HIGH ORDER FINITE ELEMENTS ON TETRAHEDRALS WITH LOCAL MASS MATRIX INVERSION

A THESIS

BY

MICHAEL J. BRAZELL

DEPARTMENT OF MECHANICAL AND AERONAUTICAL ENGINEERING

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE (MECHANICAL ENGINEERING)

December 2008

Accepted by the Graduate School
The undersigned have examined the thesis entitled

**High Order Finite Elements on Tetrahedrals with Local Mass Matrix Inversion**

presented by Michael J. Brazell, a candidate for the degree of Master of Science, and hereby certify that it is worthy of acceptance.

Date

ADVISOR
Dr. Brian T. Helenbrook

Date

CO-ADVISOR
Dr. Daniel T. Valentine

Dr. Scott R. Fulton
Abstract

A high order tetrahedral finite element method has been developed. Tetrahedrals were chosen because they enable automatic mesh generation and adaption methods for complex three dimensional geometries. The difficulty with high-order methods on tetrahedrals is that the variational integration of a time derivative term leads to the development of a “mass” matrix that is computationally expensive to invert. To overcome this problem, a new basis was developed to decouple the mass matrix and thus allow local inversion. This inversion process is \( p - 1 \) accurate and can be used in conjunction with a Diagonally Implicit Runge Kutta (DIRK) scheme to perform “dual-time-stepping” simulations in time. To test this approach, the unsteady heat equation was implemented. To accelerate convergence in pseudo-time p-multigrid was used and was shown to drastically improve the convergence making it independent of mesh size and only slightly dependent on the time step and polynomial order of the basis.
Contents

Abstract iv

1 Introduction 1

2 Local Mass Matrix Inversion on Tetrahedrals 4
  2.1 Introduction ......................................................... 4
  2.2 Mass Matrix ....................................................... 4
  2.3 Standard Tetrahedral ............................................... 8
  2.4 Tetrahedral Space ................................................ 9
  2.5 Basis ............................................................... 11
  2.6 New Orthogonal Basis ............................................. 13
    2.6.1 Mode Rearrangement ......................................... 14
    2.6.2 Mass Matrix .................................................. 15
    2.6.3 Vertex Results ............................................... 16
    2.6.4 Edge Results ................................................ 19
    2.6.5 Face Results ................................................ 21
    2.6.6 Interior Results ............................................. 22
    2.6.7 Vertex Ball .................................................... 22
  2.7 Accuracy of $p - 1$ Mass Matrix Inversion .................. 26
  2.8 Iterative Performance of $p - 1$ Mass Matrix Inversion .... 28
3 Unsteady Heat Equation

3.1 Introduction ........................................................................... 32
3.2 Formulation ............................................................................ 32
  3.2.1 Variational Form ............................................................... 33
  3.2.2 Time Advancement ............................................................. 33
  3.2.3 $p$-Multigrid ................................................................. 34
3.3 Verification ............................................................................. 35
  3.3.1 Temporal Accuracy ........................................................... 35
  3.3.2 Spatial Accuracy ............................................................... 36
  3.3.3 Iterative Convergence ....................................................... 37
3.4 Conclusion .............................................................................. 39

4 Conclusions ............................................................................... 42

Bibliography ............................................................................... 43

5 Appendix A ............................................................................. 46

  5.1 Jacobi Polynomial .............................................................. 46
List of Figures

1.1 4th order GL nodal basis ................................................. 2
2.1 Linear Finite Elements ..................................................... 6
2.2 Mass Matrix for three linear elements in 1D .............................. 6
2.3 Mass matrix for $p = 4$ basis ............................................ 7
2.4 Mass matrix for $p = 8$ basis ............................................ 8
2.5 New vertex 1 function ..................................................... 19
2.6 Constraints-degrees of freedom vs $p$ for edge modes. .................. 20
2.7 Constraints-degrees of freedom vs $p$ for face modes. .................. 22
2.8 Contours of new face mode. ............................................. 23
2.9 A set of tetrahedrals with a common vertex. ............................. 23
2.10 Vertex ball: red = spokes, blue = shell. ............................... 24
2.11 Decoupled vertex ball. .................................................... 24
2.12 Constraints-dofs vs $p$ for vertex ball. .................................. 25
2.13 Mesh used in resolution studies ......................................... 27
2.14 $L_2$ error vs Mesh size ($h$) for $p = 2$ ................................. 28
2.15 $L_2$ error vs Mesh size ($h$) for $p = 3$ ................................. 28
2.16 Residual vs Iteration for $p = 1,2,3$ .................................... 29
2.17 Residual vs Iteration for $p = 2,3$ ...................................... 30
3.1 $L_2$ Error vs Time Step ................................................... 36
3.2 $L_2$ Error vs Mesh Size for $p = 1, 2, 3$  

3.3 cycle/decade vs $\Delta t^{-1}$ for $p = 1, 2, 3$ on fine mesh  

3.4 cycle/decade vs $h$ for $p = 1, 2, 3$ and $\Delta t^{-1} = 40$  

3.5 $p = 2$ p-MG Residual vs Iteration for $\Delta t^{-1} = 0, 40, 400$ on fine mesh  

3.6 $p = 3$ p-MG Residual vs Iteration for $\Delta t^{-1} = 0, 40, 400$ on fine mesh  

