligned}
\mathcal{A}_\tau\Big(\eta_j - \phi_j, \tilde{\phi}_k^{(j)}; t_{i,j}\Big) &\leq C_{\mathcal{A},\kappa} \left(\|\eta_j\|_1^2 + \|\phi_j\|_1^2 + \big\|\tilde{\phi}_k^{(j)}\big\|_1^2 \right) \\
&\leq C_{\mathcal{A},\kappa} \left(\|\eta_j\|_1^2 + \|\phi_j\|_1^2 + C_{\mu,d,p} \|\phi_k\|_1^2 \right). \quad (5.20)
\end{aligned}$$

From (5.17)–(5.20), we get

$$\left\|\tilde{\phi}_k^{(0)}\right\|_0^2 \leq C_{\hat{\mathcal{Q}},\mu,d,p}\Big\{\|\phi(t_{i-1^+})\|_0^2 + C_{\mathcal{A},\kappa}\Delta t_i \sum_{j=1}^{p}\Big(\|\eta_{j,\tau}\|^2_{(-1,\mathcal{V}_h^p(t_{i,j}))} + \|\eta_j\|_1^2 + \|\phi_j\|_1^2\Big)\Big\}. \tag{5.21}$$

From the shift lemma and $\Delta t_i \leq 1/2\tilde{c}_{\mu,d}$,

$$\left\|\tilde{\phi}_k^{(0)}\right\|_0^2 \leq \frac{1}{1-\tilde{c}_{\mu,d}\Delta t_i}\|\phi_k\|_0^2 \leq 2\|\phi_k\|_0^2, \tag{5.22}$$

and since the weights of the quadrature rule are positive, there's a constant $C_{\hat{\mathcal{Q}}} > 0$ such that

$$\sum_{j=1}^{p}\Big(\|\eta_{j,\tau}\|^2_{(-1,\mathcal{V}_h^p(t_{i,j}))} + \|\eta_j\|_1^2 + \|\phi_j\|_1^2\Big) \leq C_{\hat{\mathcal{Q}}}\mathcal{Q}_i\Big(\|\eta_\tau(\cdot)\|^2_{(-1,\mathcal{V}_h^p(\cdot))} + \|\eta(\cdot)\|_1^2 + \|\phi(\cdot)\|_1^2\Big). \tag{5.23}$$

Since $\|\phi_k\|_0 = \max_{1\leq j\leq p}\|\phi_j\|_0$, combining the bounds (5.21)–(5.23) yields the desired result. □

The next bound is the discrete Grönwall lemma that bounds the errors that accumulates over the time partitions. This discrete Grönwall lemma comes directly from [65] and was also used in the error analysis [18] for linear moving finite elements with backward Euler time stepping — in the current context, this corresponds to the case where $\hat{t}_0 = 0$ and $\hat{t}_1 = 1$.

Lemma 8 (Discrete Grönwall Inequality)**.** *Let $\Delta t_i > 0$ and $\alpha_i, \gamma_i, \theta_i, q_i \geq 0$, for $1 \leq i \leq m$, with $\theta_i\Delta t_i \leq \frac{1}{2}$ and $\theta = \max_i \theta_i$. Then, if*

$$\frac{q_i - q_{i-1}}{\Delta t_i} + \gamma_i \leq \alpha_i + \theta_i(q_i + q_{i-1}),$$

there exists a positive constant C_θ such that

$$\max_{1\leq i\leq m} q_i + \sum_{i=1}^{m}\gamma_i\Delta t_i \leq C_\theta\left\{q_0 + \sum_{i=1}^{m}\alpha_i\Delta t_i\right\}.$$

This theorem comes directly from [65] and the proof can be found therein.

5.5 Error estimates of the finite element solution

In section 4.3, approximation properties are established for the finite element space. In this section, we give two error estimates for the finite element solution that is defined by formulation 4. These are the main results for the space-time moving finite element method proposed in this chapter. It is shown that the finite element solution satisfies a quasi-optimal error bound, and displays order-p convergence to the true solution of formulation 3 with respect to mesh refinement.

5.5.1 A symmetric error estimate

The most important analytical result of this chapter is the a priori error bound given here. This bound shows that the finite element method proposed in this chapter gives a quasi-optimal solution to the differential equation when measured in the energy semi-norm, defined in (4.13). It is implicitly assumed that the time step is sufficiently small so that the finite element formulation is well-posed.

Theorem 2. *Suppose that $\mathcal{V}_h^p$ is a finite element space with a non-degenerate mesh and collocation nodes determined by a non-truncating reference quadrature rule. Furthermore, assume that there exist positive constants μ and κ such that at each collocation node*

$$\rho\Big(\mathcal{H}_e(t_{i,j})\Big) \leq \mu, \tag{5.24}$$

and

$$\|b - x_t\|_\infty \leq \kappa. \tag{5.25}$$

Then, if $\Delta t = \max_{1\leq i\leq m} \Delta t_i$ is sufficiently small, there exists a positive constant C such that the finite element solution defined by formulation 4 satisfies

$$|\!|\!| u - u_h |\!|\!| \leq C \inf_{\chi\in\mathcal{V}_h^p} |\!|\!| u - \chi |\!|\!|, \tag{5.26}$$

where C depends on μ, κ, d, p, the reference collocation nodes $\{\hat{t}_j\}_{j=1}^p$, and the differential equation.

To comment on the result (5.26), note that the bounding constant does not depend on the finite element space, except indirectly by the shape regularity assumption assumption (5.24) and the boundedness of $b - x_t$, whereas the energy semi-norm does in fact depend on the choice of finite element space. This dependence on the mesh can be removed by the property that $\|v\|_{(-1,\mathcal{V}_h^p(t))} \leq \|v\|_{-1}$, for all v in $\mathcal{L}_2(\Omega)$ and $0 < t \leq T$. However, this comes at the cost of breaking the symmetry of the error bound.

Proof. This proof follows the arguments of Douglas and Dupont in [37]; this argument was originally designed for semi-discrete formulations, where the differential equation (6.5) holds for all t in the time domain. Subsequently, the argument was modified for fully discrete linear finite elements in [18], [65], and [40]. The proof can be summarized as first bounding the error introduced at the collocation nodes by the spatial discretization by using the Galerkin orthogonality (5.3) and then applying the Grönwall lemmas in order to aggregate the spatial errors over the time domain.

The constraint (5.24) in the hypothesis is the mesh regularity assumption to bound the degree to which elements in the mesh can change their size or shape; controlling element deformation through a time step is analogous to controlling the size of μ. The bound (5.25) is a constraint that bounds the mesh velocity. Smaller bounding constants in these required constraints imply improved space-time shape regularity and alignment between the mesh and convection, which leads to a smaller bounding constant in (5.26).

From the Galerkin orthogonalities, at each collocation node

$$\big(\partial_\tau u_h(t_{i,j}), \chi\big) + \mathcal{A}_\tau\big(u_h, \chi; t_{i,j}\big) = \big(\partial_\tau u(t_{i,j}), \chi\big) + \mathcal{A}_\tau\big(u, \chi; t_{i,j}\big),$$

for all χ in $\mathcal{V}_h^p(t_{i,j})$ with $i = 1, \ldots, m$ and $j = 1, \ldots, p$. Let $\psi \in \mathcal{V}_h^p$ and define $\phi \equiv u_h - \psi$ in $\mathcal{V}_h^p$ and $\eta \equiv u - \psi$. Re-write the statement of the Galerkin orthogonality as

$$(\partial_\tau \phi_{i,j}, \chi) + \mathcal{A}_{i,j}(\phi, \chi) = (\partial_\tau \eta_{i,j}, \chi) + \mathcal{A}_{i,j}(\eta, \chi), \tag{5.27}$$

for all χ in $\mathcal{V}_h^p(t_{i,j})$. Using equation (5.27) we will show

$$|\!|\!| \phi |\!|\!| \leq C |\!|\!| \eta |\!|\!|$$

and use the triangle inequality to obtain the sought after bound (5.26).

Fix i and j and choose $\chi = \phi(t_{i,j})$ so that (5.27) gives

$$(\phi_\tau, \phi) + \mathcal{A}_\tau(\phi, \phi) = (\eta_\tau, \phi) + \mathcal{A}_\tau(\eta, \phi), \tag{5.28}$$

at time $t = t_{i,j}$. The bound at this collocation node comes from bounding the terms in (5.28) individually. For the first term on the left,

$$(\phi_\tau, \phi) = \frac{1}{2}\partial_\tau \|\phi\|_0^2. \tag{5.29}$$

And, since the mesh motion satisfies (5.25), the lower bound (4.2) from lemma 1 shows

$$\mathcal{A}_\tau(\phi, \phi) \geq C_{\mathcal{A}} \|\phi\|_1^2 - C_{\mathcal{A},\kappa} \|\phi\|_0^2. \tag{5.30}$$

Now, choose $\varepsilon > 0$ to be sufficiently small and note

$$(\eta_\tau, \phi) \leq C \|\eta_\tau\|^2_{(-1,\mathcal{V}_h^p(t_{i,j}))} + \varepsilon \|\phi\|_1^2 \tag{5.31}$$

and, since (5.25) holds, apply lemma 1 to $\frac{1}{\sqrt{\varepsilon}}\eta$ and $\sqrt{\varepsilon}\phi$ to get

$$\mathcal{A}_\tau(\eta, \phi) \leq C_{\mathcal{A},\kappa} \|\eta\|_1^2 + \varepsilon \|\phi\|_1^2. \tag{5.32}$$

Hence, from (5.28)–(5.32), it is true at $t = t_{i,j}$ that

$$\frac{1}{2}\partial_\tau \|\phi\|_0^2 + C_{\mathcal{A}} \|\phi\|_1^2 \leq C_{\mathcal{A},\kappa} \left\{ \|\eta_\tau\|^2_{(-1,\mathcal{V}_h^p(t_{i,j}))} + \|\eta\|_1^2 + \|\phi\|_0^2 \right\}. \tag{5.33}$$

The bound (5.33) is reminiscent of the hypothesis of a Grönwall inequality, except that our bound only holds at the discrete time steps. Furthermore, the term corresponding to the time derivate in our bound is actually a characteristic derivative, which we must contend with before the time discretization error is bounded. Using the bound (4.18) from lemma 2 together with lemma 3,

$$\begin{aligned} \frac{1}{2}\partial_\tau \|\phi\|_0^2 = (\phi_\tau, \phi) &\geq (\tilde{\phi}_t, \tilde{\phi}) - C_{\mu,d,p}\left(\left\| \Delta t_i \tilde{\phi}_t \right\|_0^2 + \left\| \tilde{\phi} \right\|_0^2 \right) \\ &\geq \frac{1}{2}\frac{d}{dt} \left\| \tilde{\phi} \right\|_0^2 - C_{\hat{\mathcal{Q}},\mu,d,p} \max_{1 \leq j \leq p} \|\phi_{i,j}\|_0^2. \end{aligned} \tag{5.34}$$

Applying the quadrature rule to the i^{th} partition gives

$$\begin{aligned} &\mathcal{Q}_i \left(\frac{d}{dt} \|\tilde{\phi}\|_0^2 \right) + C_{\mathcal{A}} \mathcal{Q}_i \left(\|\phi\|_1^2 \right) \\ &\qquad \leq C_{\mathcal{A},\hat{\mathcal{Q}},\mu,\kappa,d,p} \left\{ \mathcal{Q}_i \left(\|\eta_\tau\|^2_{(-1,\mathcal{V}_h^p(\cdot))} + \|\eta\|_1^2 \right) + \mathcal{Q}_i \left(\|\phi\|_0^2 \right) \right\}. \end{aligned} \tag{5.35}$$

This is almost of the form to which the discrete Grönwall lemma (lemma 8) applies, except for the terms $\mathcal{Q}_i(\frac{d}{dt}\|\tilde{\phi}\|_0^2)$ and $\mathcal{Q}_i(\|\phi\|_0^2)$. To get these terms in the proper form, we apply the local Grönwall lemma (lemma 7).

For the first term, since $\hat{\mathcal{Q}}$ is assumed to be non-truncating and $\frac{d}{dt}\|\tilde{\phi}(t)\|_0^2$ is a polynomial of degree $2p-1$ with a positive leading coefficient, we use the shift lemma:

$$\begin{aligned}\mathcal{Q}_i\left(\frac{d}{dt}\|\tilde{\phi}\|_0^2\right) &\geq \frac{1}{\Delta t_i}\int_{t_{i-1}}^{t_i}\frac{d}{dt}\|\tilde{\phi}(t)\|_0^2\,dt \\ &= \frac{\left\|\tilde{\phi}(t_{i^-})\right\|_0^2 - \|\phi(t_{i-1^+})\|_0^2}{\Delta t_i} \\ &\geq \frac{\|\phi(t_{i^-})\|_0^2 - \|\phi(t_{i-1^+})\|_0^2}{\Delta t_i} - C_{\mu,d,p}\|\phi(t_{i^-})\|_0^2. \end{aligned} \tag{5.36}$$

The next term is bounded by lemma 7,

$$\begin{aligned}\mathcal{Q}_i\left(\|\phi\|_0^2\right) &\leq C\max_{1\leq j\leq p}\|\phi_{i,j}\|_0^2 \\ &\leq C\left\{\|\phi(t_{i-1^+})\|_0^2 + \Delta t_i\mathcal{Q}_i\left(\|\eta_\tau\|^2_{(-1,\mathcal{V}_h^p(\cdot))} + \|\eta\|_1^2 + \|\phi\|_1^2\right)\right\}. \end{aligned} \tag{5.37}$$

Thus, if Δt_i is sufficiently small, bounds (5.35)–(5.37) combine to yield

$$\begin{aligned}&\frac{\|\phi(t_{i^-})\|_0^2 - \|\phi(t_{i-1^+})\|_0^2}{\Delta t_i} + C_0\mathcal{Q}_i\left(\|\phi\|_1^2\right) \\ &\qquad\leq C_1\mathcal{Q}_i\left(\|\eta_\tau\|^2_{(-1,\mathcal{V}_h^p(\cdot))} + \|\eta\|_1^2\right) + C_2\big(\|\phi(t_{i^-})\|_0^2 + \|\phi(t_{i-1^+})\|_0^2\big),\end{aligned}$$

where $C_0, C_1, C_2 > 0$ depend on μ, κ, d, p, the reference quadrature rule, and the differential equation. Since $\phi(t_{i-1^+})$ is the $\mathcal{L}_2$-projection of $\phi(t_{i-1^-})$, we have

$$\|\phi(t_{i-1^+})\|_0 \leq \|\phi(t_{i-1^-})\|_0,$$

which we use to get

$$\begin{aligned}&\frac{\|\phi(t_{i^-})\|_0^2 - \|\phi(t_{i-1^-})\|_0^2}{\Delta t_i} + C_0\mathcal{Q}_i\left(\|\phi\|_1^2\right) \\ &\qquad\leq C_1\mathcal{Q}_i\left(\|\eta_\tau\|^2_{(-1,\mathcal{V}_h^p(\cdot))} + \|\eta\|_1^2\right) + C_2(\|\phi(t_{i^-})\|_0^2 + \|\phi(t_{i-1^-})\|_0^2).\end{aligned} \tag{5.38}$$

The bound (5.38) satisfies the hypothesis of the discrete Grönwall lemma so that, for sufficiently small Δt_i, we apply the lemma to obtain

$$\max_{0\leq i\leq m}\|\phi(t_{i^-})\|_0^2 + \mathcal{Q}(\|\phi\|_1^2) \leq C\left\{\|\phi(0)\|_0^2 + \|\eta\|^2\right\}. \tag{5.39}$$

Note that the initial condition of formulation 4 gives

$$\|\phi(0)\|_0 \leq \|\eta(0)\|_0 \leq |||\eta|||. \tag{5.40}$$

Furthermore, at each collocation node, $t_{i,j}$,

$$(\partial_\tau \phi, \chi) = (\partial_\tau \eta, \chi) + \mathcal{A}_\tau(\eta, \chi) - \mathcal{A}_\tau(\phi, \chi)$$

for all χ in $\mathcal{V}_h^p(t_{i,j})$, which implies that

$$\|\partial_\tau \phi\|_{(-1,\mathcal{V}_h^p(t_{i,j}))} \leq C_{\mathcal{A},\kappa} \left\{ \|\partial_\tau \eta\|_{(-1,\mathcal{V}_h^p(t_{i,j}))} + \|\eta\|_1 + \|\phi\|_1 \right\}. \tag{5.41}$$

From (5.39)–(5.41),

$$\max_{0\leq i\leq m} \|\phi(t_{i^-})\|_0^2 + \mathcal{Q}\big(\|\phi\|_1^2 + \|\phi\|_{(-1,\mathcal{V}_h^p(\cdot))}^2\big) \leq C|||\eta|||^2.$$

All that is needed to conclude the proof is the local Grönwall inequality once more to show

$$\max_{1\leq j\leq p} \|\phi_{i,j}\|_0^2 \leq C \left\{ \|\phi(t_{i-1^+})\|_0^2 + \Delta t_i \mathcal{Q}_i \left(\|\eta_\tau\|_{(-1,\mathcal{V}_h^p(\cdot))}^2 + \|\eta\|_1^2 + \|\phi\|_1^2 \right) \right\} \leq C|||\eta|||^2 \tag{5.42}$$

for $i = 1, \ldots, m$. Thus, we have $|||\phi||| \leq C|||\eta|||$, as desired. □

5.5.2 An a posteriori error estimate

We now use the shape regularity assumptions of $\mathcal{T}_h^p$ to bound the error of the finite element solution, u_h, in terms of the mesh size. This result is useful for determining the rate of convergence of u_h to u as the mesh is refined. A direct corollary of theorems 1 and 2 is the following a posteriori bound.

Corollary 1. *Let the finite element space, $\mathcal{V}_h^p$, and the reference quadrature rule, $\hat{\mathcal{Q}}$, satisfy the hypotheses of theorems 1 and 2. Then, there exists a constant $C > 0$ such that finite element solution determined by formulation 4 satisfies*

$$|||u - u_h||| \leq C(\Delta x^p + \Delta t^p)\mathcal{E}_{(\mathcal{Q},\tau,p+1)}(u), \tag{5.43}$$

where C depends on $\delta, \mu, \kappa, d, p$, the reference collocation nodes $\{\hat{t}_j\}_{j=1}^p$, and the differential equation.

5.5.3 Closing remarks

While we attain a symmetric error estimate and order p convergence of the finite element solution to the true solution of the weak formulation, it is unfortunate that the choice of reference quadrature rule is restricted to be a non-truncating quadrature rule. This is a drawback of analyzing the error of the finite element solution, which is determined by a method of lines approach, in a purely finite element framework. As far as we know, the requirement of a non-truncating quadrature rule is required to make the theoretical error analysis provide the symmetric error bound and the corresponding restrictions for the case $p = 1$ are present in the error analysis for linear elements in [40]. One difficulty with these choices for quadrature rules is that they require the finite element solution be computed at all collocation nodes in a time partition simultaneously, which can significantly increase the computational complexity when $p > 1$. These choices of collocation nodes rule out many choices for more computationally efficient time integration schemes. The following section generalizes the space-time finite element method to allow for more general time stepping schemes where the discrete time integration no longer needs to correspond to some interpolatory quadrature rule. Moreover, an error bound for finite element solutions computed by a second order diagonally implicit time integration scheme, TR-BDF2 as proposed by Bank et al. [23], is proven.

Chapter 5, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 6

Space-Time Moving Finite Elements with Runge-Kutta Methods for Time Integration

The chapter begins with a brief discussion of Runge-Kutta methods for solving ordinary differential equations. These methods are designed for solving ordinary differential equations and compute the solution of the PDE sequentially through the time domain. In the current context, they are employed as time integration schemes of the semi-discrete formulation to propagate the finite element solution in time. In particular, we discuss the TR-BDF scheme proposed in [23], as this is an A-stable, L-stable, second order, and diagonally implicit time stepping scheme. The well-posedness of Runge-Kutta space-time moving finite element methods is analyzed as in section 5.3, where we see that TR-BDF time stepping yields a unique solution for sufficiently small time steps. While we have not found a satisfactory error estimate for general Runge-Kutta methods applied to the semi-discrete finite element formulation, this chapter presents an error estimate for a space-time moving finite element method that employs TR-BDF time integration.

6.1 A brief overview of some Runge-Kutta schemes

Consider the initial value problem

$$v_t(t) = \Phi(t, v(t)), \qquad v(0) = v_0, \tag{6.1}$$

where $v \in I\!R^N$ and Φ is some source function, $N \geq 1$. To discretize equation (6.1), we define a p-stage Runge-Kutta scheme by a set of collocation nodes $t_{i,j} = t_{i-1} + \hat{t}_j \Delta t_i$, along with a set of basis nodes $\zeta_{i,k} = t_{i-1} + \hat{\zeta}_k \Delta t_i$, $k = 1, \ldots, p$, where it is assumed that $0 = \hat{\zeta}_0 < \ldots < \hat{\zeta}_p \leq 1$. The solution is propagated over the time step $(t_{i-1}, t_i]$ using some approximation for the time derivative

$$\bar{\delta}_t v(t_{i,j}) \equiv \sum_{k=0}^{p} \omega_{j,k} v(\zeta_{i,k}) \approx \Delta t_i v_t(t_{i,j}), \tag{6.2}$$

and some approximation of the function itself

$$v_{RK}(t_{i,j}) \equiv \sum_{k=0}^{p} \xi_{j,k} v(\zeta_{i,k}) \approx v(t_{i,j}), \tag{6.3}$$

where coefficients $\{\omega_{j,k}\}$ and $\{\xi_{j,k}\}$, with $0 \leq j,k \leq p$, are determined by the Runge-Kutta method. Thus, we need to find the values of v at times $t = t_{i,j}$ such that the discretization of equation (6.1), given by

$$\bar{\delta}_t v(t_{i,j}) = \Delta t_i \Phi\big(v_{RK}(t_{i,j})\big), \tag{6.4}$$

is satisfied for $i = 1, \ldots, m$ and $j = 1, \ldots, p$. Though these schemes may be p-stage methods (as they employ intermediate timesteps $t_{i-1} < t_{i,j} \leq t_i$), they are necessarily *one-step* methods, meaning that we only require information from the end of previous time step, t_{i-1^-}, to compute the approximate solution at time t_i. An important quality of one-step methods is that they do not give spurious solutions, as is the case for multi-step methods, from which the desired computed solution must then be selected [44].

An example: collocation based formulae

A special case of Runge-Kutta schemes correspond to approximating the function v in (6.1) by a piecewise polynomial on the time partition $0 < t_1 \leq \ldots \leq t_m = T$. Define the collocation nodes

$$0 = \hat{t}_0 < \ldots < \hat{t}_p \leq 1$$

and

$$0 = \hat{\zeta}_0 < \ldots < \hat{\zeta}_p \leq 1,$$

as in section 5.3. Then define the basis functions $\{\hat{\lambda}_j(\hat{t})\}$ to be the Lagrange basis polynomial with nodes at $\{\hat{\zeta}_j\}$, $j = 0, \ldots, p$. If we define

$$v_{RK}(t_{i,j}) = \sum_{k=0}^{p} \hat{\lambda}_k(\hat{t}_j) v(\hat{\zeta}_k),$$

and

$$\bar{\delta}_t v(t_{i,j}) = \sum_{k=0}^{p} \hat{\lambda}_k(\hat{t}_j) v(\hat{\zeta}_k),$$

then we have exactly recovered the time integration scheme developed in section 5.3.

Another example: TR-BDF

The motivating example for this chapter is to employ the two-stage, diagonally implicit time integration scheme TR-BDF. The fact that this is a diagonally implicit scheme is significant when $v(t)$ in equation (6.1) represents a vector of length N and N is large. The diagonal implicitness leads to great savings in computation costs for such cases, which includes the important and relevant case where (6.1) corresponds to a system of ordinary differential equations arising from a method of lines approach. This scheme was proposed in [23] and has been analyzed in several other papers for its efficiency and stability ([72],[43],[36]). TR-BDF actually refers to a family of time stepping methods that is parametrized by the location of the intermediate basis node.

For this method, we define the collocation nodes to be

$$\hat{t}_0 = 0, \quad \hat{t}_1 = \varepsilon/2, \quad \text{and} \quad \hat{t}_2 = 1,$$

where $0 < \varepsilon < 1$ is the free parameter that determines the exact time stepping scheme within the TR-BDF family. We choose the basis nodes to be

$$\hat{\zeta}_0 = 0, \quad \hat{\zeta}_1 = \varepsilon, \quad \text{and} \quad \hat{\zeta}_2 = 1$$

so that $\hat{t}_1$ is the midpoint of $\hat{\zeta}_0$ and $\hat{\zeta}_1$. The Runge-Kutta coefficients correspond to integrating the computed solution a step of length $\varepsilon\Delta t_i$ by the trapezoid rule, then completing the time step by a second-order backward Euler difference:

$$\begin{aligned} v_{RK}(t_{i,1}) &= \frac{1}{2}v(\zeta_{i,0}) + \frac{1}{2}v(\zeta_{i,1}), \\ v_{RK}(t_{i,2}) &= v(\zeta_{i,2}). \end{aligned}$$

The coefficients for the time derivative are determined by evaluating the Lagrange basis polynomials associated with $\{\zeta_{i,0}, \zeta_{i,1}, \zeta_{i,2}\}$ at the collocation nodes:

$$\begin{aligned} \bar{\delta}_t v(t_{i,1}) &= -\frac{1}{\varepsilon}v(\zeta_{i,0}) + \frac{1}{\varepsilon}v(\zeta_{i,1}), \\ \bar{\delta}_t v(t_{i,2}) &= \frac{1-\varepsilon}{\varepsilon}v(\zeta_{i,0}) - \frac{1}{\varepsilon(1-\varepsilon)}v(\zeta_{i,1}) + \frac{2-\varepsilon}{1-\varepsilon}v(\zeta_{i,2}). \end{aligned}$$

The optimal choice for the parameter is known to be $\varepsilon = 2 - \sqrt{2}$, as it minimizes the local truncation error ([23],[72]). This value also allows identical

Jacobian matrices for solving at the TR step and the BDF step when the source terms (and boundary conditions) relatively smooth and gradually changing (this is the Richardson TR-BDF method and we refer to the choice $\varepsilon = 2 - \sqrt{2}$ as the Richardson basis node) ([31],[23]). Furthermore, improved stability in the right side of the complex plane has been shown for this value of ε [36]. The TR-BDF scheme is A-stable (or unconditionally stable), which guarantees that the computed solution decays in time when the reaction term in the differential equation is positive [44]. The L-stability is another very important property of this scheme (in fact, the scheme has been shown to possess strong S-stability [43]), as it implies that the computed solution does not display artificial oscillations induced by the time stepping scheme [23]. Thus, this scheme is well suited to diffusion problems; if the convection and mesh velocity are sufficiently aligned, this scheme provides the stability properties to ensure that the diffusion is propagated accurately in time.

6.2 A space-time moving finite element method with Runge-Kutta time integration

In this section, we describe how to apply the Runge-Kutta methods discussed in section 6.1 to the linear system

$$U_\tau(t) = \mathcal{M}^{-1}(t)\big[F(t) + G(t) - \mathcal{S}(t)U(t)\big] \equiv \Phi(t, U(t)),$$

which is equivalent to (5.9), derived in section 5.3, except that we impose this system for $0 < t \leq T$ instead of only the discrete collocation nodes. Applying the Runge-Kutta coefficients to this system of ordinary differential equations, we get the discrete problem

$$\bar{\delta}_t U(t_{i,j}) = \Delta t_i \mathcal{M}^{-1}(t_{i,j})\big[F(t_{i,j}) + G(t_{i,j}) - \mathcal{S}(t_{i,j})U_{RK}(t_{i,j})\big].$$

By reverse engineering, this linear system can be shown to correspond to the constraint

$$\big(\bar{\delta}_\tau \bar{u}(t_{i,j}), \chi\big) + \mathcal{A}_\tau\big(\bar{u}_{RK}, \chi; t_{i,j}\big) = \big(f(t_{i,j}), \chi\big) + \big\langle g(t_{i,j}), \chi\big\rangle$$

at each collocation node, where $\bar{\delta}_\tau$ signifies the derivative approximation along the characteristic trajectories of the mesh and the approximate solution, $\bar{u}$ in $\mathcal{V}_h^p$, is determined by the vectors of basis coefficients, $U(\zeta_{i,j})$, $1 \leq j \leq p$ and $1 \leq i \leq m$. This method is formalized in formulation 5.

Formulation 5. *Let a, b, c, and f be smooth and bounded functions on $\mathcal{F}$ satisfying $a \geq \bar{a} > 0$ and $c \geq \bar{c} \geq 0$, and let $g(t)$ be in $\mathcal{L}_2(\partial\Omega)$ for $0 < t \leq T$. Given the mesh velocity, x_t, and collocation nodes, $\{t_{i,j}\}$, find u_h in $\mathcal{V}_h^p$ such that for each $t_{i,j}$ and all χ in $\mathcal{V}_h^p(t_{i,j})$, the finite element solution satisfies*

$$\big(\bar{\delta}_\tau \bar{u}(t_{i,j}), \chi\big) + \mathcal{A}_\tau\big(\bar{u}_{RK}, \chi; t_{i,j}\big) = \big(f(t_{i,j}), \chi\big) + \big\langle g(t_{i,j}), \chi\big\rangle \tag{6.5}$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, p$, and when $t = 0$,

$$\big(\bar{u}_{RK}(\cdot, 0), \chi\big) = \big(u_0, \chi\big).$$

As before, the constraint $([\bar{u}](t_i), \chi) = 0$ must hold for all χ in $\mathcal{V}_h^p(t_{i^+})$, $i = 1, \ldots, m$, to ensure that $\bar{u} \in \mathcal{V}_h^p$.

6.2.1 The importance of stability

It is commented that the system of ordinary differential equations arising as the semi-discrete formulation may correspond to a very stiff problem ([57],[55], [**?**]). Accordingly, it is important to choose a stable time integration scheme to ensure that the qualitative behavior of the computed solution matches that of the true solution. An A-stable time integration method implies that the computed solution decays in time for any sized time step when $c > 0$ in the differential equation [44]. However, the computed solution may still display some artificial oscillations in time that are not a reflection of the true solution's behavior when the time step is too large. To avoid this undesirable behavior, L-stable schemes should be used, as they dampen such numerically-induced "ringing" [23].

When $p = 1$, backward Euler time stepping possesses both A-stability and L-stability; when $p = 2$, TR-BDF and second difference backward Euler serve as stable methods, where TR-BDF also benefits from its diagonally implicitness.

For higher order methods, implicit backward difference methods possess favorable stability properties ([44],[62]), though this comes at the cost of greater CPU time when numerically solving the differential equation.

6.3 The associated linear system and well-posedness

Formulation 5 is a generalization of the finite element formulation given in chapter 5. To prove that the Runge-Kutta formulation admits a unique solution, an analysis following the one presented in section 5.3 is conducted. Since the current formulation 5 does not alter the spatial discretization, the only matrices that must be modified are Λ and Λ', defined in (5.10). These matrices correspond to evaluations of the time basis functions at the collocation nodes, which are now replaced by the Runge-Kutta coefficients; we define the reduced $p \times p$ coefficient matrices

$$\mathcal{W} = \left[\omega_{j,\ell}\right]_{(j,\ell)} \quad \text{and} \quad \mathcal{Z} = \left[\xi_{j,\ell}\right]_{(j,\ell)},$$

with $1 \leq j, \ell \leq p$. The matrices $\mathcal{W}$ and $\mathcal{Z}$ replace the L' and L matrices of section 5.3, respectively. The corresponding $(pN_i) \times (pN_i)$ block matrices used to set up the linear system are then given by

$$\Omega = \left[\omega_{j,\ell} I_{N_i}\right]_{(j,\ell)} \quad \text{and} \quad Z = \left[\xi_{j,\ell} I_{N_i}\right]_{(j,\ell)}.$$

Following the arguments of section 5.3, the linear system that is solved to find the basis coefficients for u_{RK} on the i^{th} time partition is

$$\left[\frac{1}{\Delta t_i} M_i \Omega + S_i Z\right] U_i = F_i + G_i - \left[\frac{1}{\Delta t_i} M_i \Omega_0 + S_i Z_0\right] U(t_{i-1^+}). \tag{6.6}$$

where the right side vectors, $U(t_{i-1^+})$, F_i, and G_i, are given as before, and Ω_0 and Z_0 are the $(pN_i) \times N_i$ block matrices

$$\Omega_0 = \left[\omega_{j,0} I_{N_i}\right]_j \quad \text{and} \quad Z_0 = \left[\xi_{j,0} I_{N_i}\right]_j,$$

for $1 \leq j \leq p$. As before, the solution's coefficient vector U_i provides the basis coefficients for the solution at the time basis nodes, $\zeta_{i,j}$.

Ultimately, the condition that must be satisfied is that the mesh is shape regular at the collocation nodes, $\{t_{i,j}\}$, so that the mass matrix at each collocation node is invertible, and the matrix of Runge-Kutta derivative coefficients, Ω, must be also be nonsingular — recall that Ω is nonsingular if and only if the $p \times p$ matrix $\mathcal{W}$ is nonsingular, following from a simple row reduction argument. If these two conditions are met, then the coefficient matrix $A_{RK,i} = \frac{1}{\Delta t_i} M_i \Omega + S_i Z$ in the linear system is nonsingular and, therefore, when $\Delta t_i > 0$ is sufficiently small, the coefficient matrix is invertible.

It is interesting to note that Δt_i can give a solution for sufficiently small for any choice of ordered basis nodes $\zeta_{i,k}$ and coefficients $\xi_{j,k}$, so long as the coefficients $\omega_{j,k}$ lead to a nonsingular matrix. However, the parameters $\{\zeta_{i,k}, \xi_{j,k}\}$ play a major role in determining not only what the computed solution is, but also what "sufficiently small" means; this relates to the *stability* of the time stepping scheme, as it details how the choice of Runge-Kutta method can improve the conditioning of the linear system. Furthermore, certain choices of these parameters can decouple the linear system (6.6) into several smaller linear systems, which can lead to great savings in computation time. This is the case when the reduced matrices $\mathcal{W}$ and $\mathcal{Z}$ are lower triangular, corresponding to *diagonally implicit* Runge-Kutta methods, such as TR-BDF.

6.3.1 Well-posedness of the TR-BDF scheme

Checking that the formulation based on TR-BDF time stepping is well-posed, we merely set up the matrices of Runge-Kutta coefficients and check for nonsingularity. The matrices are given by

$$\mathcal{W} = \begin{bmatrix} \frac{1}{\varepsilon} & 0 \\ -\frac{1}{\varepsilon(1-\varepsilon)} & \frac{2-\varepsilon}{1-\varepsilon} \end{bmatrix}$$

and

$$\mathcal{Z} = \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & 1 \end{bmatrix},$$

which are clearly nonsingular as they are lower triangular, when $0 < \varepsilon < 1$. Consequently, if Δt is sufficiently small, then we are guaranteed a that $\bar{u}$ exists.

Furthermore, the lower triangular structure implies that the solution at the mid-step of each time partition, $\bar{u}(t_{i-1} + \varepsilon \Delta t_i)$, can be computed independently of the solution at the end of the time step, $\bar{u}(t_{i^-})$, which implies that two linear systems of size N_i can be solved on each partition instead of solving a single system of size $2N_i$, where N_i indicates the number of spatial nodes in $\mathcal{T}_{h,i}^p$. For fine meshes, N_i is large and sequentially solving for the finite element solution leads to significant savings in computation time.

6.4 Error analysis for TR-BDF

In this section, we prove an error estimate for the TR-BDF time integration scheme. The proof follows that of theorem 2 with some additional arguments that bound the error introduced by the trapezoid approximation at the mid-step of each time partition. Due to the departure of this method from a finite element framework, the symmetry of the error bound is broken and an additional term, proportional to the error of the trapezoid approximation of the true solution, is introduced to the bounding quantity. Since the trapezoid approximation is substituted into the bilinear form, $\mathcal{A}_\tau(\cdot, \cdot)$, the diffusion coefficient must have a bounded characteristic derivative,

$$\|a_\tau\|_\infty \leq \alpha,$$

for some $\alpha > 0$.

One final aspect of the error bound that comes into play is that this approach is more sensitive to the discontinuous changes in the mesh at the beginning of the time steps. Recently, Bank and Yserentant [22] have proven the $\mathcal{H}^1$-stability of $\mathcal{L}_2$-projections onto finite element spaces with potentially nonuniform meshes. Making use of this result, it holds that $|\chi(t_{i^+})|_1 \leq C_{\mathcal{H}} |\chi(t_{i^-})|_1$, for χ in $\mathcal{V}_h^p$. As can be seen from [22], the bounding constant, $C_{\mathcal{H}}$, is smaller when the mesh $\mathcal{T}_h^p(t_{i^+})$ is close to $\mathcal{T}_h^p(t_{i^-})$. This intuitively makes sense, since $\chi(t_{i^+}) \approx \chi(t_{i^-})$ in such cases. For a given differential equation, since $0 < \bar{a} \leq a \leq \|a\|_\infty$, we have an equivalence

of norms

$$\frac{1}{c_{\mathcal{A}}}\|a^{1/2}(t_i)\nabla\phi\|_0 \leq |\phi|_1 \leq c_{\mathcal{A}}\|a^{1/2}(t_i)\nabla\phi\|_0$$

for some positive $c_{\mathcal{A}}$. We define the stability constant for the diffusion-weighted semi-norm by

$$|a^{1/2}(t_i)\nabla\phi(t_{i^+})|_0^2 \leq C_{\mathcal{A},\mathcal{H}}|a^{1/2}(t_i)\nabla\phi(t_{i^-})|_0^2. \tag{6.7}$$

The norm in which the error is bounded employs the trapezoid approximation at the mid-step collocation node of each time partition. Let $u \in \mathcal{V}$ and $u_{TR}(t_{i,1}) \equiv \frac{\tilde{u}(\zeta_{i,1})+\tilde{u}(\zeta_{i,0})}{2}$, where $\tilde{u}(t)$ represents u evaluated at time t following the characteristic from time $t_{i,1}$. (Note that $\tilde{u}(t)$ therefore represents the function u shifted onto the mesh at time t.) The semi-norm in which we bound the error of the finite element function determined by formulation 5 with TR-BDF time stepping is

$$\|\!|u|\!\|_{TR-BDF}^2 \equiv \max_{\substack{1\leq i\leq m\\ 1\leq j\leq p}} \|u(t_{i,j})\|_0^2 + \mathcal{Q}\big(\|\bar{\delta}_\tau u\|_{(-1,\mathcal{V}_h^p(\cdot))}^2 + \|u_{RK}\|_1^2\big) + \sum_{i=1}^{m}\Delta t_i\|u(\zeta_{i,1})\|_1^2,$$

where $u_{RK}(t_{i,1}) = u_{TR}(t_{i,1})$ and $u_{RK}(t_{i,2}) = u(t_{i,2})$.

Theorem 3. *Suppose that $\mathcal{V}_h^p$ is a finite element space with a non-degenerate mesh and let $\hat{\mathcal{Q}}$ correspond to the Gauss-Radau quadrature rule on the unit interval. Furthermore, assume that there exist positive constants μ, κ, and α such that at each collocation node*

$$\|a_\tau\|_\infty \leq \alpha, \tag{6.8}$$

$$\rho\Big(\mathcal{H}_e(t_{i,j})\Big) \leq \mu, \tag{6.9}$$

$$\|b - x_t\|_\infty \leq \kappa, \tag{6.10}$$

and that the mesh discontinuities are controlled and the spatial meshes and length of the time steps are graded so that

$$C_{\mathcal{A},\mathcal{H}} \leq 7/2 \tag{6.11}$$

and

$$\Delta t_i \leq 2\Delta t_{i-1}. \tag{6.12}$$

Then, if $\Delta t = \max_{1\le i\le m}\Delta t_i$ *is sufficiently small, there exists a positive constant* C *such that the finite element solution defined by formulation 4 satisfies*

$$\|\!|u-\bar{u}|\!\|^2_{TR-BDF} \le C\Big\{\inf_{\chi\in\mathcal{V}_h^p}\|\!|u-\chi|\!\|^2_{TR-BDF} + \int_0^T \|\Delta t_i^2 u_{\tau\tau}(t)\|_1^2\, dt\Big\}, \tag{6.13}$$

where C *depends on* μ,κ,d,p*, and the differential equation.*

Note that this proof is restricted to the case where we use the collocation nodes determined by Gauss-Radau quadrature; this is necessary for the proof as it maintains two important properties. First, Gauss-Radau quadrature is non-truncating, which is needed to avoid a buildup of the quadrature errors. And secondly, Gauss-Radau fixes a collocation node at the end of the time steps, which is required for the TR-BDF scheme. This is important for creating a telescoping sum that allows us to avoid a buildup of the approximation error of the trapezoid rule within each partition. Notice that the intermediate time basis node for Gauss-Radau is $2/3$, which is close to the optimal value $\varepsilon = 2-\sqrt{2}\approx 0.58578$ for TR-BDF. Furthermore, the assumption (6.11) and (6.12) lead to a cleaner proof, though they are slightly stricter than necessary.

Proof. For this proof, we use the discrete Galerkin orthogonalities

$$(u_\tau - \bar{u}_\tau,\chi) + \mathcal{A}_\tau(u-\bar{u}_{RK},\chi) = 0,$$

for χ in $\mathcal{V}_h^p(t)$ at $t=t_{i,1}$ and $t_{i,2}$, respectively, for $i=1,\dots,m$. As in the proof for theorem 2, let ψ in $\mathcal{V}_h^p$ be an arbitrary function and define $\phi=\bar{u}-\psi$ and $\eta = u-\psi$. Then, we have

$$(\phi_\tau,\chi)+\mathcal{A}_\tau(\phi_{i,TR},\chi;t_{i,1}) = (\eta_\tau,\chi)+\mathcal{A}_\tau(\eta_{i,TR},\chi;t_{i,1})+\mathcal{A}_\tau\big(u(t_{i,1})-u_{i,TR},\chi;t_{i,1}\big) \tag{6.14}$$

at $t=t_{i,1}$ for χ in $\mathcal{V}_h^p(t_{i,1})$ and

$$(\phi_\tau,\hat{\chi})+\mathcal{A}_\tau(\phi,\hat{\chi};t_{i,2}) = (\eta_\tau,\hat{\chi})+\mathcal{A}_\tau\big(\eta,\hat{\chi};t_{i,2}\big) \tag{6.15}$$

at $t=t_{i,2}$ for $\hat{\chi}$ in $\mathcal{V}_h^p(t_{i,2})$.

We begin with bounding ϕ at the end step (6.15), which follows the proof of theorem 2 exactly. We choose $\hat{\chi} = \phi(t_{i^-})$ to get

$$(\phi_\tau, \phi) = \frac{1}{2}\partial_\tau \|\phi(t_{i^-})\|_0^2, \tag{6.16}$$

$$\mathcal{A}_\tau(\phi, \phi; t_{i,2}) \geq (1-\delta)\|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2 - C_\kappa \|\phi(t_{i^-})\|_0^2, \tag{6.17}$$

$$(\eta_\tau, \phi) \leq C\Big(\|\eta_\tau\|^2_{(-1,\mathcal{V}_h^p(t_{i^-}))} + \|\phi(t_{i^-})\|_0^2\Big) + \epsilon\|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2, \tag{6.18}$$

$$\mathcal{A}_\tau\big(\eta, \phi; t_{i,2}\big) \leq C_\kappa\Big(\|\eta\|_1^2 + \|\phi(t_{i^-})\|_0^2\Big) + \epsilon\|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2, \tag{6.19}$$

where δ and ϵ are assumed to be small positive constants. Combining (6.16)–(6.19) gives the bound

$$\begin{aligned}\frac{1}{2}\partial_\tau\|\phi(t_{i^-})\|_0^2 + (1-\epsilon)\|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2 \\ \leq C\big\{\|\eta_\tau(t_{i^-})\|^2_{(-1,\mathcal{V}_h^p(t_{i^-}))} + \|\eta(t_{i^-})\|_1^2 + \|\phi(t_{i^-})\|_0^2\big\},\end{aligned} \tag{6.20}$$

where this ϵ is not the same constant as in the bounds (6.18)–(6.19), but can still be made arbitrarily small at the cost of increasing the bounding constant on the right side.