3.7 $p = 2$ p-MG Residual vs Iteration for three meshes at $\Delta t^{-1} = 400$  

3.8 $p = 3$ p-MG Residual vs Iteration for three meshes at $\Delta t^{-1} = 400$
Chapter 1

Introduction

The spectral element method (SEM) is a high order finite element method (FEM). SEM simulations require less degrees of freedom and are more computationally efficient than their lower order counterparts when a high accuracy is desired. Because of this, SEM methods have been popular for turbulent flows. Patera was one of the first to employ these methods on turbulent flow in simple geometry [1]. Fischer (his student) has since performed some very high Reynolds number (>30,000) direct numerical simulations (DNS) such as hairpin vortices induced by flow over a sphere, oscillatory boundary layers over a sphere, blood flow in the carotid artery, and LES simulations of lid driven cavities with Reynolds numbers of 12,000 [2, 3, 4].

SEM uses a high order nodal basis on quadrilaterals and hexahedrals. It hinges on the existence of the Gauss-Labotto (GL) integration rule. Gauss-Labotto points are used for both integration and for the nodes of the Lagrangian polynomial basis used in SEM. The inner product of the basis is orthogonal with respect to this integration rule, which leads to the ability to perform fast explicit simulations. This is absolutely necessary for turbulent calculations. Figure 1.1 shows an example of a one dimensional fourth order GL nodal basis.
Unfortunately, SEM is limited to quadrilaterals and hexahedrals and has not been implemented on triangles and tetrahedrals. This makes it difficult to apply in complicated geometries because automated mesh generation and adaption techniques typically use triangles and tetrahedrals. For tetrahedral meshes, adaptation techniques are well-developed and robust [5]. The obvious way to proceed is to develop a SEM for triangles and tetrahedrals, which will be referred to as an \( hp \)-Finite Element Method (FEM) (\( h \) stands for mesh refinement and let \( p \) be the polynomial degree). Unfortunately, this is not a trivial task. Helenbrook [6] has recently proven that a GL rule, equivalent to that used on a quad, does not exist on triangles or tetrahedrals. A GL rule will be defined as a set of nodes that can be used for both a \( 2p - 1 \) accurate integration and the creation of a nodal basis. Although triangles and tetrahedrals lack a GL integration rule, Helenbrook has found that a basis with orthogonality properties similar to that used in SEM does exist. Using techniques based on this discovery, Helenbrook [7], has implemented an \( hp \)-FEM code using triangles. He has successfully simulated multiphase flow including liquid sheets, impinging jets, falling drops, and coating flows with a high-order of accuracy. [8, 9, 10, 11].

The basis discovered by Helenbrook has properties similar to that of SEM. This new basis does not have a GL rule but does have the similar orthogonality properties making it
just as useful. In unsteady differential equations a mass matrix is formed by the variational integration of the time derivative term. This matrix must be inverted to advance in time. The new basis can be used to create a nearly orthogonal mass matrix. This can be approximated as diagonal and is then easily inverted. This makes explicit time advancement schemes feasible.

There have been many other attempts at finding other high order basis with properties similar to SEM on triangles and tetrahedrals [12, 13, 14, 15, 16]. One of the most popular are the Fekete points. This method yields an approximate diagonal mass matrix but suffers from accuracy problems. The most common method for inverting the mass matrix is using mass lumping. Mass lumping creates a diagonal matrix by making each diagonal term equal to the row sum of the mass matrix. This may lead to an unstable iterative inversion scheme and it is unclear how to implement this with a high order basis.

The specific research goal for this paper is to develop and apply a 3D high-order accurate finite element method on tetrahedrals to solve the unsteady heat equation through the new techniques developed by Helenbrook. This paper will broken into two major chapters. Chapter one will describe the tetrahedral basis and derivation of a new basis with properties similar to SEM. This new basis will be tested using a mass matrix inversion scheme and compared to other methods such as Fekete Points. The second chapter will focus on the formulation and verification of solutions to the unsteady heat equation using the new $hp$-FEM developed.
Chapter 2

Local Mass Matrix Inversion on Tetrahedrals

2.1 Introduction

As mentioned in the introduction, in FEM simulations of unsteady differential equations a “mass matrix” is formed by the variational integration of the time derivative term. Even for explicit time advancement schemes, this matrix must be inverted at every time step of the simulation. This matrix is coupled across the entire mesh and in higher dimensions can be computationally expensive to invert. In this chapter, we give further details on the mass matrix problem and then develop a new polynomial basis on tetrahedrals that allows one to avoid the difficulties of the mass matrix when performing 3D simulations on a tetrahedral.