The bound at the mid-step collocation node follows from a new choice of test function; define χ in (6.14) to be

$$\chi = \phi_{i,TR} + \frac{\Delta t_i}{24}\phi_\tau(t_{i,1}) = \phi(t_{i,1}) + \frac{\Delta t_i}{24}\phi_\tau(t_{i,1}) + \frac{\Delta t_i^2}{18}\phi_{i,\tau\tau},$$

where $\frac{\Delta t_i^2}{18}\phi_{i,\tau\tau}$ is merely the error of the trapezoid approximation. The term $\frac{1}{24}\Delta t_i\phi_\tau(t_{i,1})$ is not an intuitive addition to the test function; however, it will lend itself to perfectly offset the additional errors that arise front the trapezoid approximation of $\bar{u}$. Using this test function, we have

$$\begin{aligned}\Big(\phi_\tau(t_{i,1}), \phi(t_{i,1}) + \frac{\Delta t_i}{24}\phi_\tau(t_{i,1}) + \frac{\Delta t_i^2}{18}\phi_{i,\tau\tau}\Big) \\ \geq \frac{1}{2}\partial_\tau\|\phi(t_{i,1})\|_0^2 + \frac{\Delta t_i}{24}\|\phi_\tau(t_{i,1})\|_0^2 - \frac{\Delta t_i^2}{18}\big\|\phi_\tau(t_{i,1})\big\|_0\big\|\phi_{i,\tau\tau}\big\|_0 \\ \geq \frac{1}{2}\partial_\tau\|\phi(t_{i,1})\|_0^2 + \frac{\Delta t_i}{24}\|\phi_\tau(t_{i,1})\|_0^2 - \left(\frac{\Delta t_i}{24}\|\phi_\tau(t_{i,1})\|_0^2 + \frac{\Delta t_i^{-1}}{54}\|\Delta t_i^2\phi_{i,\tau\tau}\|_0^2\right) \\ \geq \frac{1}{2}\partial_\tau\|\phi(t_{i,1})\|_0^2 - \frac{\Delta t_i^{-1}}{54}\|\Delta t_i^2\phi_{i,\tau\tau}\|_0^2.\end{aligned} \tag{6.21}$$

For the bilinear form, we bound this in two parts:

$$\mathcal{A}_\tau\big(\phi_{i,TR}, \phi_{i,TR}; t_{i,1}\big) \geq (1-\delta')\|a^{1/2}(t_{i,1})\nabla\phi_{i,TR}\|_0^2 - C_\kappa\|\phi_{i,TR}\|_0^2 \tag{6.22}$$

and

$$\begin{aligned}
&\mathcal{A}_\tau\big(\phi_{i,TR}, \Delta t_i\phi_\tau(t_{i,1})/24; t_{i,1}\big) \\
&\qquad = \frac{1}{24}\mathcal{A}_\tau\bigg(\frac{\tilde\phi(\zeta_{i,1}) + \tilde\phi(\zeta_{i-1^+})}{2}, 3\frac{\tilde\phi(\zeta_{i,1}) - \tilde\phi(\zeta_{i-1^+})}{2}; t_{i,1}\bigg) \\
&\geq \frac{1}{32}\Big\{(1-\tilde\delta')\|a^{1/2}(t_{i,1})\nabla\tilde\phi(\zeta_{i,1})\|_0^2 - (1+\tilde\delta)\|a^{1/2}(t_{i,1})\nabla\tilde\phi(t_{i-1^+})\|_0^2\Big\} - C_\kappa\|\Delta t_i\phi_\tau\|_0^2 \\
&\geq \frac{1-\tilde\delta'}{32}\|a^{1/2}(t_{i,1})\nabla\tilde\phi(\zeta_{i,1})\|_0^2 - \frac{1+\tilde\delta}{32}\|a^{1/2}(t_{i,1})\nabla\tilde\phi(t_{i-1^+})\|_0^2 - C_\kappa\|\Delta t_i\phi_\tau\|_0^2,
\end{aligned} \tag{6.23}$$

where $\delta', \tilde\delta, \tilde\delta' > 0$ are all arbitrarily small. Bounds on the right side follow the usual arguments. For arbitrarily small $\epsilon > 0$, we have

$$\begin{aligned}
(\eta_\tau, \phi_{i,TR} + \Delta t_i\phi_\tau/24) &\leq C\Big(\|\eta_\tau(t_{i,1})\|^2_{(-1,\mathcal{V}_h^p(t_{i,1}))} + \big\|\phi_{i,TR}\big\|_0^2 + \big\|\Delta t_i\phi_\tau(t_{i,1})\big\|_0^2\Big) \\
&\quad + \epsilon\big\{\|a^{1/2}(t_{i,1})\nabla\phi_{i,TR}\|_0^2 + \|a^{1/2}(t_{i,1})\nabla\tilde\phi(\zeta_{i,1})\|_0^2 + \|a^{1/2}(t_{i,1})\nabla\tilde\phi(\zeta_{i,1})\|_0^2\big\},
\end{aligned} \tag{6.24}$$

$$\begin{aligned}
\mathcal{A}_\tau\big(\eta_{i,TR}, \phi_{i,TR} + \Delta t_i\phi_\tau/24; t_{i,1}\big) &\leq C_\kappa\Big(\|\eta_{i,TR}\|_1^2 + \big\|\phi_{i,TR}\big\|_0^2 + \big\|\Delta t_i\phi_\tau(t_{i,1})\big\|_0^2\Big) \\
&\quad + \epsilon\big\{\|a^{1/2}(t_{i,1})\nabla\phi_{i,TR}\|_0^2 + \|a^{1/2}(t_{i,1})\nabla\tilde\phi(\zeta_{i,1})\|_0^2 + \|a^{1/2}(t_{i,1})\nabla\tilde\phi(\zeta_{i,1})\|_0^2\big\},
\end{aligned} \tag{6.25}$$

and

$$\begin{aligned}
&\mathcal{A}_\tau\big(u(t_{i,1}) - u_{i,TR}, \phi_{i,TR} + \Delta t_i\phi_\tau/24; t_{i,1}\big) \\
&\qquad\qquad \leq C_\kappa\Big(\|u(t_{i,1}) - u_{i,TR}\|_1^2 + \big\|\phi_{i,TR}\big\|_0^2 + \big\|\Delta t_i\phi_\tau(t_{i,1})\big\|_0^2\Big) \\
&\quad + \epsilon\big\{\|a^{1/2}(t_{i,1})\nabla\phi_{i,TR}\|_0^2 + \|a^{1/2}(t_{i,1})\nabla\tilde\phi(\zeta_{i,1})\|_0^2 + \|a^{1/2}(t_{i,1})\nabla\tilde\phi(\zeta_{i,1})\|_0^2\big\}.
\end{aligned} \tag{6.26}$$

Hence, at the mid-step collocation node corresponding to the trapezoid step of the TR-BDF rule, the following bound follows from (6.21)–(6.26):

$$\begin{aligned}
&\frac{1}{2}\partial_\tau\|\phi(t_{i,1})\|_0^2 - \frac{\Delta t_i^{-1}}{54}\|\Delta t_i^2\phi_{i,\tau\tau}\|_0^2 + (1-\epsilon')\|a^{1/2}(t_{i,1})\nabla\phi_{i,TR}\|_0^2 \\
&\qquad + \frac{1-\tilde\epsilon'}{32}\|a^{1/2}(t_{i,1})\nabla\tilde\phi(\zeta_{i,1})\|_0^2 - \frac{1+\tilde\epsilon}{32}\|a^{1/2}(t_{i,1})\nabla\tilde\phi(t_{i-1^+})\|_0^2 \\
&\qquad\qquad \leq C\Big\{\|\eta_\tau(t_{i,1})\|^2_{(-1,\mathcal{V}_h^p(t_{i,1}))} + \|\eta_{i,TR}\|_1^2 + \|u(t_{i,1}) - u_{i,TR}\|_1^2 \\
&\qquad\qquad\qquad\qquad + \big\|\phi_{i,TR}\big\|_0^2 + \big\|\Delta t_i\phi_\tau(t_{i,1})\big\|_0^2\Big\}.
\end{aligned} \tag{6.27}$$

Thus, we have the spatial bounds at each collocation node. We now turn to bound the error introduced by the time stepping scheme. Since we are using Gauss-Radau quadrature, we have

$$\hat{\mathcal{Q}}(f) = \frac{3}{4}f(1/3) + \frac{1}{4}f(1) = \int_0^1 f(t)\,dt + \frac{1}{6^3}f'''(\zeta),$$

for some ζ in $[0,1]$ and any bounded function f on $[0,1]$. For the characteristic derivative terms in (6.20) and (6.27), applying lemma 2 and this quadrature rule over the time partition gives

$$\begin{aligned}
\frac{3\Delta t_i}{4}\partial_\tau\|\phi(t_{i,1})\|_0^2 + \frac{\Delta t_i}{4}\partial_\tau\|\phi(t_{i,2})\|_0^2 &\geq \frac{3\Delta t_i}{4}\frac{d}{dt}\|\tilde{\phi}(t_{i,1})\|_0^2 + \frac{\Delta t_i}{4}\frac{d}{dt}\|\tilde{\phi}(t_{i,2})\|_0^2 \\
&\quad - C_{\mu,d}\Delta t_i\Big\{\|\Delta t_i\tilde{\phi}_t(t_{i,1})\|_0\|\tilde{\phi}(t_{i,1})\|_0 + \|\Delta t_i\tilde{\phi}_t(t_{i,2})\|_0\|\tilde{\phi}(t_{i,2})\|_0\Big\} \\
&= \|\tilde{\phi}(t_{i^-})\|_0^2 - \|\tilde{\phi}(t_{i-1^+})\|_0^2 + \frac{\Delta t_i^4}{6^3}\left(\frac{d}{dt}\right)^4\|\tilde{\phi}\|_0^2 - C\Delta t_i\max_{0\leq j\leq 2}\|\tilde{\phi}(t_{i,j})\|_0^2 \\
&\geq \|\phi(t_{i^-})\|_0^2 - \|\phi(t_{i-1^+})\|_0^2 + \frac{\Delta t_i^4}{6^3}\partial_\tau^4\|\phi\|_0^2 - \hat{C}\Delta t_i\max_{0\leq j\leq 2}\|\phi(t_{i,j})\|_0^2.
\end{aligned}$$

Since $\partial_\tau^3\phi \equiv 0$, we have $\Delta t_i^4\partial_\tau^4\|\phi\|_0^2 = 6\|\Delta t_i^2\phi_{i,\tau\tau}\|_0^2$. Hence, applying the quadrature rule to the characteristic derivative gives

$$\begin{aligned}
&\frac{3\Delta t_i}{4}\partial_\tau\|\phi(t_{i,1})\|_0^2 + \frac{\Delta t_i}{4}\partial_\tau\|\phi(t_{i,2})\|_0^2 \\
&\qquad \geq \|\phi(t_{i^-})\|_0^2 - \|\phi(t_{i-1^-})\|_0^2 + \frac{1}{36}\|\Delta t_i^2\phi_{i,\tau\tau}\|_0^2 - \hat{C}\Delta t_i\max_{0\leq j\leq 2}\|\phi(t_{i,j})\|_0^2, \quad (6.28)
\end{aligned}$$

where we used $\|\phi(t_{i-1^-})\|_0^2 \geq \|\phi(t_{i-1^+})\|_0^2$. The quadrature rule applies to the additional terms in the test function at time $t_{i,1}$, which we combine with (6.28) to attain to the bound

$$\begin{aligned}
&\frac{3\Delta t_i}{4}\frac{1}{2}\partial_\tau\|\phi(t_{i,1})\|_0^2 + \frac{\Delta t_i}{4}\frac{1}{2}\partial_\tau\|\phi(t_{i,2})\|_0^2 - \frac{3\Delta t_i}{4}\frac{\Delta t_i^{-1}}{54}\|\Delta t_i^2\phi_{i,\tau\tau}\|_0^2 \\
&\geq \frac{\|\phi(t_{i^-})\|_0^2 - \|\phi(t_{i-1^-})\|_0^2}{2} + \Big(\frac{1}{72} - \frac{1}{72}\Big)\|\Delta t_i^2\phi_{i,\tau\tau}\|_0^2 - \hat{C}\Delta t_i\max_{0\leq j\leq 2}\|\phi(t_{i,j})\|_0^2. \quad (6.29)
\end{aligned}$$

Accordingly, we have the bound

$$\begin{aligned}
&\|\phi(t_{i^-})\|_0^2 - \|\phi(t_{i-1^-})\|_0^2 \\
&\qquad + \frac{\Delta t_i}{2}\Big\{3(1-\epsilon')\|a^{1/2}(t_{i,1})\nabla\phi_{i,TR}\|_0^2 + (1-\epsilon)\|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2 \\
&\qquad + \frac{3(1-\tilde{\epsilon}')}{32}\|a^{1/2}(t_{i,1})\nabla\tilde{\phi}(\zeta_{i,1})\|_0^2 - \frac{3(1+\tilde{\epsilon})}{32}\|a^{1/2}(t_{i,1})\nabla\tilde{\phi}(t_{i-1^+})\|_0^2\Big\} \\
&\le C\Delta t_i\mathcal{Q}_i\big(\|\eta_\tau(\cdot)\|^2_{(-1,\mathcal{V}_h^p(\cdot))} + \|\eta_{RK}(\cdot)\|_1^2 + \|\phi(\cdot)\|_0^2\big) + C\Delta t_i\|u(t_{i,1}) - u_{TR}(t_{i,1})\|_1^2.
\end{aligned} \tag{6.30}$$

The term preventing us from applying the discrete Grönwall lemma, as in the proof of theorem 2, is $-\frac{3(1+\tilde{\epsilon})}{32}\|a^{1/2}(t_{i,1})\nabla\tilde{\phi}(t_{i-1^+})\|_0^2$ on the left side. Our approach for getting rid of this term is to create a telescoping sum once we apply the discrete Grönwall lemma. From (6.8) and lemma 2, and the $\mathcal{H}^1$-stability of $\mathcal{L}_2$-projection [22], we have

$$\begin{aligned}
&\frac{3\Delta t_i}{32}(1+\tilde{\epsilon})\|a^{1/2}(t_{i,1})\nabla\tilde{\phi}(t_{i-1^+})\|_0^2 \\
&\qquad\qquad \le \frac{3\Delta t_i}{32}(1+\tilde{\epsilon})(1+\alpha\Delta t_i)\|\tilde{a}^{1/2}(t_{i-1})\nabla\tilde{\phi}(t_{i-1^+})\|_0^2 \\
&\qquad \le \frac{3\Delta t_i}{32}(1+\tilde{\epsilon})(1+\alpha\Delta t_i)(1+C_{\mu,d}\Delta t_i)\|a^{1/2}(t_{i-1})\nabla\phi(t_{i-1^+})\|_0^2 \\
&\quad \le \frac{3C_{\mathcal{A},\mathcal{H}}\Delta t_i}{32}(1+\tilde{\epsilon})(1+\alpha\Delta t_i)(1+C_{\mu,d}\Delta t_i)\|a^{1/2}(t_{i-1})\nabla\phi(t_{i-1^-})\|_0^2.
\end{aligned} \tag{6.31}$$

If we choose $\tilde{\epsilon} \le \frac{1}{3}$, then the controlled mesh discontinuities (6.11) and graded time stepping (6.12) gives $3C_{\mathcal{A},\mathcal{H}}\Delta t_i/32 \le 7\Delta t_{i-1}/8$. Choose $\epsilon = 1/16$ so that

$$\begin{aligned}
&\frac{15\Delta t_i}{16}\|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2 - \frac{7\Delta t_{i-1}}{8}\|a^{1/2}(t_{i,1})\nabla\tilde{\phi}(t_{i-1^+})\|_0^2 \\
&\quad \ge \frac{7}{8}\Big\{\Delta t_i\|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2 - (1+C_{\alpha,\mu,d}\Delta t_i)\Delta t_{i-1}\|a^{1/2}(t_{i-1})\nabla\phi(t_{i-1^-})\|_0^2\Big\} \\
&\qquad\qquad + \frac{\Delta t_i}{16}\|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2.
\end{aligned} \tag{6.32}$$

Then, we use $a \geq \bar{a}$ and lemma 2 to show that the bound for the i^{th} partition is

$$\begin{aligned}
&\left[\|\phi(t_{i^-})\|_0^2 + \frac{7}{16}\Delta t_i \|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2\right] \\
&\qquad - (1 + C'\Delta t_i)\left[\|\phi(t_{i-1^-})\|_0^2 + \frac{7}{16}\Delta t_{i-1}\|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2\right] \\
&\qquad + \frac{\bar{a}\Delta t_i}{2}\left\{3(1-\epsilon')\|\phi_{i,TR}\|_1^2 + \frac{1}{16}\|\phi(t_{i^-})\|_1^2 + \frac{3(1-\tilde{\epsilon}')}{32}\|\phi(\zeta_{i,1})\|_1^2\right\} \\
&\leq C\Delta t_i\Big\{\mathcal{Q}_i\Big(\|\eta_\tau(\cdot)\|^2_{(-1,\mathcal{V}_h^p(\cdot))} + \|\eta_{RK}(\cdot)\|_1^2\Big) + \max_{0\leq j\leq 2}\|\phi(t_{i,j})\|_0^2 + \|u(t_{i,1}) - u_{TR}(t_{i,1})\|_1^2\Big\}.
\end{aligned} \tag{6.33}$$

From the local Grönwall inequality, we can get the bound

$$\begin{aligned}
&\left[\|\phi(t_{i^-})\|_0^2 + \frac{7}{16}\Delta t_i \|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2\right] \\
&\qquad - (1 + C'\Delta t_i)\left[\|\phi(t_{i-1^-})\|_0^2 + \frac{7}{16}\Delta t_{i-1}\|a^{1/2}(t_i)\nabla\phi(t_{i^-})\|_0^2\right] \\
&\qquad + \theta\Delta t_i\left\{\|\phi_{i,TR}\|_1^2 + \|\phi(t_{i^-})\|_1^2 + \|\phi(\zeta_{i,1})\|_1^2\right\} \\
&\leq C\Delta t_i\mathcal{Q}_i\big(\|\eta_\tau(\cdot)\|^2_{(-1,\mathcal{V}_h^p(\cdot))} + \|\eta_{RK}(\cdot)\|_1^2\big) + C\Delta t_i\|u(t_{i,1}) - u_{TR}(t_{i,1})\|_1^2,
\end{aligned} \tag{6.34}$$

for some $\theta > 0$, if Δt_i is sufficiently small. This is now in the appropriate form to apply the discrete Grönwall lemma, where the error in the $\mathcal{H}^1$-norm arising from the trapezoid approximation is joined in with the telescoping terms from the characteristic derivative.

Applying the discrete Grönwall lemma gives

$$\begin{aligned}
&\max_{1\leq i\leq m}\|\phi(t_{i^-})\|_0^2 + \mathcal{Q}\big(\|\phi_{RK}(\cdot)\|_1^2\big) + \sum_{i=1}^m \Delta t_i\|\phi(\zeta_{i,1})\|_1^2 \\
&\leq C\left\{|\!|\!|\eta|\!|\!|_{TR-BDF}^2 + \sum_{i=1}^m \Delta t_i\|u(t_{i,1}) - u_{TR}(t_{i,1})\|_1^2 + \|\phi(0)\|_0^2 + \Delta t|\phi(0)|_1^2\right\}.
\end{aligned} \tag{6.35}$$

As in the proof of theorem 2, we have

$$\|\phi(0)\|_0 \leq \|\eta(0)\|_0 \leq |\!|\!|\eta|\!|\!|_{TR-BDF}, \tag{6.36}$$

$$\begin{aligned}
&\|\phi_\tau(t_{i,j})\|_{(-1,\mathcal{V}_h^p(t_{i,j}))} \\
&\quad \leq C\Big\{\|\eta_\tau(t_{i,j})\|_{(-1,\mathcal{V}_h^p(t_{i,j}))} + \|\eta_{RK}(t_{i,j})\|_1 + \|u(t_{i,j}) - u_{RK}(t_{i,j})\|_1 + \|\phi_{RK}(t_{i,j})\|_1\Big\}
\end{aligned} \tag{6.37}$$

for $j = 1, 2$, and use the local Grönwall lemma again to bound the maximum $\|\phi\|_0$ at the intermediate collocation nodes to get

$$\max_{\substack{1\le i\le m\\ 1\le j\le 2}} \|\phi(t_{i,j})\|_0^2 + \mathcal{Q}\Big(\|\phi_{RK}(\cdot)\|_1^2 + \|\phi_\tau(\cdot)\|^2_{(-1,\mathcal{V}_h^p(\cdot))}\Big) + \sum_{i=1}^m \Delta t_i \|\phi(\zeta_{i,1})\|_1^2$$
$$\le C\bigg\{ |\!|\!|\eta|\!|\!|^2_{TR-BDF} + \sum_{i=1}^m \Delta t_i \|u(t_{i,1}) - u_{TR}(t_{i,1})\|_1^2 + \Delta t |\phi(0)|_1^2 \bigg\}. \quad (6.38)$$

Since the trapezoid approximation is second order, we have

$$\Delta t_i \|u(t_{i,1}) - u_{TR}(t_{i,1})\|_1^2 \le C_{TR} \int_{t_{i-1}}^{t_i} \|\Delta t_i^2 u_{\tau\tau}(t)\, dt\|_1^2 \quad (6.39)$$

and by the $\mathcal{H}^1$-stability of $\mathcal{L}_2$-projection, we have

$$\Delta t |\phi(0)|_1^2 \le C\Delta t |\eta(0)|_1^2 \le C\Delta t \Big(|\eta_{1,TR}|_1^2 + |\eta(\zeta_{1,1})|_1^2 \Big) \le C |\!|\!|\eta|\!|\!|^2_{TR-BDF}. \quad (6.40)$$

Thus, combining (6.35)–(6.40), we have

$$|\!|\!|\phi|\!|\!|^2_{TR-BDF} \le C\bigg\{ |\!|\!|\eta|\!|\!|^2_{TR-BDF} + \int_0^T \|\Delta t_i^2 u_{\tau\tau}(t)\, dt\|_1^2 \bigg\},$$

as desired. □

As can be seen from this proof, the values of the Runge-Kutta coefficients and time basis nodes are used explicitly to prove the result. So far, attempts that rely on generic Runge-Kutta approximation error bounds have not been fruitful in providing a satisfactory space-time error bound. It is suspected that test functions of the form $\chi = \sum_{j=0}^p c_j \Delta t_i^j \partial_\tau^j \phi$ might lead to a more general result.

Chapter 6, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 7

Numerical Methods and Experiments with Moving Meshes

In this chapter, we report the results of some numerical experiments to evaluate the validity of the theory given in chapters 3–6. A solver for time-dependent linear convection-diffusion-reaction equations with a single dimensional space domain was written in C++ and used to conduct these numerical experiments. All experiments employ a piecewise quadratic tensor product finite element space as described in chapter 3, with $p = 2$ and $d = 1$, and TR-BDF time integration. Experiments have been constructed to measure the gains of using a moving mesh versus a static mesh. The costs and benefits of a simple moving mesh scheme, following the method of characteristics, is analyzed. Overall, these experiments are designed to evaluate whether moving meshes provide improved accuracy and efficiency over standard finite element methods with non-moving meshes. We also test the effects of using interpolation rather than $\mathcal{L}_2$-projection of the finite element solution across mesh discontinuities (as in equations 3.7 and 5.2). This leads to significant savings in CPU time and we verify that the accuracy of the computed solution is not necessarily degraded by this "shortcut." We also experiment by comparing the results of using TR-BDF with several different values for the mid-step time basis nodes, including values corresponding to Gauss-Radau and Richardson's TR-BDF. The intent of this experiment is to test whether non-truncating quadrature rules are required to see second order convergence of the solution with respect to the time discretization.

7.1 Two test problems

For these experiments, we test our methods on two problems, both of which live on the domain $\mathcal{F} = [-3, 3] \times (0, 1]$. The first problem is a convection dominated problem given by

$$u_{1,t}(x,t) - 0.01u_{1,xx}(x,t) + 3u_{1,x}(x,t) = f_1(x,t), \tag{7.1}$$

with f_1, the initial condition, and Neumann boundary condition chosen such that the solution is given by

$$u_1(x,t) = e^{-(x-3t)^2}.$$

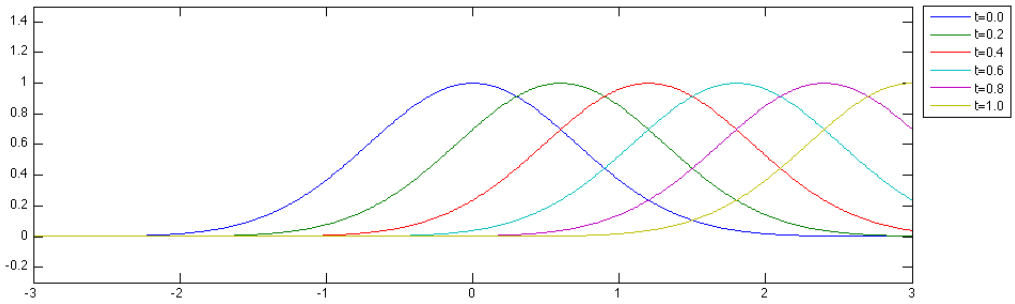

Figure 7.1: Above is the solution to the equation described in (7.1). We have a bell-curve initially centered at $x = 0$ that gradually shifts to the right as t increases. The solution is plotted at times $t = 0, .0.2, 0.4, 0.6, 0.8,$ and 1.

Due to the small diffusion coefficient and the null reaction term, this problem is not well-conditioned. The convection velocity in this problem follows the characteristic trajectories of the solution exactly and, accordingly, it is expected that the moving mesh can be a great benefit for discretization of this problem, as $u_\tau \equiv 0$ when $x_t = b$. Note that neither the Neumann boundary condition nor the partial differential equation describe the exact value of the solution, meaning that they only describe the solution up to an additive constant; the initial condition is the only given information that fixes the exact value of the solution. Figure 7.1 displays the solution plotted at discrete times.

The second problem used for experimentation is given by

$$u_{2,t}(x,t) - \left(\left(x^2 + t^2 + 0.1\right) u_{2,x}(x,t)\right)_x + 0.1(x^3 - 9x)u_{2,x}(x,t) + u_2(x,t) = f_2(x,t), \tag{7.2}$$

and the source term and boundary condition are chosen so that the solution of the differential equation is given by

$$u_2(x,t) = \sin\left(\frac{\pi}{6}(x + 5t)\right).$$

The solution is a traveling sine wave that is depicted in figure (7.2). The diffusion coefficient has a much stronger presence in this problem and the reaction term also helps to improve the condition of the linear system that determines the finite element solution. Notice that the characteristic directions of the solutions and the convection term are rather misaligned. According to the theory in chapters 5 and

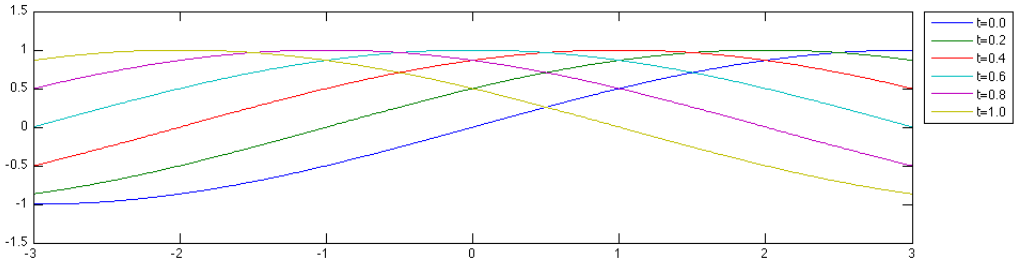

Figure 7.2: The solution to equation (7.2) is plotted at times $t = 0, .0.2, 0.4, 0.6, 0.8,$ and 1.

6, moving the mesh will improve the conditioning of the linear system and reduce the bounding constants of theorems 2 and 3, although moving the mesh along the characteristics will not help flow the mesh with the structures of the computed solution.

7.2 Moving meshes and static meshes

In this experiment, we use TR-BDF time stepping and set the mid-step time basis node at $\varepsilon = 2 - \sqrt{2}$ on the reference element, corresponding to Richardson's TR-BDF. At the mesh discontinuities, we use interpolation instead of $\mathcal{L}_2$-projection. Though these discretizations do not conform exactly to the theory assumed in chapters 3, 5, and 6, the experiments in sections 7.3 and 7.4 demonstrate that these modifications do not necessarily deteriorate the quality of the computed solution. Uniform time steps are taken and the mesh on each time partition is initialized with a uniform grid, with the spatial nodes moving to follow the convection term, as in the method of characteristics. To solve for the mesh velocity, we perform two steps of forward Euler, propagating the solution to the mid-step and end-step basis nodes, respectively, so that $x_t \approx b(x)$. Two examples of moving meshes are depicted in figures 7.3 and 7.4.

Problem (7.1) sees a great advantage when the mesh moves with the convection velocity, with the error seeing greater than a 300-fold improvement in the relative error on highly refined meshes with large time steps. The ratio of the final $\mathcal{L}_2$-errors at the end of the simulation and CPU time from the static mesh solutions

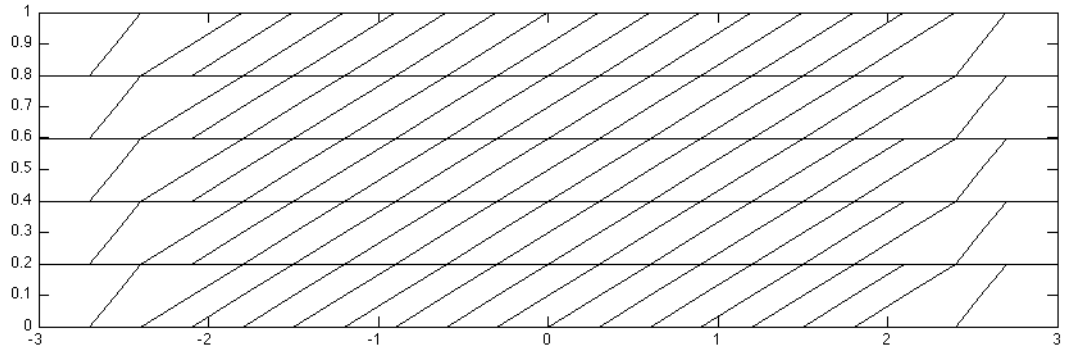

Figure 7.3: An example of a moving mesh given by the method of characteristics for equation (7.1). In this example, we have $m = 5$ time steps and initialize each time partition with $n = 11$ spatial nodes. As a result of using the method of characteristics, the spatial nodes on the interior of the domain satisfy $x_t = 3$; some spatial nodes have been deleted near the outflow boundary and inflow boundary. The hat nodes and the bump nodes are both displayed and the bump nodes are always chosen to be the midpoint of the element's hat nodes. The vertical axis corresponds to the time dimension and the horizontal axis corresponds to the spatial dimension.

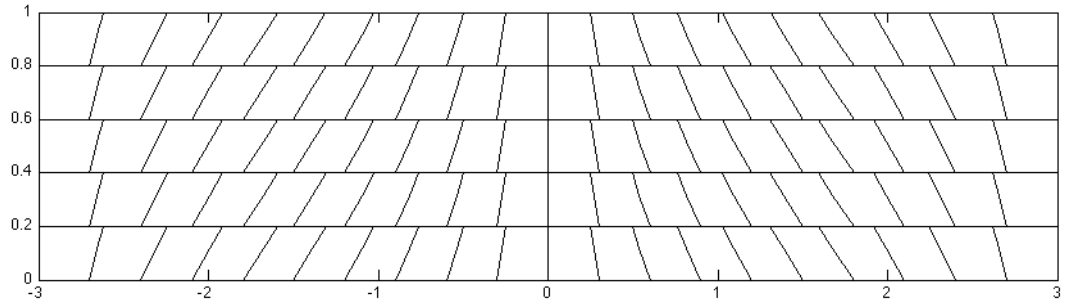

Figure 7.4: The mesh generated by the method of characteristics applied to (7.2). Though the curvature is subtle, these mesh trajectories are quadratic and approximately satisfy the evolution equation $x_t = 0.1(x^3 - 9x)$. In this example, we take $m = 5$ time steps and initialize each partition with $n = 11$ spatial nodes. The hat nodes and the bump nodes are both displayed and the bump nodes are always chosen to be the midpoint of the element's hat nodes. The vertical axis corresponds to the time dimension and the horizontal axis corresponds to the spatial dimension.

to the moving mesh solutions are reported in table 7.1. These numbers are relative to the solution computed using a static mesh; values less than 1 correspond to an decrease in the norm of the error or a speedup in the CPU time, when using moving meshes.

The greatest gains are realized when Δt is large relative to Δx, since the mesh is following the characteristic directions of the differential equation. Furthermore, the characteristic directions of the differential equation perfectly track the solution of this problem, which may contribute additional accuracy to the computed solution in the moving mesh cases.

It is important to note when Δt is relatively small compared to Δx, indicating many time steps and coarse meshes, that moving meshes lead to larger error; such behavior is the result of interpolating the computed solution across many coarse mesh discontinuities, causing a buildup of local interpolation errors. This buildup of errors disappears as the spatial mesh is refined and one would expect that it is not as pronounced if the grid at the beginning of each time partition were to be initialized by the mesh at the end of the previous time step, as interpolation will only be necessary for a few element deletions taking place near the boundaries. Investigations using adaptive meshes might also help and such experiments are carried out in 7.5.1, where h-adaptivity is used, and in [50], which investigates graph massage techniques. Note that moving meshes typically require more CPU time, as the mesh motion must be computed for each time partition. However, the motion of the nodes are independent of one another (except when checking for element degeneration), so this process scales well with the number of spatial nodes, and this is reflected in the relative change in the CPU time for larger n.

For the differential equation given in (7.2), a slight advantage is recognized by using moving meshes, when Δt is not too small relative to Δx. A comparison of the errors and CPU time is given in table 7.2. Comparing the results of this experiment to those of problem (7.1), it is clear that this problem does not benefit nearly as much from a moving mesh nor is the accuracy as dramatically impacted when the time discretization is much finer than the spatial discretization. It is suspected that mesh motion does not benefit this problem as much because the

convection velocity in the differential equation does not align very well with the structures of the solution, though small gains come from cancellation of the mesh and convection velocities.

The results of this experiment indicate that superior accuracy can be attained by moving the finite element mesh along with the convection term of the differential equation, especially in cases when the mesh trajectories follow the structures of the solution to the equation. One must be careful, however, that the spatial discretization is sufficiently refined relative to the time discretization in order to avoid a buildup of the errors incurred at the discrete mesh discontinuities. This suggests that a carefully crafted adaptive meshing scheme can be of great benefit to moving finite element schemes. Note that this experiment did not employ a non-truncating quadrature rule nor $\mathcal{L}_2$-projection at the mesh discontinuities. The effects of these modifications are studied in the following two experiments.

7.3 Non-truncating time integration and other collocation nodes

The experiment we describe in this section is designed to test the requirement of non-truncating quadrature rules to perform the time integration. Namely, we did not use non-truncating quadrature rules to advance the solution in time for our initial numerical experiments that encouraged the theoretical research of this thesis. Instead, stable second order time stepping methods were used and we observed satisfactory results. In this experiment, we choose the time basis nodes and the collocation nodes such that TR-BDF is used as the time stepping scheme: that is, we define the collocation nodes to be

$$\hat{t}_0 = 0, \quad \hat{t}_1 = \varepsilon/2, \quad \text{and} \quad \hat{t}_2 = 1,$$

and choose the basis nodes to be

$$\hat{\zeta}_0 = 0, \quad \hat{\zeta}_1 = \varepsilon, \quad \text{and} \quad \hat{\zeta}_2 = 1,$$

for $\varepsilon = 1/2, 2/3$, and $2 - \sqrt{2}$. The choice $\varepsilon = 1/2$ is tested here to evaluate the simplest choice of time discretization, where the mid-step basis node is chosen to be

Table 7.1: A comparison table for problem (7.1) between static mesh methods and moving mesh methods using interpolation. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative increase in CPU time of the moving mesh solution to the static mesh solution. Moving the mesh leads to clear advantages when the spatial mesh is sufficiently refined, though buildup in error occurs when the time discretization is too refined.

	$n = 51$		$n = 101$		$n = 501$		$n = 1001$		$n = 3001$	
m	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU
10	0.023152	1.055864	0.009600	1.018076	0.004912	1.008814	0.003609	1.007134	0.003064	1.014075
20	0.051109	1.046093	0.011640	1.046732	0.003174	1.043950	0.002690	1.040906	0.001831	1.047888
50	0.060497	1.083411	0.023513	1.066356	0.003185	1.061553	0.001900	1.059921	0.001645	1.072040
75	0.500296	1.075000	0.255983	1.067271	0.003120	1.059279	0.001619	0.575458	0.003136	1.062460
100	0.764073	1.103303	0.047134	1.072274	0.001862	1.064108	0.002876	0.520995	0.004224	1.069523
200	1.183061	1.073083	4.812077	1.068128	0.030458	1.069802	0.003648	1.062029	0.006529	1.064384
500	1.456289	1.076347	23.420443	1.073212	0.003740	1.067745	0.004751	1.062858	0.008277	1.069470
1000	1.552423	1.091603	30.122944	1.078852	3.814286	1.078889	0.004676	1.078591	0.008693	1.074301

Table 7.2: A comparison table for problem (7.2) between static mesh methods and moving mesh methods using interpolation. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative increase in CPU time of the moving mesh solution to the static mesh solution.

	$n = 51$		$n = 101$		$n = 501$		$n = 1001$		$n = 3001$	
m	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU
10	0.800877	1.056480	0.801248	1.072159	0.801366	1.073340	0.801370	1.082202	0.801371	1.069765
20	0.793183	1.068505	0.792684	1.068246	0.792828	1.075190	0.792831	1.052709	0.792832	1.070972
50	0.799117	1.065658	0.791186	1.069526	0.789285	1.072070	0.789292	1.068113	0.789292	1.072106
75	0.809921	1.063884	0.794616	1.069009	0.788798	1.071747	0.788809	1.070029	0.788806	1.084289
100	0.822145	1.070346	0.799659	1.070767	0.788575	1.071889	0.788627	1.060262	0.788628	1.065949
200	0.866538	1.095616	0.829764	1.070428	0.789477	1.082275	0.788421	1.070758	0.788457	1.066289
500	0.975237	1.071320	0.940467	1.072312	0.800857	1.070549	0.790758	1.071168	0.788341	1.066546
1000	1.070513	1.072439	0.979116	1.063287	0.838710	1.068927	0.801045	1.071827	0.788402	1.071928

the midpoint of the other basis nodes. The choice $\varepsilon = 2/3$ corresponds to Gauss-Radau quadrature, as the collocation nodes are $\hat{t}_1 = 1/3$ and $\hat{t}_2 = 1$. This is the only non-truncating quadrature rule used in this experiment and, therefore, is the only choice for which our theory guarantees second order convergence. When $\varepsilon = 2-\sqrt{2}$, we are using Richardson's TR-BDF, which has been shown to optimize the truncation error of the TR-BDF scheme in other contexts (this choice of collocation node was discussed in more detail in section 6.1) ([23],[72]).

We use a static mesh with a high spatial resolution to keep the error introduced by the spatial discretization below that of the time discretization; furthermore, the mesh is chosen to be static to avoid incurring additional error at mesh discontinuities, as seen in section 7.2. We partition the time domain $(0, 1]$ of each problem with $m = 10, 20, 50, 100$, and 500 time steps to measure how the error of the solution at time $t = 1$ scales with the length of the time step.

For differential equation (7.1), the errors are reported in table 7.3 and plotted on a logarithmic scale in figure 7.5 to show the order of convergence. From table 7.3, it is clear that the convergence is second order with respect to the shrinking the time step length for any of the choices of collocation node. This is a validation that the requirement of a non-truncating quadrature rule is primarily a theoretical issue, or that we simply have not found a proof that can successfully bound the local quadrature of the norm of the characteristic derivative term without this assumption. The best error of our three time-stepping schemes is attained by using the mid-step time basis node at $t_{i,1} = t_{i-1} + (2 - \sqrt{2})\Delta t_i$. Similar findings are found for the problem described in equation (7.2) and these are reported in table 7.4 and figure 7.6.

Overall, this experiment suggests that strict adherence to using non truncating quadrature rules for time stepping is more restrictive than necessary, and small advantages can result by simply adjusting the placement of the time collocation and basis nodes. The benefits of choosing the time basis node to correspond to Richardson's TR-BDF appears to maintain its status of reducing the truncation error better than the other choices considered in this experiment. Perhaps additional gains can be made by testing the Gauss rule for time integration, though

Table 7.3: The error table for problem (7.1). The bottom row is the ratio of the differences of the log errors and the log of the number of time steps; the error displays second order convergence.

	$\varepsilon = 1/2$		$\varepsilon = 2/3$		$\varepsilon = 2 - \sqrt{2}$	
m	$\mathcal{L}_2$-error	$\mathcal{H}^1$-error	$\mathcal{L}_2$-error	$\mathcal{H}^1$-error	$\mathcal{L}_2$-error	$\mathcal{H}^1$-error
10	0.0326283	0.0899276	0.0326081	0.0897437	0.0316864	0.0873559
20	0.0083078	0.0221942	0.0083053	0.0221491	0.0080639	0.0215409
50	0.0013328	0.0034945	0.0013327	0.0034878	0.0012936	0.0033900
100	0.0003333	0.0008698	0.0003332	0.0008682	0.0003234	0.0008436
500	0.0000133	0.0000347	0.0000133	0.0000346	0.0000129	0.0000336
1000	0.0000033	0.0000087	0.0000033	0.0000087	0.0000032	0.0000084
order	1.9954068	2.0076131	1.9952674	2.0075588	1.9955322	2.0079723

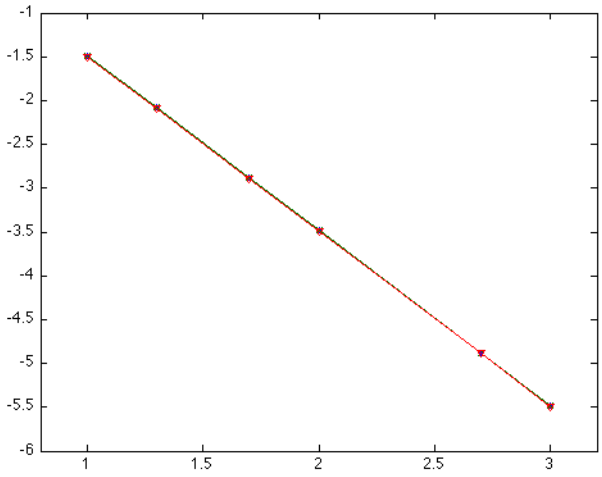

Figure 7.5: The $\mathcal{L}_2$-errors for problem (7.1) plotted on a logarithmic scale. The blue line with filled diamonds corresponds to the basis node $\varepsilon = 1/2$. The red line with hollow triangles corresponds to the basis node $\varepsilon = 2 - \sqrt{2}$. The green line with crosses corresponds to the basis node $\varepsilon = 2/3$. The errors are almost exactly the same in this problem, with a slight gain for $\varepsilon = 2 - \sqrt{2}$. This suggests that non-truncating quadrature rules are not the only quadrature rules that can be used to define the time stepping scheme and still maintain second order convergence with respect to the time discretization.

Table 7.4: The error table for problem (7.2). The bottom row is the ratio of the differences of the log errors and the log of the number of time steps; the error displays second order convergence.

	$\varepsilon = 1/2$		$\varepsilon = 2/3$		$\varepsilon = 2 - \sqrt{2}$	
m	$\mathcal{L}_2$-error	$\mathcal{H}^1$-error	$\mathcal{L}_2$-error	$\mathcal{H}^1$-error	$\mathcal{L}_2$-error	$\mathcal{H}^1$-error
10	0.003439	0.002184	0.002920	0.001851	0.002703	0.001797
20	0.000832	0.000537	0.000714	0.000465	0.000648	0.000444
50	0.000130	8.51E-05	0.000113	7.51E-05	0.000101	7.08E-05
100	3.24E-05	2.12E-05	2.81E-05	1.90E-05	2.508E-05	1.77E-05
500	1.29E-06	8.50E-07	1.12E-06	7.65E-07	9.969E-07	7.13E-07
1000	3.22E-07	2.15E-07	2.81E-07	1.95E-07	2.491E-07	1.82E-07
order	2.0142919	2.002926	2.008410	1.988639	2.017711	1.99714210

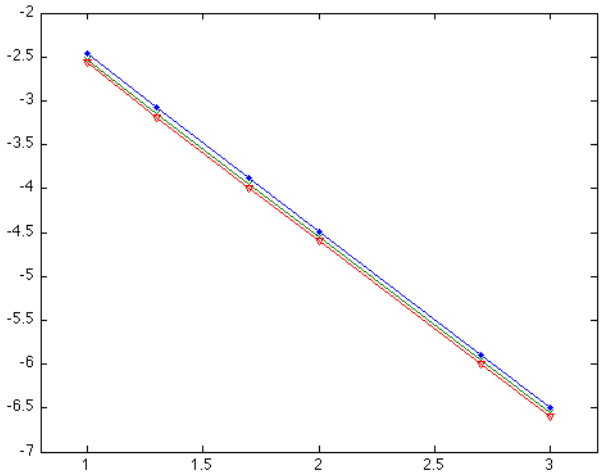

Figure 7.6: The $\mathcal{L}_2$-errors for problem (7.2) plotted on a logarithmic scale. The blue line with filled diamonds corresponds to the basis node $\varepsilon = 1/2$. The red line with hollow triangles corresponds to the basis node $\varepsilon = 2 - \sqrt{2}$. The green line with crosses corresponds to the basis node $\varepsilon = 2/3$. This problem shows a clearer advantage in selecting $\varepsilon = 2 - \sqrt{2}$ compared to the results shown in figure 7.5.

the resulting linear system of this time-stepping scheme would not decouple as TR-BDF does, and would significantly increase the time required to compute the finite element solution.

7.4 Mesh Discontinuities: $\mathcal{L}_2$-projection and interpolation

The experiments in this section address the transfer of the finite element solution across the mesh discontinuities at the end of each time step. Formally, the finite element solution should satisfy

$$(u_h(t_{i-1^+}), \chi) = (u_h(t_{i^-}), \chi),$$

for all χ in $\mathcal{V}_h^p(t_{i-1^+})$, to live in the finite element space. Unfortunately, implementing this condition requires solving a linear system of N_i equations, where N_i is the number of nodes in the time partition. This condition becomes costly to implement when N_i is large and, accordingly, we seek an alternative means to transfer the computed solution across the mesh discontinuities.

In practice, computing a finite element solution by means of interpolation at the discontinuities is much cheaper than performing $\mathcal{L}_2$-projection, especially when Δt or Δx are small. This is due to the fact that interpolation is an element-wise operation, whereas $\mathcal{L}_2$-projection requires solving a linear system that couples neighboring elements. We conduct an experiment to verify that interpolation is a valid technique to more quickly transfer a computed solution over the discrete mesh discontinuities. We compute numerical solutions on identical meshes and compare the results of using interpolation versus $\mathcal{L}_2$-projection at the mesh discontinuities. For this experiment, the final $\mathcal{L}_2$-error of the computed solution at the end of the simulation and the computation time are measured for solving problems (7.1) and (7.2).

For both experiments, we employ uniform time steps and the spatial mesh is initialized to be uniform on each time partition, as in section 7.2. The mesh motion within the partition is again determined by two steps of forward Euler. The mid-

step is chosen to give the Richardson collocation node $t_{i,1} = t_{i-1} + (2 - \sqrt{2})\Delta t_i/2$. Note that the operation to compute the solution at the mesh discontinuities has no effect on the mesh selection; the meshes used for interpolation and $\mathcal{L}_2$-projection are identical, and examples are depicted in figures 7.3 and 7.4.