2.2 Mass Matrix

To understand where the mass matrix is formed the one dimensional heat equation with no source term will be examined.

\[
\frac{\partial u}{\partial t} = k \frac{\partial^2 u}{\partial x^2}
\]  

(2.1)
Multiplying by a test function $v$ and integrating we arrive at the weighted integral form.

$$
\int_\Omega v \frac{\partial u}{\partial t} dx = k \int_\Omega v \frac{\partial^2 u}{\partial x^2} dx \tag{2.2}
$$

Integrating by parts on the right hand side to make it symmetric yields

$$
\int_\Omega v \frac{\partial u}{\partial t} dx = k \int_\Omega \frac{\partial v}{\partial x} \frac{\partial u}{\partial x} dx \tag{2.3}
$$

where $u$ and $v$ are represented by the coefficients $a$ and $b$ respectively and multiplied the basis $\phi$ where $N$ is the total number of unknowns.

$$
u = \sum_{i=0}^{N} a_i \phi_i, \quad v = \sum_{i=0}^{N} b_i \phi_i \tag{2.4}
$$

Substituting this into the equation 2.3 yields.

$$
b^T \int_\Omega \phi \phi^T dx \frac{\partial a}{\partial t} = b^T k \int_\Omega \frac{\partial \phi}{\partial x} \frac{\partial \phi^T}{\partial x} dx a \tag{2.5}
$$

simplifying this to

$$
b^T \left[ M \frac{\partial a}{\partial t} - K a \right] = 0 \tag{2.6}
$$

This must be true $\forall b$ and further simplifies to

$$
M \frac{\partial a}{\partial t} = K a \tag{2.7}
$$

Where $M$ is the mass matrix and $K$ is the stiffness matrix. The mass matrix is the integration of the outer product of the set of basis functions

$$
M_{i,j} = \int_\Omega \phi_i \phi_j d\Omega. \tag{2.8}
$$

For this example the time derivative can be approximated using a first order forward difference

$$
M \frac{a^{n+1} - a^n}{\Delta t} = K a^n, \tag{2.9}
$$

where $n$ represents the solution in time. Solving for $a^{n+1}$

$$
a^{n+1} = M^{-1} (Ma^n + \Delta t Ka^n) = a^n + \Delta t M^{-1} Ka^n \tag{2.10}
$$
shows that the inverse of the mass matrix is needed to advance in time even for explicit calculations. To show the structure of the mass matrix on a FEM mesh, a simple case of three linear elements is shown in figure 2.1. The elements are represented by E and the basis functions by $\phi$. These are typically called hat functions as for each element the adjacent linear vertex modes are forced to be continuous.

![Figure 2.1: Linear Finite Elements](image)

The structure of the mass matrix can be seen in figure 2.2. One dimensional linear finite elements create a mass matrix that is tri-diagonal. In higher dimensions the mass matrix bandwidth is dependent on the number of elements and for high order methods the mass matrix bandwidth is also dependent on the polynomial order which creates more basis functions per element.

The structure of high order tetrahedral mass matrices are shown in figures 2.3 and 2.4 for polynomial basis of $p = 4, 8$ where the total number of modes are 35 and 165 respectively. These are mass matrices for only one element and the global mass matrix would be much more complex.

$$
M = 
\begin{bmatrix}
\int \phi_1 \phi_1 & \int \phi_1 \phi_2 & 0 & 0 \\
\int \phi_2 \phi_1 & \int \phi_2 \phi_2 & \int \phi_2 \phi_3 & 0 \\
0 & \int \phi_3 \phi_2 & \int \phi_3 \phi_3 & \int \phi_3 \phi_4 \\
0 & 0 & \int \phi_4 \phi_3 & \int \phi_4 \phi_4
\end{bmatrix}
$$

![Figure 2.2: Mass Matrix for three linear elements in 1D](image)
For high order finite elements the mass matrix becomes computationally expensive to invert each time step. Instead of inverting, methods such as SEM use an approximate mass matrix inversion. The GL points are used for both integration and the basis. When performing a GL numerical integration it is accurate for polynomials of order $2N - 3$, where $N$ is the number of nodes. It can be shown that SEM relies on an inaccurate under-integration. For example in 1D there are $p + 1$ GL nodes making a nodal basis of order $p$. This means that this integration will be accurate to order $2(p + 1) - 3 = 2p - 1$. The maximum order polynomial integrated in the mass matrix is of order $2p$, which means that SEM uses an inexact integration. The SEM yields a diagonal mass matrix, which is trivial to invert. This mass matrix inversion will be exact if the function of order $p - 1$.

In the case of high order finite elements on tetrahedrals the mass matrix becomes a sparse matrix with a large bandwidth proportional to $(p + 1)^3 N_{total}^{2/3}$. Inverting such a matrix is computationally expensive and limits the efficiency of this approach. To avoid this a new basis will be derived that will allow local approximate inversion on each element similar to SEM. Helenbrook has developed a method that locally inverts the mass matrix for triangles. It lacks a GL integration rule similar to the SEM method but does have the approximate orthogonality property making it feasible to locally invert the mass matrix.
2.3 Standard Tetrahedral

Before deriving this new basis, we first need to give some fundamental information about tetrahedrals. A standard tetrahedral is defined in the figure below. The four vertices are labeled $V_1 - V_4$, the six edges are labeled $E_1 - E_6$, and the four faces are referenced by the index of the vertex directly opposite of each face. Face 1 is in the $t = -1$ plane, face 2 in the $s = -1$ plane, face 3 in the $r + s + t = -1$ plane, and face 4 in the $r = -1$ plane.
In some cases it will be convenient to use a mapping between a tetrahedral and a hexahedral. The transformation from tetrahedral coordinates to hexahedral coordinates is

\[
\begin{align*}
    x &= 2 \frac{(1 + r)}{(-s - t)} - 1, \\
    y &= 2 \frac{(1 + s)}{(1 - t)} - 1, \\
    z &= t.
\end{align*}
\] (2.11)

Similarly a backwards transformation can be derived. This is done so that numerical integration can be performed in hexahedral coordinates using Gauss quadrature in three dimensions [17].