The result comparisons for equation (7.1) are reported in table 7.5 and the result comparisons for equation (7.2) are given in table 7.6. These tables are of the same form as tables 7.1 and 7.2 and report the ratio of the $\mathcal{L}_2$-errors and computation times of the solutions computed using $\mathcal{L}_2$-projection and interpolation. For entries with $\mathcal{L}_2$-error ratios greater than 1, the corresponding discretization computes a more accurate solution using interpolation. The CPU columns report the relative speedup achieved by implementing interpolation instead of $\mathcal{L}_2$-projection.

In both experiments, we observe that the $\mathcal{L}_2$-error of the computed solutions are very similar when m is not significantly larger than n. When m is large compared to n, we notice that the solution computed using interpolation has lower accuracy, especially for problem (7.1). This behavior suggests that adaptive meshing techniques can be used to effectively safeguard from such cases when the number of time steps greatly exceeds the number of spatial nodes. The most significant discrepancies correspond to performing many interpolations on coarse meshes, which lead to deterioration in the quality of the computed solution. This effect is more pronounced when solving equation (7.1), where the third derivative attains a larger magnitude than the solution of equation (7.2). This sort of behavior has a greater impact on the interpolation error than the error of the projection, which may explain why coarser meshes suffer more from interpolation at the mesh discontinuities. Notice, however, that there is virtually no loss of accuracy when the spatial discretization is comparable to the time discretization. Typically, the use of interpolation realizes a relative speedup of 20% to 30% versus $\mathcal{L}_2$-projection. The results of this experiment suggest that interpolation at the end of the mesh discontinuities more quickly provides a satisfying solution across mesh discontinuities compared to $\mathcal{L}_2$-projection, when the spatial mesh is sufficiently refined.

Considering the results of this experiment in conjunction to those of section 7.2, $\mathcal{L}_2$-projection shows improved stability in cases where the solutions computed

Table 7.5: A comparison table for problem (7.1) of using interpolation instead of $\mathcal{L}_2$-projection at the mesh discontinuities. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative speedup. It is clear that $\mathcal{L}_2$-projection stabilizes the error when the time-discretization is highly refined compared to the spatial discretization, though there are significant saving in CPU time when the spatial mesh is sufficiently refined.

	$n = 51$		$n = 101$		$n = 501$		$n = 1001$		$n = 3001$	
m	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU
10	1.013675	1.269337	1.074038	1.203423	1.116945	1.205104	1.163790	1.211096	1.143329	1.199366
20	0.799725	1.321854	1.018111	1.321415	1.062456	1.324921	1.092720	1.208971	1.078861	1.188541
50	1.216248	1.317664	1.020982	1.192735	1.068564	1.185323	1.087942	1.191887	1.025615	1.193506
75	0.149524	1.323340	0.141877	1.334094	0.802703	1.336129	1.001145	1.334009	1.004031	1.190121
100	0.101092	1.307476	1.135373	1.329163	1.047601	1.343684	1.024721	1.179533	1.001336	1.189121
200	0.087734	1.338993	0.033193	1.335889	0.096340	1.334904	1.000953	1.337970	0.999573	1.344363
500	0.154542	1.335014	0.022868	1.335159	0.953255	1.339678	1.021493	1.188049	0.999287	1.193233
1000	0.260828	1.333412	0.023213	1.337691	0.003171	1.331389	0.944734	1.324594	1.002914	1.339926

Table 7.6: A comparison table for problem (7.2) of using interpolation instead of $\mathcal{L}_2$-projection at the mesh discontinuities. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative speedup. The effects of interpolation are not nearly as present for this problem.

	$n = 51$		$n = 101$		$n = 501$		$n = 1001$		$n = 3001$	
m	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU
10	1.000059	1.234955	1.000004	1.233083	1.000000	1.239047	1.000000	1.220121	1.000000	1.239665
20	0.999235	1.243979	1.000068	1.246374	1.000000	1.245967	1.000000	1.248949	1.000000	1.247735
50	0.988619	1.259846	0.997570	1.249274	1.000005	1.250706	1.000000	1.251428	1.000000	1.255418
75	0.977007	1.258335	0.992764	1.254055	1.000010	1.252234	0.999999	1.252459	1.000000	1.240115
100	0.966102	1.253231	0.986446	1.252675	1.000069	1.252282	1.000005	1.252187	1.000000	1.255037
200	0.959739	1.222131	0.951979	1.256058	0.998731	1.257640	1.000069	1.251025	1.000000	1.254438
500	1.005170	1.251002	0.865667	1.257647	0.984550	1.255728	0.997101	1.253644	1.000027	1.256245
1000	0.933884	1.254346	0.948486	1.254684	0.940213	1.258346	0.984305	1.262315	0.999554	1.253275

on moving meshes with interpolation suffered the greatest losses in accuracy, compared to solutions computed on static meshes. Consequently, when CPU time is not an issue, one can stabilize the moving mesh methods by maintaining the $\mathcal{L}_2$-projection at the mesh discontinuities. Tables reporting the comparative performances of static mesh methods and moving mesh methods using $\mathcal{L}_2$-projection are given by 7.7 and 7.8. Reported are the relative decrease in the $\mathcal{L}_2$-error and the increase in CPU times.

Plots of the final $\mathcal{L}_2$-errors are plotted in figures 7.7–7.10 as a function of the number of time steps. For these plots, the spatial resolution is fixed, we take $n = 101$ for example, and plot the $\mathcal{L}_2$-error of the computed solution at the end of the simulation. Note that the time domain is not changing by taking more time steps, rather we are refining the time discretization. These plots are simply designed to help visualize the behavior of the error of the computed solution with respect to the various discretization schemes. The stabilization property of $\mathcal{L}_2$-projection for coarse meshes is striking in figure 7.7 — note that the interpolation error is much greater when the third derivative attains larger magnitudes, as in problem (7.1). These figures indicate that properly implemented adaptive time-stepping techniques for the moving mesh methods can yield great savings and avoid time steps that are too small, which can stabilize the moving meshes in the case of using interpolation at the mesh discontinuities.

7.5 Moving finite elements with spatial adaptivity

We describe a basic procedure for adaptively defining a moving mesh based on derivative recovery. The mesh is initialized on each time partition by an h-refinement of a given mesh, coming from the previous time partition (or an initial configuration when $t = 0$). In earlier experiments, r-adaptivity (or mesh smoothing) was implemented [20] though later disabled due to its high computational costs. Note that p-methods are not straightforward to implement for moving finite elements as the number of collocation nodes is determined by the order of the

Table 7.7: A comparison table for problem (7.1) between static mesh methods and moving mesh methods using $\mathcal{L}_2$-projection. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative increase in CPU time of the moving mesh solution to the static mesh solution. It is clear that $\mathcal{L}_2$-projection stabilizes the error for coarse spatial meshes.

	$n = 51$		$n = 101$		$n = 501$		$n = 1001$		$n = 3001$	
m	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU
10	0.023469	1.340248	0.010311	1.225176	0.005486	1.215725	0.004201	1.219736	0.003504	1.216247
20	0.040873	1.382782	0.011851	1.383168	0.003372	1.383152	0.002940	1.258424	0.001975	1.245459
50	0.073579	1.427572	0.024006	1.271880	0.003404	1.258282	0.002067	1.263306	0.001687	1.279486
75	0.074806	1.422589	0.036318	1.423840	0.002505	1.415333	0.001621	0.767666	0.003149	1.264457
100	0.077242	1.442543	0.053515	1.425227	0.001951	1.429824	0.002947	0.614530	0.004230	1.271793
200	0.103795	1.436850	0.159726	1.426901	0.002934	1.428083	0.003651	1.420963	0.006526	1.430918
500	0.225058	1.436938	0.535570	1.432909	0.003565	1.430435	0.004853	1.262727	0.008271	1.276127
1000	0.404916	1.455557	0.699234	1.443170	0.012095	1.436421	0.004418	1.428695	0.008719	1.439483

Table 7.8: A comparison table for problem (7.2) between static mesh methods and moving mesh methods using $\mathcal{L}_2$-projection. Reported for each mesh discretization is the ratio of final $\mathcal{L}_2$-errors and the relative increase in CPU time of the moving mesh solution to the static mesh solution.

	$n = 51$		$n = 101$		$n = 501$		$n = 1001$		$n = 3001$	
m	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU	$\mathcal{L}_2$-error	CPU
10	0.800924	1.304705	0.801251	1.322061	0.801366	1.329919	0.801370	1.320417	0.801371	1.326150
20	0.792576	1.329198	0.792738	1.331434	0.792828	1.339651	0.792831	1.314780	0.792832	1.336290
50	0.790022	1.342566	0.789264	1.336132	0.789289	1.340845	0.789292	1.336667	0.789292	1.345942
75	0.791299	1.338722	0.788867	1.340597	0.788806	1.342078	0.788808	1.340167	0.788806	1.344643
100	0.794276	1.341391	0.788821	1.341324	0.788629	1.342307	0.788631	1.327645	0.788628	1.337806
200	0.831651	1.338985	0.789918	1.344520	0.788475	1.361113	0.788475	1.339546	0.788457	1.337594
500	0.980278	1.340222	0.814131	1.348590	0.788483	1.344318	0.788466	1.342864	0.788362	1.339843
1000	0.999735	1.345209	0.928678	1.334090	0.788566	1.345081	0.788472	1.352983	0.788050	1.343420

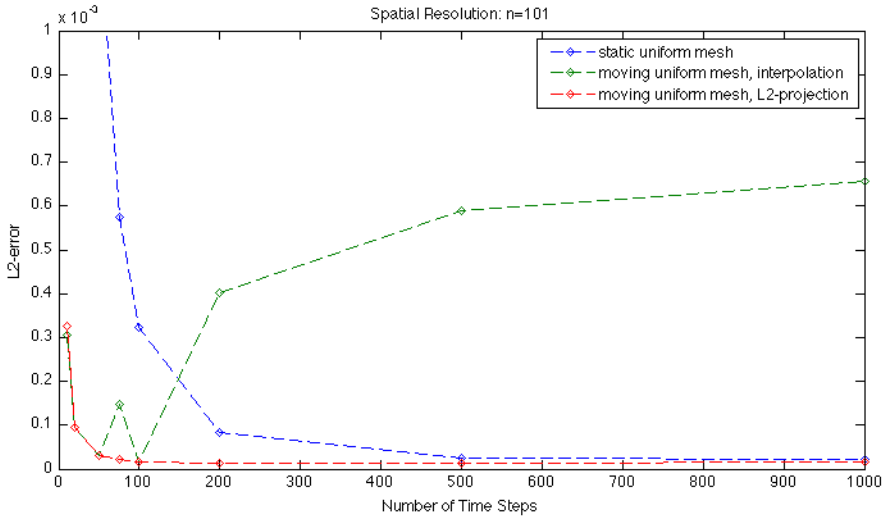

Figure 7.7: The $\mathcal{L}_2$-errors for problem (7.1) plotted with respect to the number of time steps. For this plot, each discretization employs 101 spatial nodes. As the number of time steps increases, the time discretization is refined, rather than extending the time domain. The most striking feature here is that the $\mathcal{L}_2$-error does not decrease monotonically when using a moving mesh and interpolation at the mesh discontinuities. The plot suggests that using $\mathcal{L}_2$-projection maintains the benefit of smaller error of fewer time steps, as compared to using a static mesh, though it avoids the build-up of error when more time steps are used.

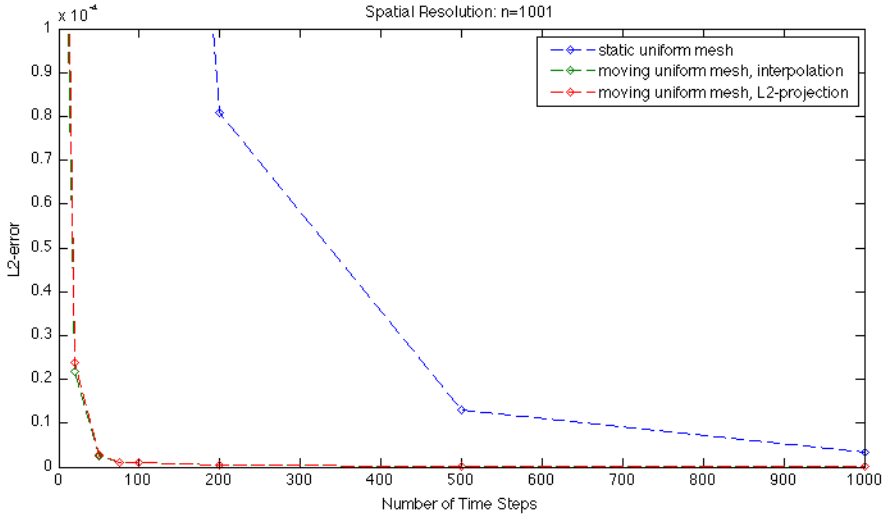

Figure 7.8: The $\mathcal{L}_2$-errors for problem (7.1) plotted with respect to the number of time steps. For this plot, each discretization uses 1001 spatial nodes. Unlike the case where only 101 spatial nodes are used, the interpolation error does not buildup as the number of time steps increases when using a moving mesh; the benefits of using $\mathcal{L}_2$-projection are not significant compared to using interpolation, and the increase in CPU time is more dramatic as the mesh is more refined than the case when 101 spatial nodes are used.

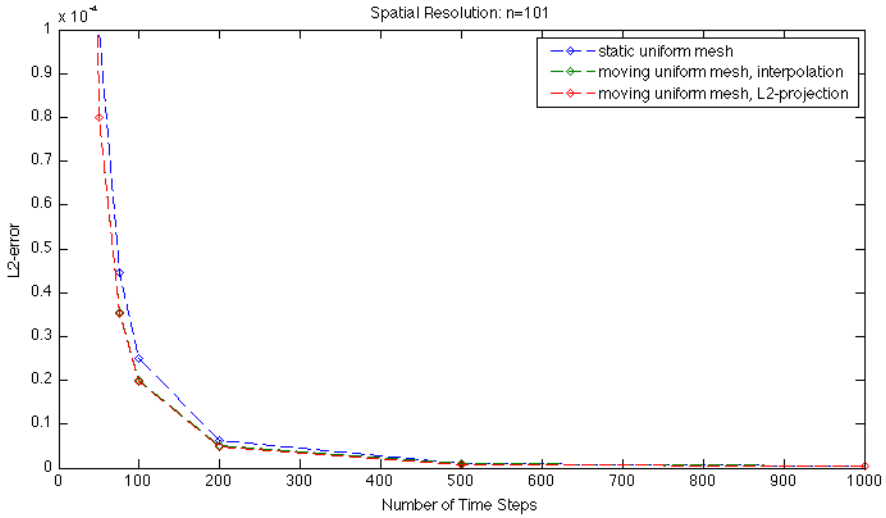

Figure 7.9: The $\mathcal{L}_2$-errors for problem (7.2) plotted with respect to the number of time steps. For this plot, each discretization employs 101 spatial nodes. As the number of time steps increases, the time discretization is refined, rather than extending the time domain. The smallest $\mathcal{L}_2$- error of the computed is attained by using a moving mesh with $\mathcal{L}_2$-projection at the mesh discontinuities, though the slight reduction in error over using interpolation at the mesh discontinuities might not justify the more prominent increase in CPU time.

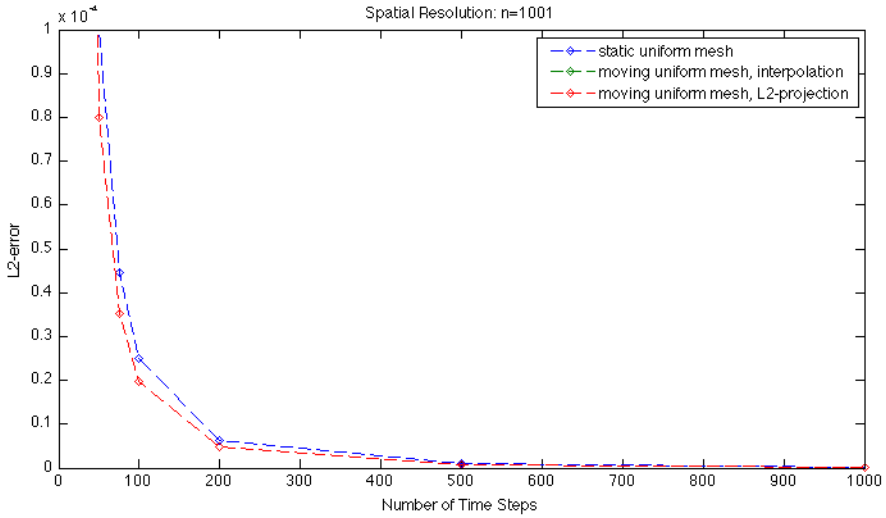

Figure 7.10: The $\mathcal{L}_2$-errors for problem (7.2) plotted with respect to the number of time steps. For this plot, each discretization uses 1001 spatial nodes. In comparison to figure 7.9, there are virtually no gains in using 1001 spatial nodes versus 101 nodes for this problem.

finite element space. The motion of the mesh is again chosen to approximate the method of characteristics.

7.5.1 An h-adaptive method

For each time partition, we assume that we are given an initial mesh and an initial approximation of the solution, coming from either a previous time partition or the initial condition. To find the mesh on the new time partition, we begin by computing the cubic spline of the approximate solution, $\mathcal{S}u_h$, which provides a piecewise constant approximation of the third derivative of the solution. Since we are using piecewise quadratic polynomials to approximate the solution, it is assumed for each element, $e_k(t_{i,j}) = [x_{k-1}(t_{i,j}), x_k(t_{i,j})] \subset \Omega$, that

$$|u(t) - u_h(t)|_{(1,e_k(t))} = C\Delta x_k^2(t)|u(t)|_{(3,e_k(t))} \approx C\Delta x_k^2|\mathcal{S}u_h(t)|_{(3,e_k(t))},$$

for some positive constant C. The third derivative recovered from the cubic spline is then used to indicate the error of our computed solution on each element. The aim of the adaptive procedure is to find a mesh such that the error indicated by the recovered derivative is equally distributed on the new mesh.

For h-adaptivity, we use the local error indicator derived from the cubic spline to determine which elements to bisect or join with neighboring elements. We approximate the number of nodes in the new mesh by

$$n_{i+1} \approx 1 + (n_i - 1)\sqrt{\frac{|\mathcal{S}u_h(t_{i^-})|_3}{|\mathcal{S}u_h(t_{i-1^-})|_3}},$$

which comes from setting

$$(n_{i+1} - 1)^{-2}|\mathcal{S}u_h(t_{i^-})|_3 \approx (n_i - 1)^{-2}|\mathcal{S}u_h(t_{i-1^-})|_3.$$

This statement suggests that the errors at the end of two consecutive time steps should be similar. To determine whether an element should be refined, we check whether the local error indicator is twice as big as the average predicted error:

$$\Delta x_k^4|\mathcal{S}u_h(t_{i^-})|^2_{(3,e_k(t_{i^-}))} \leq 2(n_{i+1} - 1)^{-4}|\mathcal{S}u_h(t_{i^-})|_3^2.$$

If the error indicator is more than twice as large as the predicted average, the element is refined until the local error indicator is below average on each refined element. To coarsen the mesh, we sweep through the mesh evaluating the local error indicator on the element that results from deleting a spatial vertex. If the local error indicator on the coarsened element is below the refinement threshold, $2(n_{i+1}-1)^{-4}|\mathcal{S}u_h(t_{i^-})|_3^2$, then the element is coarsened; otherwise, the vertex is replaced and the next vertex is tested for deletion.

Two example meshes are displayed in figures 7.11 and 7.12. Comparing the density of the spatial nodes to their respective solutions, shown in figures 7.1 and 7.2, it is clear the the heaviest concentration of spatial nodes occur where the solution is changing rapidly in spatial directions, as desired.

7.5.2 Time discretization

For moving meshes, two steps of forward Euler are applied to advance the mesh to the time basis nodes according to the evolution equation $x_t(t) = b(x,t)$. This choice of mesh motion helps improve the condition of the linear system to solve for the finite element solution, though other choices of mesh motion are viable.

Another viable candidate would be to attempt to distribute the $\mathcal{L}_2$-error or $\mathcal{H}^1$-error of the solution between the elements and move the mesh so that the proportion of the error on each element is preserved ([1],[3]) — essentially, the error would serve as a continuous-in-time monitor function that describes the mesh motion. A predictor-corrector approach is one way to implement such a mesh motion scheme, where the 'predictor' step computes a coarse approximation of the solution on a time-step and its error indicator can be used as the monitor function to determine the mesh motion for a 'correction' step. A version of this scheme has been explored by [3], where numerical experiments indicate that this mesh motion improves the accuracy of the computed solution. Another approach to determining the motion of the mesh would be to employ mesh smoothing techniques [20], where an error indicator at each collocation node can be computed to determine how to perturb the placement of the spatial nodes, while preserving the mesh topology. Smoothing the mesh at the time collocation nodes would, therefore, determine the

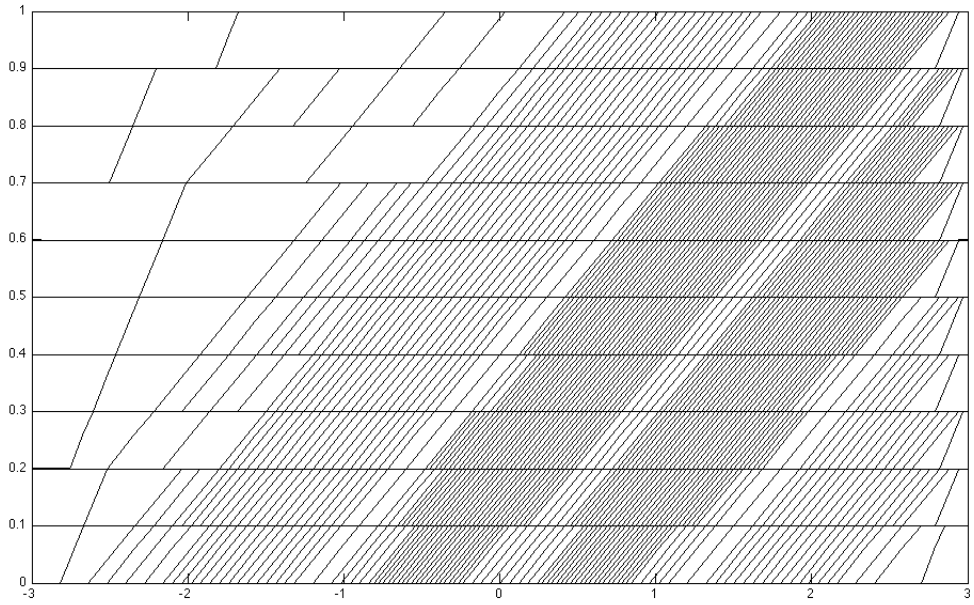

Figure 7.11: An example of a spatially adaptive moving mesh, given by the method of characteristics for equation (7.1). In this example, we have $m = 10$ time steps and initialize each time partition with $n = 51$ spatial nodes. As a result of using the method of characteristics, the spatial nodes on the interior of the domain satisfy $x_t = 3$; some spatial nodes have been deleted near the outflow boundary and inflow boundary. Comparing the density of the spatial nodes to the figure of the solution in figure 7.1, the spatial resolution is higher where the solution is changing rapidly. Since the convection term aligns the mesh with the solution of the differential equation, the nodes of the mesh trace out level sets of the solution in time (except near the boundaries), which reduces the impact of the discontinuities between time partitions.