2.4 Tetrahedral Space

The space of polynomials used on the the tetrahedral is given by \( T(p) = \{x^l y^m z^n | 0 \leq l, m, n; l + m + n \leq p \} \). The \( T(p) \) space can be broken down into groups; vertex, edge, face, and interior spaces. A vertex space is zero along the face opposite the vertex. For example, the vertex one function is zero along face one, defined by the plane \( t = -1 \). Vertex spaces are are shown in equations 2.12-2.15.

\[
\begin{align*}
    \mathcal{V}_1(p) &= (1 + t) T(p - 1) \quad (2.12) \\
    \mathcal{V}_2(p) &= (1 + s) T(p - 1) \quad (2.13) \\
    \mathcal{V}_3(p) &= (1 + r + s + t) T(p - 1) \quad (2.14) \\
    \mathcal{V}_4(p) &= (1 + r) T(p - 1) \quad (2.15)
\end{align*}
\]

An edge space is defined as being non-zero on the two adjacent faces to the edge and zero on the opposite faces. For example an edge one function is non-zero on faces one and two, and zero on faces three and four. The six edge spaces are shown in equations 2.16-2.21.
\[ \mathcal{E}_1(p) = (1 + r)(1 + r + s + t)T(p - 2) \]  
\[ \mathcal{E}_2(p) = (1 + r)(1 + s)T(p - 2) \]  
\[ \mathcal{E}_3(p) = (1 + s)(1 + r + s + t)T(p - 2) \]  
\[ \mathcal{E}_4(p) = (1 + t)(1 + s)T(p - 2) \]  
\[ \mathcal{E}_5(p) = (1 + t)(1 + r + s + t)T(p - 2) \]  
\[ \mathcal{E}_6(p) = (1 + t)(1 + r)T(p - 2) \]  
\[ \mathcal{F}_1(p) = (1 + r)(1 + s)(1 + r + s + t)T(p - 3) \]  
\[ \mathcal{F}_2(p) = (1 + r)(1 + t)(1 + r + s + t)T(p - 3) \]  
\[ \mathcal{F}_3(p) = (1 + r)(1 + s)(1 + t)T(p - 3) \]  
\[ \mathcal{F}_4(p) = (1 + s)(1 + t)(1 + r + s + t)T(p - 3) \]  

A face space is defined as being non-zero on the face itself and zero the other three faces. For example a face one function is non-zero on face one and zero on faces two, three, and four. The four face spaces are shown in equations 2.22-2.25.

The interior space is defined as being non-zero on the interior of the tetrahedral and zero on all faces. The interior space is shown in equation 2.26.
\[ \mathcal{T}(p) = (1 + r)(1 + s)(1 + t)(1 + r + s + t) \mathcal{T}(p - 4) \]  \hspace{1cm} (2.26)

The \( \mathcal{T}(p) \) space can be broken down into the smaller spaces as shown above. The modes in the basis can also be grouped into these spaces. The division of the basis into vertex, edge, face, and interior modes is important because on a FEM mesh, it allows us to form \( C^0 \) local functions. For example, for two elements sharing a face, the face modes can be combined into basis functions that are continuous and yet only span those two elements similar to the hat functions shown in figure 2.1

2.5 Basis

To represent the above spaces; it will be convenient to start with a basis developed by Dubiner [18]. This basis is a hierarchal modal polynomial basis. A hierarchal modal basis was chosen because it easily defines the order of each mode. As the order of the basis is increased new modes are added without changing the lower order basis. The basis will be broken into vertex, edge, face, and interior modes as shown below. The vertex modes, are given by

\[ \phi_{V1} = \left( \frac{1 + z}{2} \right) \] \hspace{1cm} (2.27)

\[ \phi_{V2} = \left( \frac{1 + y}{2} \right) \left( \frac{1 - z}{2} \right) \] \hspace{1cm} (2.28)

\[ \phi_{V3} = \left( \frac{1 - x}{2} \right) \left( \frac{1 - y}{2} \right) \left( \frac{1 - z}{2} \right) \] \hspace{1cm} (2.29)

\[ \phi_{V4} = \left( \frac{1 + x}{2} \right) \left( \frac{1 - y}{2} \right) \left( \frac{1 - z}{2} \right) \] \hspace{1cm} (2.30)
Vertex modes are planar in $r, s, t$ coordinates being one at the vertex and zero at the posing plane. Each is a member of its respective vertex space.

The edge modes ($0 < i < p$), are given by

\[ \phi_{E1} = \left( \frac{1 - x}{2} \right) \left( \frac{1 + x}{2} \right) P_{i-1}^{2,2}(x) \left( \frac{1 - y}{2} \right)^{i+1} \left( \frac{1 - z}{2} \right)^{i+1} \] (2.31)

\[ \phi_{E2} = \left( \frac{1 + x}{2} \right) \left( \frac{1 - y}{2} \right) \left( \frac{1 + y}{2} \right) P_{i-1}^{2,2}(y) \left( \frac{1 - z}{2} \right)^{i+1} \] (2.32)

\[ \phi_{E3} = \left( \frac{1 - x}{2} \right) \left( \frac{1 - y}{2} \right) \left( \frac{1 + y}{2} \right) P_{i-1}^{2,2}(y) \left( \frac{1 - z}{2} \right)^{i+1} \] (2.33)

\[ \phi_{E4} = \left( \frac{1 + y}{2} \right) \left( \frac{1 - z}{2} \right) \left( \frac{1 + z}{2} \right) P_{i-1}^{2,2}(z) \] (2.34)

\[ \phi_{E5} = \left( \frac{1 - x}{2} \right) \left( \frac{1 - y}{2} \right) \left( \frac{1 - z}{2} \right) \left( \frac{1 + z}{2} \right) P_{i-1}^{2,2}(z) \] (2.35)

\[ \phi_{E6} = \left( \frac{1 + x}{2} \right) \left( \frac{1 - y}{2} \right) \left( \frac{1 - z}{2} \right) \left( \frac{1 + z}{2} \right) P_{i-1}^{2,2}(z) \] (2.36)

These edge modes each belong to their associated edge space defined previously and have the properties of those spaces i.e. they are non-zero along the two adjacent faces and zero on the remaining faces. Each edge contains $p - 1$ modes.

The face modes ($0 < i < p - 1; 0 < j < p - i$), are given by

\[ \phi_{F1} = \left( \frac{1 - x}{2} \right) \left( \frac{1 + x}{2} \right) P_{i-1}^{2,2}(x) \left( \frac{1 - y}{2} \right)^{i+1} \left( \frac{1 + y}{2} \right)^{i+j+1} \] (2.37)

\[ \phi_{F2} = \left( \frac{1 - x}{2} \right) \left( \frac{1 + x}{2} \right) P_{i-1}^{2,2}(x) \left( \frac{1 - y}{2} \right)^{i+1} \left( \frac{1 + y}{2} \right)^{i+1} \left( \frac{1 + z}{2} \right) P_{j-1}^{2i+3,2}(z) \] (2.38)
\[
\phi_{F3} = \left( \frac{1 + x}{2} \right) \left( \frac{1 - y}{2} \right) \left( \frac{1 + y}{2} \right) P_{i-1}^{2,2}(y) \left( \frac{1 - z}{2} \right) \left( \frac{1 + z}{2} \right) P_{j-1}^{2i+3,2}(z) \tag{2.39}
\]

\[
\phi_{F4} = \left( \frac{1 - x}{2} \right) \left( \frac{1 - y}{2} \right) \left( \frac{1 + y}{2} \right) P_{i-1}^{2,2}(y) \left( \frac{1 - z}{2} \right) \left( \frac{1 + z}{2} \right) P_{j-1}^{2i+3,2}(z) \tag{2.40}
\]

The face modes each belong to their associated face space, they are non-zero on the face and zero on the remaining faces. Each face contains \((p-1)(p-2)/2\) modes.

Interior modes \((0 < i < p - 2; 0 < j < p - i - 1; 0 < k < p - i - j)\),

\[
\phi_I = \left( \frac{1 - x}{2} \right) \left( \frac{1 + x}{2} \right) P_{i-1}^{2,2}(x) \left( \frac{1 - y}{2} \right) \left( \frac{1 + y}{2} \right) P_{j-1}^{2i+3,2}(y) \left( \frac{1 - z}{2} \right) \left( \frac{1 + z}{2} \right) P_{k-1}^{2i+2j+4,2}(z) \tag{2.41}
\]

There are \((p-1)(p-2)(p-3)/6\) interior modes starting with a quartic mode. Interior modes are sometimes referred to as “bubble modes.” In total there are \((p+1)(p+2)(p+3)/6\) total modes as shown below.

\[
4 + 6(p - 1) + 4\frac{(p - 1)(p - 2)}{2} + \frac{(p - 1)(p - 2)(p - 3)}{6} = \frac{(p + 1)(p + 2)(p + 3)}{6} \tag{2.42}
\]

This set of basis functions spans \(T(p)\) defined in the previous section. To create the high order modes the Jacobi polynomial \(P_{m}^{\alpha,\beta}(x)\) will be used, the Jacobi polynomials are explicitly defined in Appendix A.

### 2.6 New Orthogonal Basis

The above is one basis for \(T(p)\), but there are an infinite number of possibilities. In this section, we look for a basis that allows \(C^0\) functions to be created, but also gives a mass matrix that can be locally inverted. To explain this process, we first need to define how the above basis can be re-arranged to create a new basis.
2.6.1 Mode Rearrangement

To show how modes from the Dubiner basis can be rearranged, inspection of the groups of spaces shows that some spaces are subsets of others. Shown below are the edge, face, and interior spaces included in each vertex space. Each vertex mode can be a linear contribution of edge, face, and interior modes contained in the vertex space.

\[ I(p), F_2(p), F_3(p), F_4(p), E_1(p), E_5(p), E_6(p) \subseteq V_1(p) \]  
(2.43)

\[ I(p), F_1(p), F_3(p), F_4(p), E_2(p), E_3(p), E_4(p) \subseteq V_2(p) \]  
(2.44)

\[ I(p), F_1(p), F_2(p), F_4(p), E_1(p), E_3(p), E_5(p) \subseteq V_3(p) \]  
(2.45)

\[ I(p), F_1(p), F_2(p), F_3(p), E_1(p), E_2(p), E_6(p) \subseteq V_4(p) \]  
(2.46)