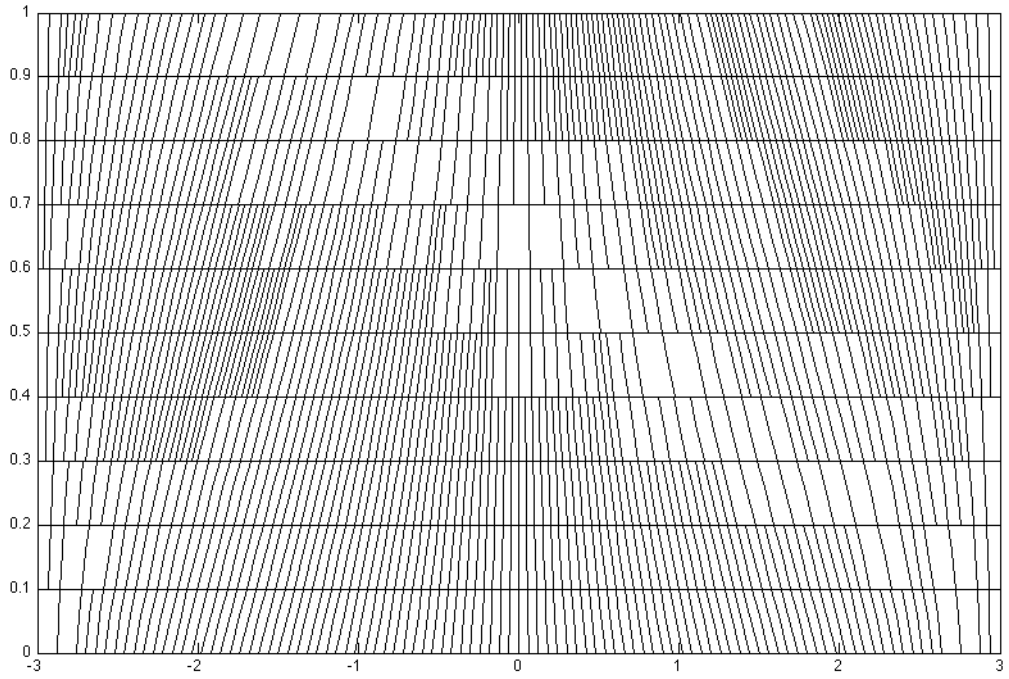

Figure 7.12: An adaptive mesh generated by the method of characteristics applied to (7.2). Though the curvature is subtle, these mesh trajectories are quadratic and approximately satisfy the evolution equation $x_t = 0.1(x^3 - 9x)$. In this example, we take $m = 10$ time steps and initialize each partition with $n = 51$ spatial nodes. In this problem, the convection term does not align with the motion of the solution in time; mesh discontinuities are more prominent than in figure 7.11, though they are still an improvement over the mesh depicted in figure 7.4, where the mesh configuration is reconfigured to be uniform on each time partition.

motion of the nodes throughout a time partition.

We also attempted to employ variable sized time steps that took advantage of the error indicators computed for the spatial adaptivity, though the discontinuity between the time partitions complicated the use of our derivative recovery schemes and we were unable to achieve satisfactory results. As a result, uniform time steps were used in the following experiments. The use of predictor-corrector methods provides a possible solution for effectively predicting the length of the time steps. By predictor-corrector methods, the time-step can be chosen so that the solution on the correction mesh is equal to a prescribed error tolerance. Another alternative would be to solve for the finite element solution on a time partition twice, where one solution is solved on a refined time partition. Then, the difference between the solution computed on the coarser mesh and the solution computed on the refined mesh can serve as an error indicator for the coarse solution and, thusly, can be used to determine the length of the time steps.

7.5.3 Experiments with h-adaptivity in space

For the experiments with h-adaptive meshing, the initial condition is projected onto a uniform mesh with a prescribed number of spatial nodes. Based on this initialization of the numerical solution, we use the derivative recovery scheme described in section 7.5.1 to compute an indication of the initial error. Then, this indicator is used to predict the number of spatial nodes in subsequent time partitions so that the error indicators are relatively constant throughout the simulation.

Tables 7.9 and 7.10 provide the number of spatial nodes left at the end of each simulation using adaptive meshing. The final $\mathcal{L}_2$-error of the computed solution is plotted against the number of time steps used to solve the differential equation in figures 7.13–7.15 to show how h-adaptivity affects the final $\mathcal{L}_2$-error of the computed solution. As in figures 7.7–7.10, more time steps lead to a more refined the time discretization rather than extending the time domain. Reviewing these tables and figures, it becomes clear that the approach for predicting the number of spatial nodes required in a time partition needs improvement. There is clear evidence that the mesh tends both to over-refine and over-coarsen the mesh,

Table 7.9: This table reports the count of spatial nodes in the mesh at the end of the simulation using adaptive meshing. The problem solved is (7.1) and these simulations were initialized using 101 (top) and 1001 (bottom) spatial nodes. The final count of nodes is expected to be low as the bump structure in the solution is exiting through the right boundary, leaving a very flat solution on the majority of the spatial domain.

initial $n = 101$

	Static Mesh		Moving Mesh	
m	Interpolation	$\mathcal{L}_2$-projection	Interpolation	$\mathcal{L}_2$-projection
10	47	45	61	61
20	59	56	62	63
50	72	75	67	67
75	79	79	67	67
100	82	84	65	68
200	63	71	65	68
500	65	68	82	67
1000	89	88	78	67

initial $n = 1001$

	Static Mesh		Moving Mesh	
m	Interpolation	$\mathcal{L}_2$-projection	Interpolation	$\mathcal{L}_2$-projection
10	237	237	223	124
20	349	348	158	158
50	528	516	256	251
75	631	621	281	279
100	701	673	318	296
200	711	767	321	387
500	645	800	616	594
1000	646	649	629	626

depending on the problem at hand and the length of the time steps.

For problem (7.1), the $\mathcal{L}_2$-error is significantly larger for solutions found on adaptive meshes. This is due, in part, to the fact the the "bump" in the solution, depicted in figure 7.1, is exiting the domain by the end of the simulation. Consequently, the error in the previous experiments decreased as the bump flowed out of the domain. Since the adaptive method attempts to keep the error constant, this decrease in error is not realized when using h-adaptivity; instead, the mesh is coarsened and the error is approximately maintained. Note that the greatest

Table 7.10: This table reports the count of spatial nodes in the mesh at the end of the simulation using adaptive meshing. The problem solved is (7.2) and simulations were initialized using 101 (top) and 1001 (bottom) spatial nodes. The final count of nodes is rather high since the errors displayed in figure 7.15 are not significantly reduced; this indicates that our mesh selection scheme needs improvement.

initial $n = 101$

	Static Mesh		Moving Mesh	
m	Interpolation	$\mathcal{L}_2$-projection	Interpolation	$\mathcal{L}_2$-projection
10	104	104	119	119
20	154	149	167	167
50	191	195	219	219
75	192	192	210	210
100	173	184	202	213
200	177	177	203	213
500	180	180	204	206
1000	180	181	203	203

initial $n = 1001$

	Static Mesh		Moving Mesh	
m	Interpolation	$\mathcal{L}_2$-projection	Interpolation	$\mathcal{L}_2$-projection
10	829	829	716	716
20	994	994	1127	1127
50	1943	1943	1762	1762
75	2065	2065	2002	2002
100	1932	1932	1884	1884
200	1965	1965	2201	2121
500	2462	2340	2245	2307
1000	945	945	2244	2265

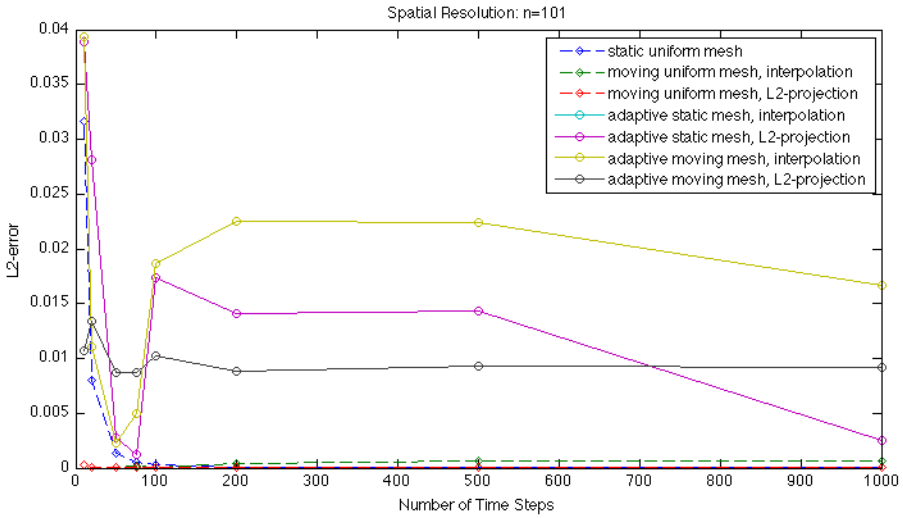

Figure 7.13: The $\mathcal{L}_2$-errors for problem (7.1) plotted with respect to the number of time steps. For this plot, each discretization employs 101 spatial nodes. The solutions computed on adaptively generated meshes display significantly higher errors. In comparing the error to the number of nodes at the end of the simulation, it is clear that predictor for the number of spatial nodes requires adjustment.

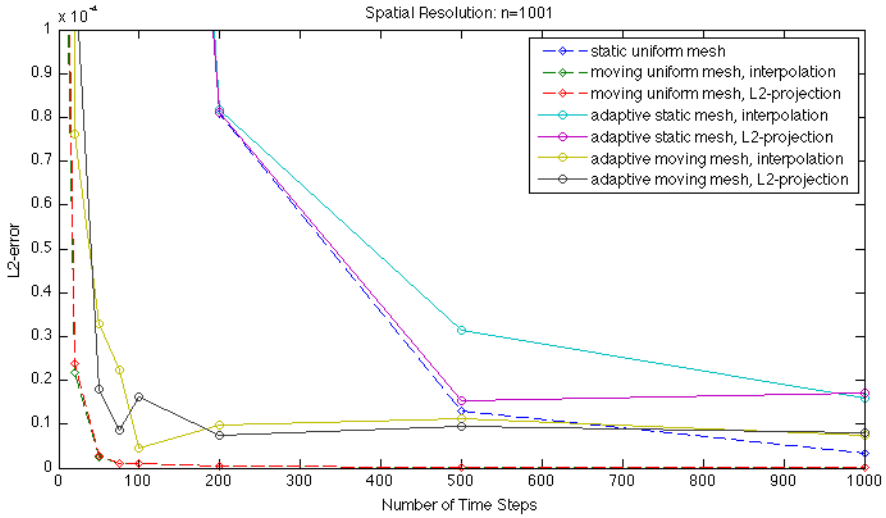

Figure 7.14: The $\mathcal{L}_2$-errors for problem (7.1) plotted with respect to the number of time steps. For this plot, each discretization uses 1001 spatial nodes. The solutions found by adaptive meshing show a great improvement in terms of their overall performance. The fact that the final errors do not shrink as much as in the non-adaptive cases may be a result of the error indicator staying relatively constant throughout the simulation, even as the bump structure is exiting the right boundary (see figure 7.1).

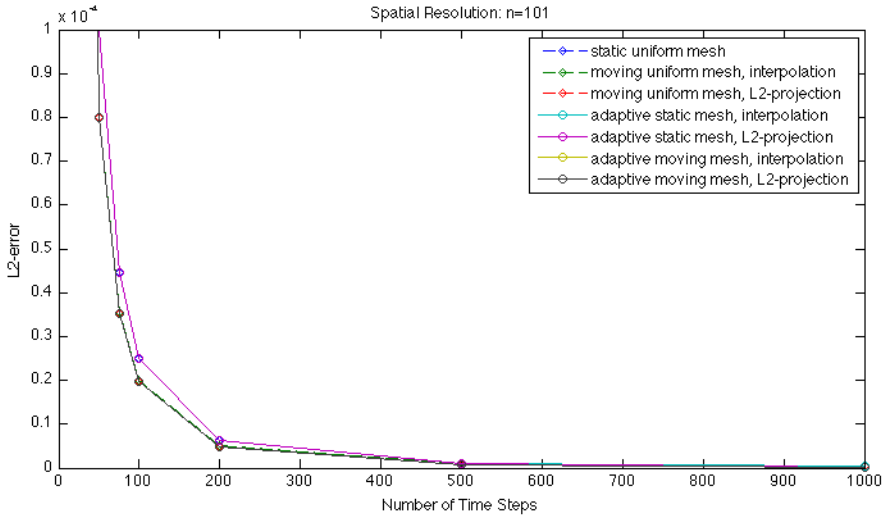

Figure 7.15: The $\mathcal{L}_2$-errors for problem (7.2) plotted with respect to the number of time steps. For this plot, each discretization employs 101 spatial nodes. Table 7.10 reports that the adaptive meshing scheme adds many spatial nodes to the mesh, though we see that the error is not significantly reduced. The adaptive method is over-solving the problem.

errors for the adaptive schemes correspond to the coarsest meshes. Perhaps a mesh grading scheme that requires neighboring elements to be comparable in size improve the overall performance of the adaptive scheme.

For the test problem given in (7.2), table 7.10 reports that the mesh tends to refine in our simulations, but figure 7.15 demonstrates that the error does not decrease significantly as the mesh refines. Thus, our adaptive method is prescribing too many spatial nodes, and we are consequently "over-solving" problem (7.2). Upon closer inspection, slight gains in accuracy are realized in using the adaptive method, though such small gains do not justify the exhaustive use of so many spatial nodes, as shown in table 7.10. As before, the solution computed using 1001 and 101 spatial nodes are nearly identical and their errors display nearly identical behavior with respect to the number of time steps.

Overall, the mechanics of the h-adaptive method seem satisfactory in the sense that the derivative recovery scheme using cubic spline interpolation seems to properly identify regions where the solution is changing rapidly; this is indicated in figures 7.11 and 7.12, which display two examples of the meshes generated by our

derivative recovery based scheme. Unfortunately, it seems that our approach for predicting the number of spatial nodes for the time partitions needs improvement, and perhaps a mesh grading constraint could yield improved shape regularity of the meshes given by the adaptive method. Another possibility is that an effective method for predicting the length of the time step could also help stabilize the error.

7.6 Moving finite elements and Burger's equation

In this section, we test our moving finite element method on Burger's equation. This is a nonlinear equation given by

$$u_t - \frac{1}{R}u_{xx} + uu_x = 0 \quad \text{on } \mathcal{F} = [-3, 3] \times (0, 2], \tag{7.3}$$

where $R > 0$, and we assume a Neumann boundary condition, $u_x(\pm 3) = 0$. For large values of R, the solution to Burger's equation can develop steep shock layers that propagate through the domain; accordingly, we anticipate that moving meshes will improve the accuracy of computed solution in these cases. The number R is the Reynolds number and it controls the amount of diffusion in equation (7.3). In the limit as R tends to infinity, the equation becomes ill-conditioned and the solution forms a discontinuity. Thus, we take R to be a large and finite number, since smaller values of R increase the amount of diffusion in the equation and, as a result, the moving front is smeared across a larger portion of the domain. We use this test equation because it is a simple nonlinear equation that leads to sharp moving fronts in the solution for large values of R.

Due to the nonlinearity of Burger's equation, the linear systems recovered in (5.11) and (6.6) become nonlinear and, consequently, we employ Newton's method to solve for the finite element solution at the time collocation nodes. Our approach to this nonlinear solve is basic; we perform a single iteration of Newton's method at each collocation node to solve for the update of the solution from one time collocation node to the next.

In these experiments, we set the initial condition to be the piecewise linear

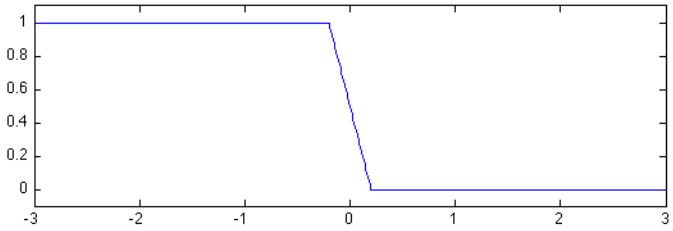

Figure 7.16: An illustration of the initial condition used for Burger's equation. As the PDE propagates the solution in time, the structure on the left side should move toward the right, creating a thin shock layer that propagates from the middle of the domain toward the right boundary.

function

$$u_0(x) = \begin{cases} 1, & \text{for } -3 \leq x \leq -0.2 \\ 0.5 - 2.5x, & \text{for } -0.2 \leq x \leq 0.2 \\ 0, & \text{for } 0.2 \leq x \leq 3 \end{cases},$$

which is depicted in figure 7.16. Since $u_0(x)$ is larger for smaller values of x, the convection velocity (given by u for this equation) moves the structure on the left of the solution towards the right, eventually forming a steep moving front that will sweep out toward the right boundary.

7.6.1 Experiments on Burger's equation

For these experiments, we compute solutions on static and moving meshes, with a finite element space of piecewise tensor product quadratic polynomials ($p = 2$). Since we are solving a nonlinear equation, the mesh velocity depends on the solution to the PDE. Thus, we want

$$x_t = u(x, t).$$

Unlike the previous experiments, we cannot solve this equation using two steps of forward Euler since the solution is unknown at the mid-step basis node. Initially, we attempted to move the mesh partway through a time partition, solve for the mid-step solution, and then continue to move the mesh through the rest of the time partition. Unfortunately, due to the thinness of the shock layer, nodes would often degenerate before solving for the solution at the end of the time step, and

this would lead to hanging nodes in the middle of the time partition. To keep our approach simple, we simply use linear mesh motion, given by a single step of forward Euler, where any node that crashed into a neighboring node (as is commonplace near the shock layer) is deleted. In a sense, deleting these nodes does not lose very much information about the solution, as these deletions only occur where the mesh is extremely dense. For these simulations, we do not allow for mesh discontinuities in order to prevent errors from transferring the computed solution from one time partition to the next. An example of one such mesh is depicted in figure 7.17.

In the first simulations, we assume a moderate amount of diffusion by setting $R = 100$. The solutions are computed on meshes initialized with 61 spatial nodes, and either 10 time steps (shown in figure 7.18) or 25 time steps (shown in figure 7.19). It is known that finite element solutions for Burger's equation typically display artificial oscillations near the shock layers in the solution [57], and this behavior is displayed for both static and moving meshes when we attempt to solve equation (7.3) in only 10 time steps. In figure 7.19, however, we see that taking 25 time steps leads to a sufficiently refined time discretization to dampen these undesirable oscillations near the shock when using moving finite elements, though the non-moving finite element solution still bears these errors. This shows that our moving finite element method shows improved accuracy over the non-moving method, even for this nonlinear equation.

We also ran simulations where the Reynolds number is set to 1000. This reduces the diffusive forces in the equation and leads to a thinner shock. The solutions computed on static and moving meshes with 100 time steps are displayed in figure 7.20. Each mesh is initialized with 301 spatial nodes, though the moving mesh deletes many nodes within the shock layer and has only 112 nodes in the final mesh. From the figure, it is evident that the moving mesh serves its purpose in controlling the artificial oscillations induced by using a relatively coarse time step.

We also ran a simulation to compute the solution using the h-adaptive method devised in section 7.5.1. In this simulation, we set the Reynolds number

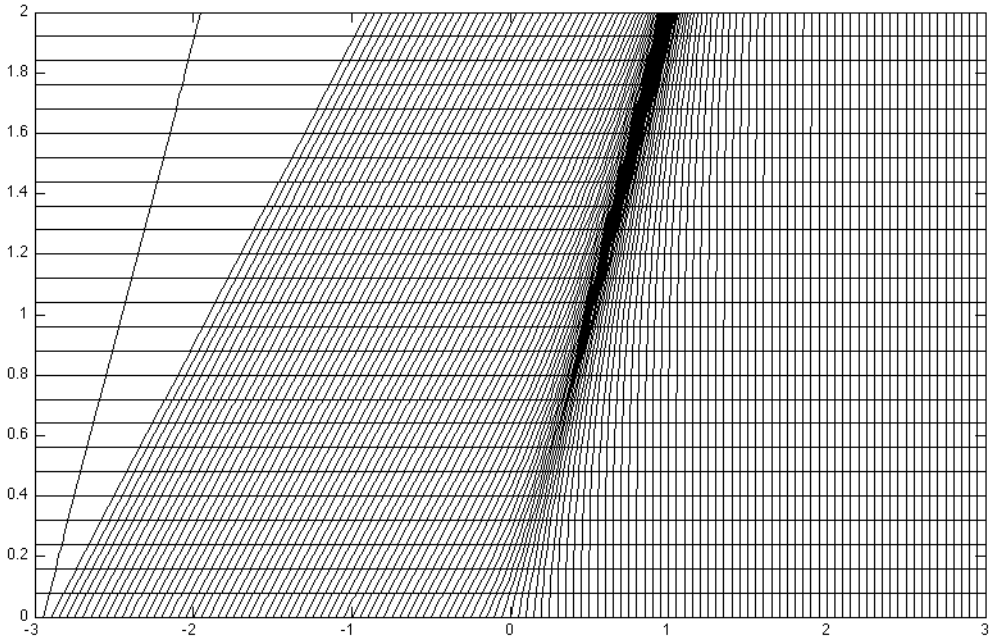

Figure 7.17: An example of a moving mesh, given by the method of characteristics for equation (7.3). In this example, we have $m = 25$ time steps and initialize each time partition with $n = 61$ spatial nodes. As a result of using the method of characteristics, the spatial nodes on the interior of the domain approximately satisfy $x_t = u(x)$. Comparing the density of the spatial nodes to the figures of the computed solutions in figures 7.18 and 7.19, the spatial nodes properly congregate near the shock layer and track it throughout the simulation. The trailing node near the left boundary is the bump node on the leftmost element. Its motion is slower as it must bisect the hat nodes, one of which is fixed on the boundary.