Similarly the edge modes can be combined with modes within the face and interior spaces as shown below, and still be contained in its edge space.

\[ I(p), F_1(p), F_2(p) \subseteq E_1(p) \]  
(2.47)

\[ I(p), F_1(p), F_3(p) \subseteq E_2(p) \]  
(2.48)

\[ I(p), F_1(p), F_4(p) \subseteq E_3(p) \]  
(2.49)

\[ I(p), F_3(p), F_4(p) \subseteq E_4(p) \]  
(2.50)
Lastly, each mode in a face space can be rearranged using any mode within the interior space. This means that any interior mode could be combined with a face mode to form a new function which is still in the face mode space.

\[ \mathcal{I}(p), \mathcal{F}_2(p), \mathcal{F}_3(p) \subseteq \mathcal{E}_6(p) \] (2.52)

Using the subsets of spaces defined above, modes within a space can be rearranged to create new modes. This creates infinite possibilities for each mode and can used to create a new basis with properties similar to spectral elements.

### 2.6.2 Mass Matrix

The SEM uses an inexact integration rule which leads to a diagonal Mass matrix. For polynomial functions that are of the order \( p - 1 \) or less the mass matrix inversion of SEM gives exact results. To create an inversion process that can compete with SEM, it must be at least \( p - 1 \) accurate on tetrahedrals as well. To define \( p - 1 \) accuracy we first set up a system of equations that uses the mass matrix. To do this we solve for the coefficient vector \( \vec{u} \) to represent the function \( f \) using the basis \( \phi \) in

\[ \phi^T \vec{u} = f \] (2.54)

then multiplying by a test function and integrating yields

\[ \int_{\Omega} \phi \phi^T \, d\Omega \, \vec{u} = \int_{\Omega} \phi f \, d\Omega \] (2.55)

simplifying to,

\[ M \vec{u} = \int_{\Omega} \phi f \, d\Omega. \] (2.56)
To simplify this case, only one element will be looked at. The basis will be arranged in the following order: vertex modes, edge modes, face modes, and interior modes. The system to be solved is shown in equation 2.56 and simplified to be

\[ M \vec{u} = \vec{b} \]  

(2.57)

Where structure of the mass matrix \( M \) is shown below.

\[
M = \begin{bmatrix}
\int \phi_V \phi_V & \cdots & \int \phi_V \phi_E & \cdots & \int \phi_V \phi_F & \cdots & \int \phi_V \phi_I \\
\vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\
\int \phi_E \phi_V & \cdots & \int \phi_E \phi_E & \cdots & \int \phi_E \phi_F & \cdots & \int \phi_E \phi_I \\
\vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\
\int \phi_F \phi_V & \cdots & \int \phi_F \phi_E & \cdots & \int \phi_F \phi_F & \cdots & \int \phi_F \phi_I \\
\vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\
\int \phi_I \phi_V & \cdots & \int \phi_I \phi_E & \cdots & \int \phi_I \phi_F & \cdots & \int \phi_I \phi_I \\
\end{bmatrix}
\]

Using the mass matrix a set of coefficients \( \vec{u} \) can be solved for to multiply with the basis \( \phi \) to represent the function \( f \). If the function \( f \) is a polynomial of order \( p - 1 \) than it can be represented using only low order basis functions. A mode of order \( p - 1 \) or less will be defined as a low order mode and a mode with order \( p \) will be defined as a high order mode. Our goal is to develop a local mass matrix inversion process that is exact for functions of degree \( p - 1 \) on any given tetrahedral mesh.

### 2.6.3 Vertex Results

To show how the above accuracy constraint is used to help derive a new basis, we begin with the vertex modes. The following local mass matrix inversion process will be described for only vertex one as it is identical for the remaining vertices. To describe this process lets look at the first row of the mass matrix times the coefficient \( \vec{u} \) shown in equation 2.58.

\[
\int_{\Omega} \phi_{V1} \phi^T d\Omega \vec{u} = \int_{\Omega} \phi_{V1} f d\Omega 
\]  

(2.58)
Assuming the solution to \( \vec{u} \) is known it requires the full set of basis functions \( \phi \) to represent the function \( f \). If we are to be able to find \( u_{V_1} \) using local operations, it must be able to be found without knowledge of the other components of \( \vec{u} \). To de-couple vertex one we seek a new vertex basis function \( \phi_{V_\perp} \) that is orthogonal to all of the other low order modes \( \phi_{\text{low}} \) excluding the \( \phi_{V_1} \) mode. i.e. it will be orthogonal to the remaining vertices and all of the low-order edge, face, and interior spaces, and thus can be found independently of the \( \vec{u} \) coefficients of these modes, so that

\[
\int_{\Omega} \phi_{V_\perp} \phi_{\text{low}}^T \, d\Omega \, \vec{u} = 0. \tag{2.59}
\]

To do this, we search for a vertex basis function within the vertex space. The basis functions that are contained within the vertex space can be rearranged to create a vertex function that is orthogonal to all low order modes. This can be done by adding contributions from any of the modes on the three edges containing that vertex, the face modes on the three faces containing that vertex, and all of the interior modes to that vertex mode. The set of basis functions from the original basis that are contained in the vertex space will be defined as \( \phi_{\text{dof}} \) and all of the low order modes in the original basis will be defined as \( \phi_{\text{con}} \). Both \( \phi_{\text{dof}} \) and \( \phi_{\text{con}} \) exclude the vertex one basis function. This new vertex mode will be created using

\[
\phi_{V_\perp} = \phi_V + \phi_{\text{dof}}^T \vec{c}. \tag{2.60}
\]

Where \( \vec{c} \) yields the coefficients to create this new vertex function. It is found by solving the system of equations

\[
\int_{\Omega} \phi_{\text{con}} \phi_{\text{dof}}^T \, d\Omega \, \vec{c} = \int_{\Omega} \phi_{V_1} \phi_{\text{con}} \, d\Omega. \tag{2.61}
\]

For the vertex mode the degrees of freedom and the constraints are equal for all \( p \). To show this we equate the degrees of freedom (left hand side) vs the constraints (right hand side).

\[
3(p - 1) + 3 \frac{(p - 1)(p - 2)}{2} + \frac{(p - 1)(p - 2)(p - 3)}{6} = \frac{p(p + 1)(p + 2)}{6} - 1 \tag{2.62}
\]
For the degrees of freedom there are three $p - 1$ edge modes, three $(p - 1)(p - 2)/2$ face modes, and $(p - 1)(p - 2)(p - 3)/6$ interior modes. The constraints are all of the low order modes minus the vertex itself. This can be found by substituting in for $p = p - 1$ into equation 2.42.

When the new vertex mode is integrated with the hierarchical basis it has a diagonal term and all of the low order terms are equal to zero as shown in equation 2.59. To explain how this new mode is useful, let us examine the first component of equation 2.57. By modifying the vertex function it becomes

$$b_1 = \int_{\Omega} (\phi V_1 + \phi^T_{dof} \bar{c}) f \, d\Omega$$

(2.63)

By ignoring the high order terms in the mass matrix and assuming it to be diagonal the solution to $u_1$ is given by

$$u_1 = \frac{b_1}{M_{1,1}}$$

(2.64)

where $b_1$ is calculated in equation 2.63 and $M_{1,1}$ is given by.

$$M_{1,1} = \int_{\Omega} \phi V_1 (\phi V_1 + \phi^T_{dof} c) \, d\Omega$$

(2.65)

Now that the coefficient $u_1$ has been solved for, the remaining vertices can be solved in the same way. Thus, the first four vertex coefficients can be found. These will be exact if the forcing function is low order because in that case the neglected terms have no contribution.

Once the four vertex functions can be found, these can be contracted out of the equations for the edge, face, and interior modes by modifying $\vec{b}$. Shown below the procedure to modify $\vec{b}$ to give $\vec{\tilde{b}}$, where $tm$ is the total number of modes.

$$\begin{bmatrix} \tilde{b}_{E1} \\ \vdots \\ \tilde{b}_{tm} \end{bmatrix} = \begin{bmatrix} b_{E1} \\ \vdots \\ b_{tm} \end{bmatrix} - \begin{bmatrix} \int \phi_{E1}\phi V_1 & \cdots & \int \phi_{E1}\phi V_4 \\ \vdots & \ddots & \vdots \\ \int \phi_{tm}\phi V_1 & \cdots & \int \phi_{tm}\phi V_4 \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_4 \end{bmatrix}$$

The new vertex one function also has an analytical solution as shown below. It was found by guessing at the $\alpha$ coefficient in the Jacobi polynomial and by the similarities in...
between triangles and tetrahedrals. This new vertex function is only a function of $z$ and can be plotted as a one dimensional function as shown in figure 2.5. This new mode is now high order and is concentrated near the vertex where it equals one and equals zero at all other vertices.

$$\phi_{v1} = \left(1 + \frac{z}{2}\right) \frac{P_{3,1}^{3,1}(z)}{P_{p-1}^{3,1}(1)}$$ (2.66)

Figure 2.5: New vertex 1 function

2.6.4 Edge Results

The same approach used on the vertices will now be used on the edges. Edges will be found in order of $p$ starting with the quadratic mode. The new edge mode can be found using linear combinations of the modes contained in the edge space, the two adjacent face spaces, and the interior space. For example edge one can be combined with face one, face two, and interior modes. Again the edges are forced to be orthogonal to all of the low order modes except the vertices. The constraints minus the degrees of freedom are
\[ 5(p - 2) + 4 \frac{(p - 2)(p - 3)}{2} + \frac{(p - 2)(p - 3)(p - 4)}{6} \]
\[ - \left( p - 2 + 2 \frac{(p - 1)(p - 2)}{2} + \frac{(p - 1)(p - 2)(p - 3)}{6} \right) \]
\[ = \frac{p^2 - p - 2}{2} \]
equating to a net positive over-constraint for \( p > 2 \), this is shown graphically in figure 2.6. The top line shows constraints; there are \( p - 2 \) constraints on each of the five edges, \( (p - 2)(p - 3)/2 \) constraints on the four faces, and \( (p - 2)(p - 3)(p - 4)/6 \) interior constraints. The middle line shows that there are \( p - 2 \) degrees of freedom on the edge, \( (p - 1)(p - 2)/2 \) degrees of freedom on the two adjacent faces, and \( (p - 2)(p - 3)(p - 4)/6 \) interior degrees of freedom. Unfortunately the edges are over-constrained for all \( p \).