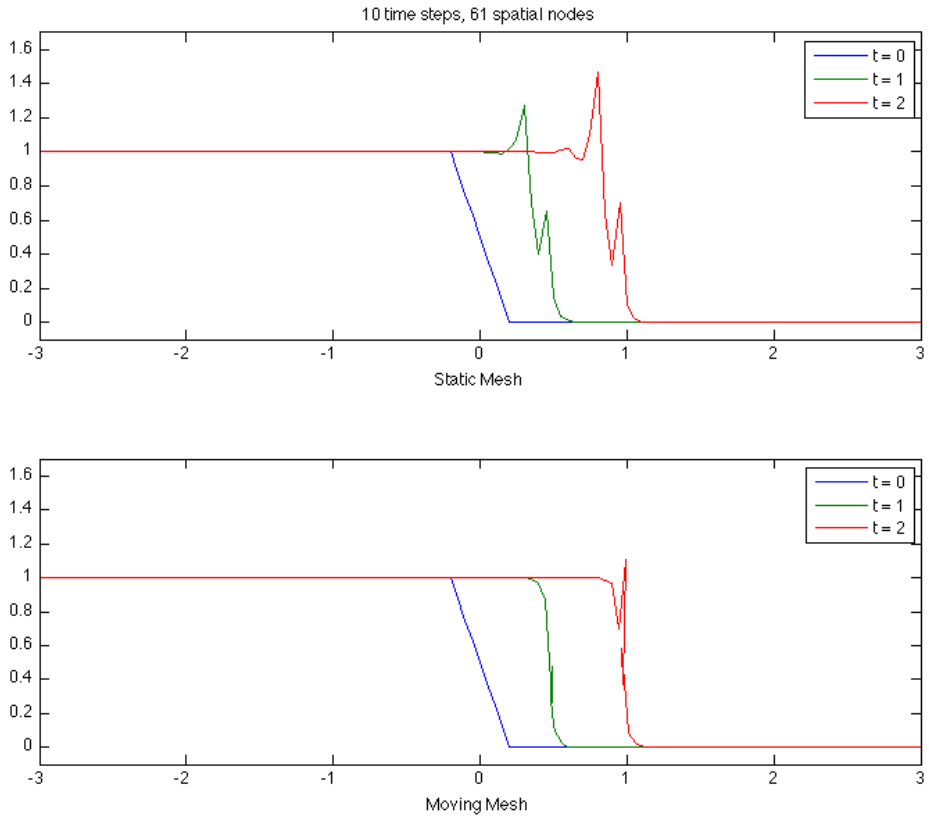

Figure 7.18: Computed solutions for equation (7.3) with $R = 100$ using a static mesh (top) and a moving mesh (bottom). The meshes are initialized to have 61 spatial nodes and 10 time steps are used to advance the solution to time $t = 2$. In both cases, the time step is too large and the computed solutions form artificial structures near and within the shock layer.

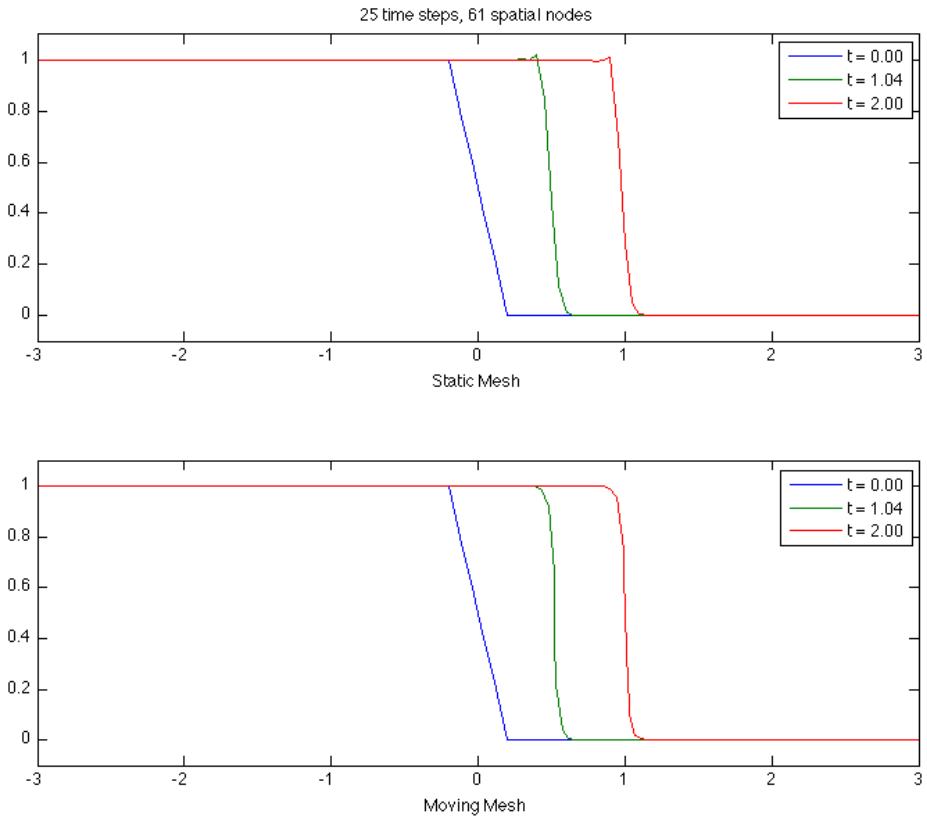

Figure 7.19: Computed solutions for equation (7.3) with $R = 100$ using a static mesh (top) and a moving mesh (bottom). The meshes are initialized to have 61 spatial nodes and 25 time steps are used to advance the solution to time $t = 2$. The solution computed using moving meshes has dampened the oscillatory behavior occurring near the shock layer, though the solution computed using a static mesh still displays this undesirable behavior.

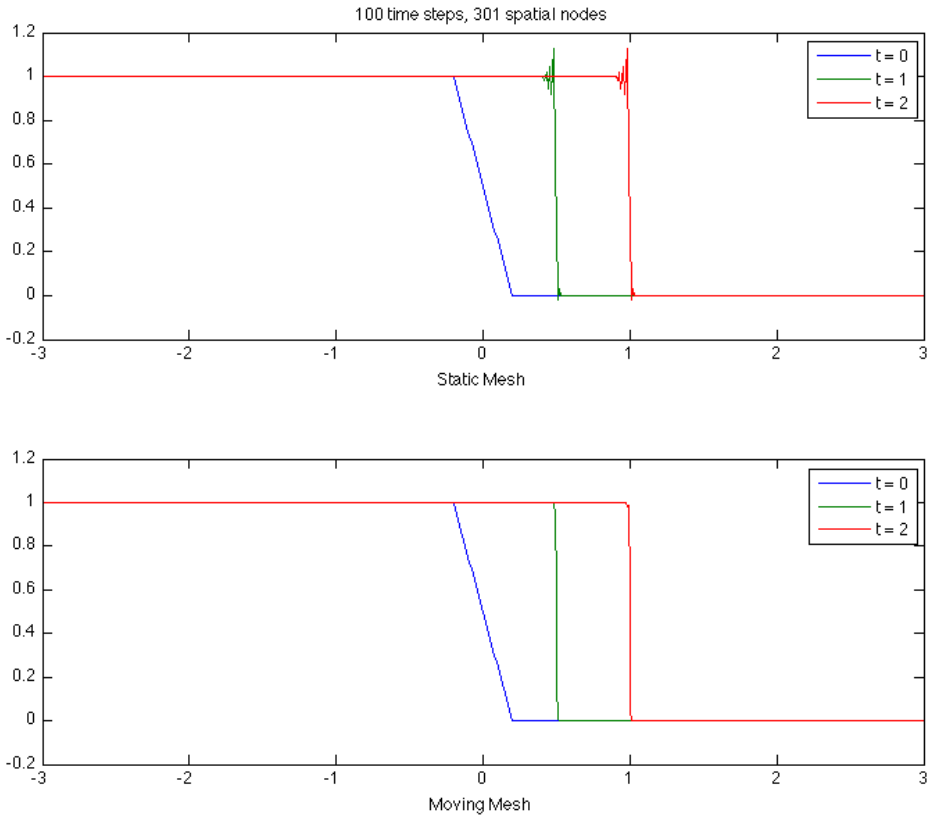

Figure 7.20: In this simulation, the Reynolds number is set to 1000, causing a thinner shock layer. Solutions were computed using $m = 100$ time steps and initialized with $n = 301$ spatial nodes. The static mesh in the top figure displays the usual artificial oscillations that occur with finite element solutions near the shock layer, whereas the moving mesh has suppressed this behavior.

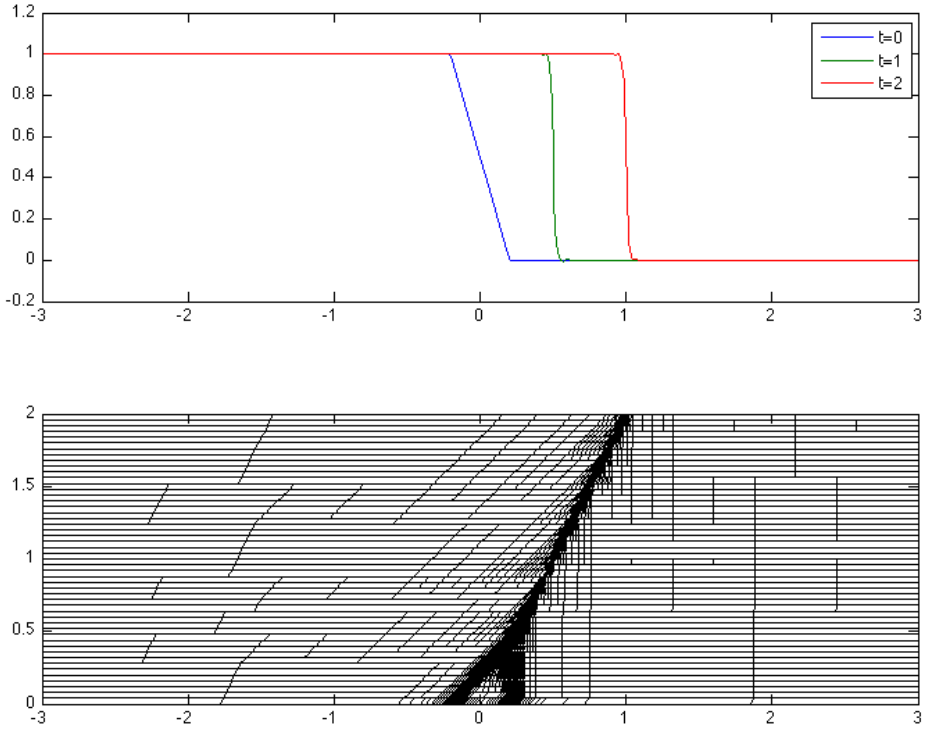

Figure 7.21: The computed solution and its mesh using the adaptive method of section 7.5.1. The adaptive method effectively coarsens the mesh where the solution is unchanging, while maintaining a high resolution near the shock layer. The computed solution still displays small oscillations, though the mesh generated by the adaptive method seems satisfactory. Mesh grading near the shock layer could potentially dampen these oscillations.

to 500 so that only 50 time steps are required for the solution to be approximated reasonable well. The moving mesh is initialized with 161 nodes and at the end of the simulation, only 21 nodes remain. The computed solution and its mesh are illustrated in figure 7.21. Small oscillations still form near the shocks, implying that the adaptive method may over-coarsen the mesh close the steep front. While it is difficult to determine the exact cause of the oscillations, the mesh discontinuities arising from the adaptive approach are also suspect. Furthermore, forcing the mesh to satisfy some grading constraint might also help eliminate this behavior.

From our experiments, it is evident that the presented moving finite element

method has the potential to solve simple nonlinear equations. Aligning the mesh velocity with the convection velocity permitted larger time steps, while avoiding artificial oscillations in the computed solution near the moving front. Further experimentation with multiple iterations of Newton's methods may prove to be useful for improving the accuracy of the computed solution, though a well-designed time step predictor may be a more cost effective approach to dealing with the nonlinearity of the PDE.

Chapter 7, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Chapter 8

Conclusion

In this thesis, we have explored some theoretical and practical aspects of space-time tensor product moving finite elements of arbitrary order. A method was proposed that used these finite element spaces and the method of lines to solve time-dependent linear convection-diffusion-reaction equations and an error analysis was provided to estimate the error of the finite element solution found by this method. A framework was proposed for generalizing the space-time finite element method to employ more flexible time integration schemes, with an error estimate proven for the special case of tensor-product quadratic finite elements and the TR-BDF method used for time integration. Furthermore, we have experimented with a numerical solver for PDEs with a single dimensional spatial domain using these moving finite elements.

The construction of the tensor product finite element space was given in chapter 3, which detailed the general properties of the mesh, the reference elements, the degrees of freedom, the isoparametric maps, and the basis functions. Furthermore, we established relationships between the isoparametric maps with the characteristic trajectories of the mesh as well as the shape regularity constraints of the mesh. These shape regularity constraints correspond to standard assumptions for the space and time discretization independently, plus another constraint that ensures the space-time shape regularity. This new space-time shape regularity assumption constrains the degree to which the eigenvalues of the Jacobian of the spatial component of the isoparametric map can with respect to time, effectively controlling how much the shape and size of the element can be distorted and ruling out the possibility of element degeneration.

In chapter 4, we used the basis functions, isoparametric maps, and shape regularity assumptions of the finite element space to establish some approximation properties of the finite element subspace to the true solution space of the differential equation. In this analysis, we derived some results the tools that allowed us to prove our symmetric error estimate for the finite element solution defined in chapter 5. The main tool was the shift lemma (lemma 2), which established the continuity of the shift operation in the $\mathcal{L}_2$ and $\mathcal{H}^1$-norms. The final result of chapter 4 was the bound on the best approximation error in the finite element space for

functions on the space-time domain. Ultimately, the best approximation error for tensor product degree p piecewise polynomials was shown to be proportional to $\Delta x^p + \Delta t^p$, for $p \geq 1$.

Chapter 5 introduced a space-time moving finite element method that was compatible with the space-time tensor product finite element spaces of arbitrary order. Minimal constraints were shown to imply well-posedness of the finite element formulation along with the shape regularity constraints of chapter 3. Two Grönwall inequalities were introduced: one to bound the local truncation error of each time partition and the other to aggregate the truncation errors over all partition. These results were used in conjunction with the discrete Galerkin orthogonality of the finite element solution to prove the symmetric error estimate of theorem 2, which was the main result of this chapter and the motivation for the error analysis of the thesis.

In chapter 5, the finite element aspects of the discrete formulation were emphasized, as this framework leads to the symmetric error estimate. In chapter 6, however, the focus shifted to the method of lines approach used for the moving finite element method. Namely, the application of Runge-Kutta time integration schemes for solving the semi-discrete formulation was discussed the existence and uniqueness of a finite element solution was verified. We also explored some of the effects that a time integration scheme could have on the finite element solution. An error estimate was proven for piecewise quadratic finite element solutions computed using TR-BDF for time integration. Unfortunately, this proof relied on some properties specific to the TR-BDF integration scheme and no general error estimate could be given for a larger class of Runge-Kutta methods.

The content of chapter 7 was the implementation of these moving mesh methods. A linear convection-dominated test problem and a convection-diffusion-reaction test problem were used to validate the expectations of the theory established in the previous chapters. Both test problems had solutions computed more accurately using moving mesh, with the benefits being most pronounce with high resolution in the spatial discretization and relatively longer time steps. This falls in line with the predictions of the theory and validates the benefits of moving fi-

nite elements for tensor product quadratic finite element spaces. Simulations were also tested using interpolation rather than $\mathcal{L}_2$-projection; when the time step was relatively close in size or larger than the size of the spatial elements, interpolation lead to a significant speedup for computing the solution without any deterioration in accuracy. However, if a coarse spatial mesh was used in conjunction with many short time steps, the interpolation error would build over the time partitions and severely impact the accuracy of the computed solution. Further experiments tested requirement for non-truncating quadrature rule to define the placement of the collocation nodes and revealed that the test problems achieved the lowest error using Richardson's TR-BDF collocation node, which does not correspond to a non-truncating quadrature rule.

A simple adaptive meshing scheme, based on h-adaptive refinement using derivative recovery, was proposed and implemented with somewhat satisfactory results. Overall, the adaptive method seemed to proportion nodes effectively throughout the domain, yet the mechanism that predicted the number of spatial nodes to be used in a time partition did not perform quite as well as expected. As a final experiment, Burger's equation was used as a test problem to evaluate the efficacy of the moving finite element method for a nonlinear equation. Numerically induced oscillations are often found near the boundary of shock layers for problems like Burger's equation when the time steps are taken to be too long. These oscillations were dampened quicker when using moving meshes rather than non-moving meshes, indicating that moving meshes permit taking larger time steps while maintaining high resolution at the faces of propagating shock layer. For both the basic and adaptive moving mesh techniques, the spatial nodes congregated near the face of the shock layer and tracked it as it swept through the spatial domain.

Some interesting aspects remain unexplored for these moving finite element methods. For example, it would be interesting to understand whether the error analysis given in this thesis can extend to more general differential equations than linear convection-diffusion problems. Another interesting problem would investigate elements that are specifically designed to reside near the boundary of the finite element space that are specifically designed to improve the accuracy of the

computed solution near inflow and outflow boundaries. This type of research could lead to great improvements in computing accurate solutions for problems where refined structures enter and propagate through the spatial domain, or refract off of the boundaries of the domain. Furthermore, the software package PLTMG [16] serves as a great launch pad for implementing moving finite elements for problems with two dimensional spatial domains, as it possesses advanced adaptive meshing techniques, data structures, and solvers for elliptic PDEs. Consequently, moving finite element solvers can be implemented by connecting these subroutines with a time integration scheme.

Applying finite elements to time-dependent problems leads to some interesting difficulties. Simply discretizing the space-time domain as a higher dimensional spatial problem seems to be an oversimplification for these problems, and often a mix of finite elements and time integration via finite differences is employed. This is exactly the approach of our moving finite element methods, though this approach often leads to difficult error analysis, due to the fundamental differences between finite element theory and the analysis of time integration schemes. This thesis sought to overcome these difficulties by defining a space-time tensor product finite element space that is compatible with the method of lines approach. In chapter 5, we found a symmetric error estimate for the finite element solution, though this came at the cost of constraining the time integration scheme. In chapter 6, we generalized the moving finite element method to encompass more general time integration schemes, though this cost the much sought after symmetric error estimate. It is interesting to wonder whether there is some way to reconcile this natural fit of space and time discretizations in an error analysis, and it is this thought that will continue to propel these methods into the work of future research.

Chapter 8, in part, is in preparation for submission for publication of this material. Bank, Randolph E. The dissertation author is the primary investigator and author of this material.

Appendix A

Final Notes

In this final portion of the thesis, we include some additional analysis for some common finite element discretizations using space-time tensor moving elements. Analysis for the case of linear elements with one spatial dimension is straightforward, since one needs only to preserve the positivity of the spatial diameter, $\Delta x_k(t) > 0$, for all time t in the respective time partition, which leads to the simple constraint that Δx_k is positive at the beginning and end of the time partition. Namely, we consider the non-degeneracy of tensor quadratic elements with one spatial dimension, where practical bounds are derived that correspond to two distinct approaches for selecting a moving mesh. Also, we find a necessary and sufficient condition for tensor linear elements with two spatial dimensions to be non-degenerate.

A.1 Tensor quadratic elements with one spatial dimension

We consider two problems that verify the non-degeneracy of a quadratic tensor element. In the first problem, we are given a trapezoidal element, which can come from a linear tensor element, and we want to understand how much we can shift the nodes at the middle collocation node before the element degenerates, while keeping the element at the beginning and end of the time partition fixed. The second problem corresponds fixing the element at the beginning of the time partition and the midpoint collocation point; this problem comes from extrapolating the mesh motion from the mid-step collocation point and analyzing how much one may perturb the extrapolated element before degeneracy occurs.

A.1.1 Mid-step perturbation of a trapezoidal element

For this problem, we let Δx_0 and Δx_1 represent the *fixed* lengths of the spatial element at the beginning and end of the time partition. Thus, a trapezoidal element with these lengths would be of the form

$$\Delta x_{TR}(\tilde{t}) = \Delta x_1 \tilde{t} + \Delta x_0 (1 - \tilde{t}).$$

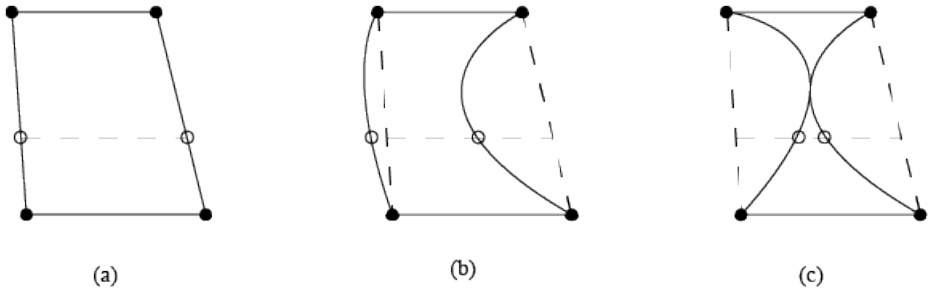

Figure A.1: Three tensor product quadratic elements. The element labeled (a) has not been perturbed at the middle collocation node. Element (b) has been perturbed at the middle collocation node, though it remains non-degenerate. Element (c) is an example of an element that degenerates because the middle collocation nodes move too close to each other. Notice how the intersection does not necessarily occur at a collocation time slice.

Let $0 < \epsilon < 1$, where ϵ denotes the relative position of the mid-step collocation node within the time partition. The problem we consider here can be characterized as finding the values of δx such that

$$\Delta x(\tilde{t}) = \Delta x_1 \tilde{t} + \Delta x_0(1-\tilde{t}) + \delta x \frac{\tilde{t}(1-\tilde{t})}{\epsilon(1-\epsilon)} > 0, \tag{A.1}$$

for all $\tilde{t}$ in $[0,1]$. The addition of this bump function corresponds to shrinking ($\delta x < 0$) or expanding ($\delta x > 0$) the element at the mid-step. This is illustrated by figure A.1. Note that (A.1) is unaffected if we merely translate the element at the mid-step, as this does not change the length the the spatial element at any time.

Since $\tilde{t}(1-\tilde{t})/\epsilon(1-\epsilon) > 0$ for $0 < \tilde{t} < 1$, we see from (A.1) that $\Delta x(\tilde{t}) > 0$ whenever $\delta x > 0$. We rewrite (A.1) as

$$\Delta x(\tilde{t}) = -\frac{1}{\epsilon(1-\epsilon)}\delta x\, \tilde{t}^2 + \left(\frac{1}{\epsilon(1-\epsilon)}\delta x + \Delta x_1 - \Delta x_0\right)\tilde{t} + \Delta x_0$$

and compute the discriminant to find the values of δx when zeros are introduced to the element diameter function $\Delta x(\tilde{t})$. The discriminant is given by

$$\left(\frac{1}{\epsilon(1-\epsilon)}\delta x + \Delta x_1 - \Delta x_0\right)^2 - \frac{4}{\epsilon(1-\epsilon)}\delta x \Delta x_0$$
$$= \left(\frac{1}{\epsilon(1-\epsilon)}\delta x + \Delta x_1 + \Delta x_0\right)^2 - 4\Delta x_1 \Delta x_0,$$

which we set equal to zero to find which values of δx correspond to introducing roots to $\Delta x(\tilde{t})$. We find that there are no roots when

$$\sqrt{\Delta x_1 \Delta x_0} > \left| \frac{\delta x}{2\epsilon(1-\epsilon)} + \frac{\Delta x_1 + \Delta x_0}{2} \right|,$$

which implies

$$-2\epsilon(1-\epsilon)\left(\frac{\Delta x_1 + \Delta x_0}{2} + \sqrt{\Delta x_1 \Delta x_0}\right) < \delta x < 2\epsilon(1-\epsilon)\left(-\frac{\Delta x_1 + \Delta x_0}{2} + \sqrt{\Delta x_1 \Delta x_0}\right).$$

Upon inspection, we see that the upper bound on δx corresponds to having roots outside the unit interval, which do not concern us. Therefore, if

$$\partial x > -2\epsilon(1-\epsilon)\left(\frac{\Delta x_1 + \Delta x_0}{2} + \sqrt{\Delta x_1 \Delta x_0}\right), \tag{A.2}$$

the element does not degenerate at any given time within the time partition.

A.1.2 End-step perturbation of an extrapolated trapezoidal element

Suppose now that the length of the spatial element is fixed at the beginning and the mid-step of the time partition. Let $0 < \epsilon < 1$, where ϵ denotes the relative position of the mid-step collocation node, as before. We assume for simplicity that the extrapolated element does not degenerate in time; that is,

$$\Delta x(\tilde{t}) = \frac{1}{\epsilon}\Delta x_{1/2}\tilde{t} + \frac{1}{\epsilon}\Delta x_0(\epsilon - \tilde{t}) > 0$$

for $0 \leq \tilde{t} \leq 1$. Then, we can add a bump function at the end of the time step and want to find the values of δx such that

$$\Delta x(\tilde{t}) = \frac{1}{\epsilon}\Delta x_{1/2}\tilde{t} + \frac{1}{\epsilon}\Delta x_0(\epsilon - \tilde{t}) + \delta x \frac{\tilde{t}(\tilde{t}-\epsilon)}{\epsilon(1-\epsilon)} > 0, \tag{A.3}$$

for all $\tilde{t}$ in $[0, 1]$. To keep the element form degenerating at the end of the time step, we require

$$\delta x > -\frac{1}{\epsilon}\big(\Delta x_{1/2}\tilde{t} - (1-\epsilon)\Delta x_0\big). \tag{A.4}$$

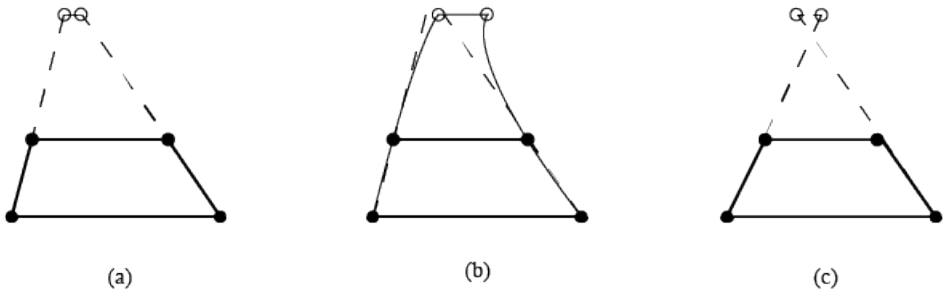

Figure A.2: Three tensor product quadratic elements. Element (a) is an element that has can be successfully extrapolated from the mid-step collocation node. Element (b) has been perturbed at the end collocation node, though it remains non-degenerate. The third element (c) is an example of an element that cannot be extrapolated; such elements should not be permitted into time partitions that are built one collocation time slice at a time.

Element (c) in figure A.2 is an example of an element that does not meet this criterion.

As before, we find the discriminant of the quadratic function $\Delta x(\tilde{t})$ and determine when it is negative, corresponding to the case when the element does not degenerate:

$$0 > \left(-\frac{\epsilon}{1-\epsilon}\delta x + \frac{1}{\epsilon}\Delta x_{1/2} - \frac{1}{\epsilon}\Delta x_0\right)^2 - \frac{4}{1-\epsilon}\delta x \Delta x_0$$
$$= \left(-\frac{\epsilon}{1-\epsilon}\delta x + \frac{1}{\epsilon}\Delta x_{1/2} + \frac{1}{\epsilon}\Delta x_0\right)^2 - \frac{4}{\epsilon^2}\Delta x_{1/2}\Delta x_0.$$

This is equivalent to

$$\frac{2(1-\epsilon)}{\epsilon^2}\left(\frac{\Delta x_{1/2} + \Delta x_0}{2} - \sqrt{\Delta x_{1/2}\Delta x_0}\right) < \delta x$$
$$< \frac{2(1-\epsilon)}{\epsilon^2}\left(\frac{\Delta x_{1/2} + \Delta x_0}{2} + \sqrt{\Delta x_{1/2}\Delta x_0}\right).$$

This time, the lower bound for δx corresponds to $\Delta x(\tilde{t})$ having roots outside the unit interval, until (A.4) is violated. Accordingly, when perturbing the end-step of

an extrapolated element, the perturbation must satisfy

$$-\frac{1}{\epsilon}\big(\Delta x_{1/2}\tilde{t}-(1-\epsilon)\Delta x_0\big) < \delta x < \frac{2(1-\epsilon)}{\epsilon^2}\left(\frac{\Delta x_{1/2}+\Delta x_0}{2}+\sqrt{\Delta x_{1/2}\Delta x_0}\right). \quad \text{(A.5)}$$

A.2 Tensor product linear elements of two spatial dimensions

Now, we find a necessary and sufficient condition for characterizing non-degenerate linear prism elements that arise when employing linear discretizations in space and time. For this problem, the isoparametric map is affine in space and changes linearly in time; let

$$\mathcal{A}_0^0 = \begin{bmatrix} \Delta x_1(0) & \Delta x_2(0) \\ \Delta y_1(0) & \Delta y_2(0) \end{bmatrix} \quad \text{and} \quad \mathcal{A}_1^1 = \begin{bmatrix} \Delta x_1(1) & \Delta x_2(1) \\ \Delta y_1(1) & \Delta y_2(1) \end{bmatrix},$$

represent the spatial Jacobian of the isoparametric map at times $\tilde{t}=0$ and $\tilde{t}=1$, respectively. Then, the Jacobian matrix at time $\tilde{t}$, where $0 \le \tilde{t} \le 1$ is given by the convex combination $\mathcal{A}(\tilde{t}) = \mathcal{A}_1^1\tilde{t} + \mathcal{A}_0^0(1-\tilde{t})$.

The degeneration of this element is characterized by $\det\big(\mathcal{A}(\tilde{t})\big) = 0$ for some $\tilde{t}$ in the unit interval. Figure 3.3 in chapter 3 depicts a linear element that degenerates at the mid-step from a twisting motion. Therefore, we make the necessary assumption that $\text{sign}\big(\det(\mathcal{A}_0^0)\big) = \text{sign}\big(\det(\mathcal{A}_1^1)\big)$, to avoid degeneracy at some intermediate time, which would otherwise follow from the intermediate value theorem. We make use of this by calculating the determinant of $\mathcal{A}(\tilde{t})$ and finding the conditions that ensure that the determinant has no roots in $[0,1]$.

To easily represent $\det\big(\mathcal{A}(\tilde{t})\big)$, we introduce the notation

$$\mathcal{A}_1^0 = \begin{bmatrix} \Delta x_1(0) & \Delta x_2(0) \\ \Delta y_1(1) & \Delta y_2(1) \end{bmatrix} \quad \text{and} \quad \mathcal{A}_0^1 = \begin{bmatrix} \Delta x_1(1) & \Delta x_2(1) \\ \Delta y_1(0) & \Delta y_2(0) \end{bmatrix},$$

which are simply the matrices $\mathcal{A}_0^0$ and $\mathcal{A}_1^1$ with alternating rows borrowed from $\mathcal{A}_0^0$

and $\mathcal{A}_1^1$. Taking the determinant of $\mathcal{A}(\tilde{t})$, one obtains

$$\begin{aligned}\det\big(\mathcal{A}(\tilde{t})\big) &= \det(\mathcal{A}_1^1)\tilde{t}^2 + \Big(\det(\mathcal{A}_0^1) + \det(\mathcal{A}_1^0)\Big)\tilde{t}(1-\tilde{t}) + \det(\mathcal{A}_0^0)(1-\tilde{t})^2 \\ &= \Big(\det(\mathcal{A}_1^1) + \det(\mathcal{A}_0^0) - \big(\det(\mathcal{A}_0^1) + \det(\mathcal{A}_1^0)\big)\Big)\tilde{t}^2 \\ &\quad + \Big(\det(\mathcal{A}_0^1) + \det(\mathcal{A}_1^0) - 2\det(\mathcal{A}_0^0)\Big)\tilde{t} + \det(\mathcal{A}_0^0),\end{aligned}$$

which can be verified directly. In the case where $\operatorname{sign}\big(\det(\mathcal{A}_0^0)\big) = \operatorname{sign}\big(\det(\mathcal{A}_1^1)\big) > 0$, we know that there are no roots for $0 \le \tilde{t} \le 1$, if $\det(\mathcal{A}_1^1) + \det(\mathcal{A}_0^0) < \det(\mathcal{A}_0^1) + \det(\mathcal{A}_1^0)$ by the concavity of the quadratic function. Likewise, if $\operatorname{sign}\big(\det(\mathcal{A}_1^1)\big) = \operatorname{sign}\big(\det(\mathcal{A}_1^1)\big) < 0$, there are no real roots when $\det(\mathcal{A}_1^1) + \det(\mathcal{A}_0^0) > \det(\mathcal{A}_0^1) + \det(\mathcal{A}_1^0)$.

To find where the roots of this quadratic polynomial occur, we apply the quadratic formula and find that there are no real roots when

$$\begin{aligned}0 > &\Big(\det(\mathcal{A}_0^1) + \det(\mathcal{A}_1^0) - 2\det(\mathcal{A}_0^0)\Big)^2 \\ &\quad - 4\det(\mathcal{A}_0^0)\Big(\det(\mathcal{A}_1^1) + \det(\mathcal{A}_0^0) - \big(\det(\mathcal{A}_0^1) + \det(\mathcal{A}_1^0)\big)\Big) \\ &\quad = \big(\det(\mathcal{A}_0^1) + \det(\mathcal{A}_1^0)\big)^2 - 4\det(\mathcal{A}_0^0)\det(\mathcal{A}_1^1).\end{aligned}$$

The equivalent condition for when $\det(\mathcal{A}(\tilde{t}))$ has any real roots is

$$\sqrt{\det(\mathcal{A}_0^0)\det(\mathcal{A}_1^1)} > \frac{|\det(\mathcal{A}_0^1) + \det(\mathcal{A}_1^0)|}{2}.$$

Note, however, that this is too restrictive since we are not concerned with roots outside the unit interval; by the above concavity (convexity) arguments of $\det(\mathcal{A}(\tilde{t}))$, we see that a necessary and sufficient condition for this element to be non-degenerate is

$$\operatorname{sign}\big(\det(\mathcal{A}_0^0)\big)\sqrt{\det(\mathcal{A}_0^0)\det(\mathcal{A}_1^1)} > -\operatorname{sign}\big(\det(\mathcal{A}_0^0)\big)\frac{|\det(\mathcal{A}_0^1) + \det(\mathcal{A}_1^0)|}{2}. \quad \text{(A.6)}$$

Bibliography

[1] S. Adjerid, J.E. Flaherty, *A local refinement finite-element method for two-dimensional parabolic systems*, SIAM J. Sci. and Stat. Comput., 9 (1988), pp. 792–811.

[2] S. Adjerid, J.E. Flaherty, *A moving finite element method with error estimation and refinement for one-dimensional time dependent partial differential equations*, SIAM J. Numer. Anal., 23 (1986), pp. 778–796.

[3] S. Adjerid, J. E. Flaherty, *A moving-mesh finite element method with local refinement for parabolic partial differential equations*, Comp. Methods Appl. Mech. Engrg., 55 (1986), pp. 3–26.

[4] R. Alexander, *Diagonally Implicit Runge-Kutta Methods for Stiff O.D.E.'s*, SIAM J. Numer. Anal., 14 (1977), pp. 1006–1021.

[5] D.C. Arney, J.E. Flaherty, *An adaptive local mesh refinement method for time-dependent partial differential equations*, Applied Numer. Math., 5 (1989), pp. 257–274.

[6] D.N. Arnold and G. Awanou, *The serendipity family of finite elements*, Foundations Comp. Math., 11 (2011), pp. 337-344.

[7] S.C. Brenner and L.R. Scott, *Mathematical theory of finite elements*, Springer-Verlag New York, LLC, 2008.

[8] D.N. Arnold, F. Brezzi, B. Cockburn, L.D. Marini, *Unified analysis of discontinuous Galerkin methods for elliptic problems*, SIAM J. Numer. Anal., 39 (2002), pp. 1749–1779.

[9] I. Babuška and M.R. Dorr, *Error Estimates for the h and p versions of the finite element method*, Numer. Math., 37 (1981), pp. 2567–2577.

[10] I. Babuška and J.M. Melenl, *The partition of unity finite element method*, Int. J. Numer. Meth. Engng., 40 (1997), pp. 727–758.

[11] I. Babuška, B. Szavo, and I. Katz, *The p version of the finite element method*, SIAM J. Numer. Anal., 18 (1981), pp. 515–545.

[12] M.J. Baines, *Moving Finite Elements*, Oxford University Press Inc., New York, NY, 1994.

[13] M.J. Baines, M.E. Hubbard, P.K. Jimack, *A moving mesh finite element algorithm for the adaptive solution of time-dependent PDEs with moving boundaries*, submitted to Elsevier Science, 2004.

[14] M.J. Baines, M.E. Hubbard, P.K. Jimack, *Velocity-based moving mesh methods for nonlinear partial differential equations*, manuscript in preparation.

[15] M.J. Baines and K. Miller, *Least squares moving finite elements*, IMA J. Numer. Anal., 21 (2001), 621–642.

[16] R.E. Bank, *PLTMG: A software package for solving elliptic partial differential equations user's guide 11.0*, Department of Mathematics, University of California, San Diego, CA, 2012.

[17] R.E. Bank and H. Nguyen, *hp Adaptive finite elements based on derivative recovery and superconvergence*, Comput. Vis. Sci., 6 (2012), pp. 287-299.

[18] R.E. Bank and R.F. Santos, *Analysis of some space-time finite element methods*, SIAM J. Numer. Anal., 30 (1993), pp. 1–18.

[19] R.E. Bank, A.H. Sherman, A. Weiser, *Some refinement algorithms and data structures for regular local mesh refinement*, Scientific Computing (Applications of Mathematics and Computing to the Physical Sciences) (R. S. Stepleman, ed.), North-Holland (1983), pp. 3–17.

[20] R.E. Bank and R.K. Smith, *Mesh smoothing using a posteriori error estimates*, SIAM J. Numer. Anal., 34 (1997), pp. 979–997.

[21] R.E. Bank and A. Weiser, *Some a posteriori error estimators for elliptic partial differential equations*, Math. Comp., 44 (1985), pp. 283–301.

[22] R.E. Bank and H. Yserentant, *On the $\mathcal{H}^1$-stability of the $\mathcal{L}_2$-projection onto finite element spaces*, preprint, 2012.

[23] R.E. Bank et al., *Transient simulation of silicon devices and circuits*, IEEE Trans. Computer-Aided Design, CAD-4 (1985), pp. 436–451.

[24] Belytschko, Y. Y. Lu, L. Gu, *Element-free Galerkin methods*, Int. J. Numer. Meth. Engng., 37 (1994), pp. 229–256.

[25] A. Bermudez, M.E. Vazquez, *Upwind Methods for Hyperbolic Conservation Laws with Source Terms*, Comput. Fluids, 23 (1994), pp. 1049–1071.

[26] G. Birkhoff, M.H. Schultz, and R.S. Varga, *Piecewise Hermite Interpolation in One and Two Variables with Applications to Partial Differential Equations*, Numerische Mathematik, 11 (1968), pp. 232-256.

[27] Brent, Voller, Reid, *Enthalpy-porosity technique for modeling convection-diffusion phase change: application to the melting of a pure metal*, Numer. Heat Transfer, 13 (1988), pp. 297–318.

[28] N.N. Carlson, K. Miller, *Design and Application of a Gradient-Weighted Moving Finite Element Code I: in 1 dimension*, SIAM J. Sci. Comput., 19 (1998), pp. 728–765.

[29] N.N. Carlson, K. Miller, *Design and Application of a Gradient-Weighted Moving Finite Element Code II: In Two Dimensions*, SIAM J. Sci. Comput., 19 (1998), pp. 766–798.

[30] M.A. Celia et al., *An Eulerian-Lagrangian localized adjoint method for the advection-diffusion equation*, Advances in Water Resources, 13 (1990), pp. 187–206.

[31] S.R. Choudhury, *Waveform relaxation techniques for linear and nonlinear diffusion equations*, J. Comput. Appl. Math., 42 (1992), pp. 253–267.

[32] S.R. Choudhury, *Multi-rate numerical methods for diffusion problems*, Comm. Numer. Methods Engrg., 9 (1993), pp. 1–8.

[33] R. Čiegis and N. Tumanova, *Parallel predictor-corrector schemes for parabolic problems on graphs*, Comput. Methods Applied Math., 10 (2010), pp. 275–282.

[34] R. Codina, *Comparison of some finite element methods for solving the diffusion-convection-reaction equation*, Comp. Methods Appl. Mech. Engrg., 156 (1998), pp. 185–210.

[35] R. Courant, E. Isaacson, and M. Rees, *On the solution of nonlinear hyperbolic differential equations by finite differences*, Comm. Pure and Applied Math., v (1952), pp. 243–255.

[36] S. Dharmaraja, G. Strang, Y. Wang, *Optimal stability for trapezoidal-backward difference split-steps*, IMA J. Numer. Anal., 30 (2010), pp. 141–148.

[37] T. Dupont, *Mesh Modification for Evolution Equations*, Math. Comp., 39 (1982), pp. 85–107.

[38] J. Douglas and T. Dupont, *Galerkin Methods for Parabolic Equations*, SIAM J. Numer. Anal., 7 (1970), pp. 575–626.

[39] J. Douglas and T. Dupont, *Interior Penalty Procedures for Elliptic and Parabolic Galerkin Methods*, Computing Methods in Applied Sciences, Lecture Notes in Physics, 58 (1976), pp. 207–216.

[40] T. Dupont and I. Mogultay, *A symmetric error estimate for a discrete-time parabolic moving mesh*, manuscript in preparation, 2011.

[41] F. Stratmann and E. R. Whitby, *Numerical solution of aerosol dynamics for simultaneous convection, diffusion and external forces*, Journal of Aerosol Science, 20 (1989), pp. 437–440.

[42] A. Gillete, A. Rand, and C. Bajaj, *Error Estimates for Generalized Barycentric Interpolation*, Adv. Comput. Math., 37 (2012), pp. 417–439.

[43] M.E. Hosea, L.F. Shampine, *Analysis and implementation of TR-BDF2*, Applied Numer. Math., 20 (1996), pp. 21–37.

[44] W. Hundsdorfer, J. G. Verwer, *Numerical solution of time-dependent advection-diffusion-reaction equations*, Springer-Verlag Berlin Heidelberg New York, Germany, 2003.

[45] J.M. Hyman, *Moving mesh methods for partial differential equations*, in Mathematics Applied to Science, Academic Press, Boston, MA, 1986.

[46] Idelsohn, E. Oñate, N. Calvo, F. Del Pin, *The meshless finite element method*, Int. J. Numer. Meth. Engng., 58 (2003), pp. 893–912.

[47] I.B. Jacques, *Predictor-Corrector Methods for parabolic PDEs*, International J. Numer. Methods Engrg., 19 (1983), pp. 451–465.

[48] J. Jia, X. Hu, J. Xu, C. Zhang, *Effects of integration and adaptivity for the Euler-Lagrangian method*, J. Comp. Math., 29 (2011), pp. 367–395.

[49] V. John, S. Kaya, and W. Layton, *A two-level variational multiscale method for convection-dominated convection-diffusion equations*, Comput. Methods Appl. Mech. Engrg., 195 (2006) 4594–4603.

[50] A.P. Kuprat, *Creation and Annihilation of Nodes for the Moving Finite Element Method*, Ph.D. Thesis, University of California, Berkeley, CA, 1992.

[51] A. Kurganov and Y. Liu, *New adaptive artificial viscosity method for hyperbolic systems of conservation laws*, J. Comput. Physics, 231 (2012), pp. 8114–8132.

[52] S. Larsson and V. Thomée, *Partial differential equations with numerical methods*, Springer-Verlag Berlin Heidelberg New York, Germany, 2003.

[53] Y. Liu, R.E. Bank, T.F. Dupont, S. Garcia, and R. Santos, *Symmetric Error estimates for moving mesh mixed methods for advection-diffusion equations*, SIAM J. Numer. Anal., 40 (2003), pp. 2270–2291.

[54] C.B. Macdonald, S. Gottlieb, and S.J. Ruuth, *A Numerical Study of Diagonally Split Runge-Kutta Methods for PDEs with Discontinuities*, J. Sci. Comput., 35 (2008), pp. 89–112.

[55] K. Miller, *Moving Finite elements. II*, SIAM J. Numer. Anal., 18 (1981), 1033–1057.

[56] K. Miller, *Stabilized Moving Finite Elements for Convection Dominated Problems*, J. Sci. Comput., 24 (2005), pp. 163–182.

[57] K. Miller and R.N. Miller, *Moving Finite elements. I*, SIAM J. Numer. Anal., 18 (1981), 1019–1032.

[58] S.P. Neuman, *Adaptive Eulerian-Lagrangian finite element method for advection-dispersion*, International J. Numer. Methods Engrg., 20 (1984), 321–337.

[59] S.E. Notaris, *The error norm of Gauss-Radau quadrature formulae for Chebyshev weight functions,* BIT Numer. Math., 50 (2010), pp. 123–147.

[60] Osher and S. Chakravarthy, *Upwind schemes and boundary conditions with applications to Euler equations in general geometries*, J. Comput. Physics, 50 (1983), pp. 447–481.

[61] J. Pudykiewicz and A. Staniforth, *Some properties and comparative performance of the semi-Lagrangian method of Robert in the solution of the advection-diffusion equation*, Atmosphere-Ocean 22 (3) 1984, 283–308.

[62] A. Quarteroni, R. Sacco, F. Saleri, *Numerical Mathematics*, Springer Berlin Heidelberg, 2010.

[63] W.H. Reed and T.R. Hill, *Triangular mesh methods for the neutron transport equation*, Los Alamos Scientific Laboratory, Los Alamos, NM, 1973.

[64] H.G. Roos, M.Stynesand L.Tobiska, *Numerical methods for singularly perturbed differential equations*, Springer Verlag Berlin Heidelberg, Germany, 1996.

[65] R.F. Santos, *Moving Finite Element Method*, Ph.D. Thesis, Department of Mathematics, University of California, San Diego, CA, 1992.

[66] A. Sard, *Integral representations of remainders*, Duke Math. J., 15 (1948), pp. 333–345.

[67] P. Šolín, K. Segeth, I. Doležel, *Higher-Order Finite Element Methods*, CRC Press LLC, Boca Raton, FL, 2004.

[68] P.J. Van der Houwen and B.P. Sommeijer, *On the Stability of Predictor-Corrector Methods for Parabolic Equations with Delay*, IMA J. Numer. Anal., 6 (1986), pp. 1–23.

[69] J. VonNeumann and R. Richtmyer, *A method for the numerical calculation of hydrodynamic shocks*, J. Appl. Phys., 21 (2009), pp. 232–237.

[70] M. Wicke, M. Botsch, and M. Gross, *A finite element method on convex polyhedra*, Computer Graphics Forum, 26 (2007), pp. 355–364.

[71] G. Yagawa and T. Yamada, *Free mesh method: A new meshless finite element method*, Comput. Mech., 18 (1996), pp. 383–386.

[72] W. Ying, C. S. Henriquez, and D. J. Rose, *Composite backward differentiation formula: and extension of the TR-BDF2 scheme*, Submitted to Applied Numerical Mathematics, 2009.