![Graph showing constraints vs degrees of freedom](image)

Figure 2.6: Constraints-degrees of freedom vs \( p \) for edge modes.

A non-square system of equations

\[
\int_{\Omega} \phi_{con} \phi_{dof}^T d\Omega c = \int_{\Omega} \phi_E \phi_{con} d\Omega
\]

(2.68)
can be created to solve for the edge mode. This system does not have a solution because
the rank is deficient. If the first edge mode cannot be solved for, then the remaining modes will be incorrect. This result comes as a surprise as this problem did not arise in the case of triangles. A possible way around this problem is given in the section “Vertex Ball.”

2.6.5 Face Results

Assuming the edge modes could be solved for and their contributions removed from the vector $b$, as was done for the vertex modes, the same process will be attempted to find the face modes. A new face mode is contained in the face space which contains all of the modes on that face and all of the interior modes. The constraints are opposing low order face and low order interior modes. The constraints minus the degrees of freedom

$$3\frac{(p-2)(p-3)}{2} + \frac{(p-2)(p-3)(p-4)}{6}$$

$$-\left(\frac{(p-1)(p-2)}{2} + \frac{(p-1)(p-2)(p-3)}{6} - 1\right)$$

$$= \frac{p^2 - 7p + 12}{2}$$

where first line gives the constraints which are the $(p-2)(p-3)/2$ low order modes on each of the three faces and the $(p-2)(p-3)(p-4)/6$ low order interior modes. The degrees of freedom are all of the $(p-1)(p-2)/2 - 1$ modes on the remaining on the face and all of the $(p-1)(p-2)(p-3)/6$ interior modes. Figure 2.7 graphically shows the constraints minus the degrees of freedom, it is over-constrained for $p > 4$ making it a non-square system like the edge modes. Although this did not work for the edge modes there happens to be a redundancy in the face modes. By either removing the redundant constraints or by using a least squares method the face modes can be solved for and an example of a $p = 4$ face mode is shown in figure 2.8 as a slice contour through the center of the tetrahedral.

Also to note, an attempt was made to make the face modes orthogonal to the edge modes in hopes that the face modes could be found before the edge modes. But the system
is over-constrained and cannot be solved for.

If a method can be found to obtain the edge modes correctly then the faces can be solved for. This mode will be orthogonal to all of the remaining low order modes. The mass matrix can be assumed diagonal and can be inverted for the correct face mode coefficient.

![Constraints-degrees of freedom vs p for face modes.](image)

**Figure 2.7:** Constraints-degrees of freedom vs p for face modes.

### 2.6.6 Interior Results

Once the vertex, edge, and face modes have been found their contributions can be removed from \( b \) and all that remains is to find the interior modes. The modes are then decoupled from the rest of the mesh and one can simply invert the interior-interior portion of the mass matrix. The interior modes shown in equation 2.41 are conveniently orthogonal to each other. Therefore it is trivial to invert this diagonal portion of the mass matrix.

### 2.6.7 Vertex Ball

In order to find the edge modes, in this section we propose an alternative method which we call the vertex ball method. The idea is to reduce the amount of orthonality required by solving for sets of edge modes simultaneously.
The set of edges that are to be solved simultaneously are "spokes" of a vertex ball. Figure 2.9 shows the idea of the vertex ball. This ball of tetrahedrals share a common vertex in the center. The edges to be solved simultaneously are shown in figure 2.10 as the red lines. After the shell of the vertex ball is decoupled as shown in figure 2.11 the spokes can be solved for.

The constraints should be all low-order face modes and interior modes plus the low order edges on the outer surface of the ball. Degrees of freedom for each edge are within the edge
space. This includes any other edge mode on that edge, all of the modes on the adjacent faces, and all of the interior modes. The constraints minus the degrees of freedom are

\[
3(p - 2) + 4\frac{(p - 2)(p - 3)}{2} + \frac{(p - 2)(p - 3)(p - 4)}{6} \\
- \left(p - 2 + 2\frac{(p - 1)(p - 2)}{2} + \frac{(p - 1)(p - 2)(p - 3)}{6}\right) \\
= \frac{p^2 - 5p + 6}{2}
\]  

(2.70)
for a single tetrahedral and is plotted in figure 2.12. It shows that for \( p = 3 \) there is equality but for \( p > 3 \) the system is overconstrained.

![Figure 2.12: Constraints-dofs vs p for vertex ball.](image)

To show how this works for \( p = 3 \) on a single tetrahedral a vertex ball will be made with vertex one being the center of the ball. The edges connected to vertex one are edges four, five, and six. Each edge can be recombined with modes within its edge space defined as \( \phi_{dof} \). This includes the other modes on that edge, the modes on the two adjacent faces, and the interior modes. For \( p = 3 \) this includes one high order edge mode and one mode on each of the two adjacent faces. Edges four, five, and six will be made orthogonal to the low order modes on the shell which includes edges one, two, and three and are defined as \( \phi_{con} \). For \( p = 3 \) there is only one high order face mode and no interior modes. Solve for the coefficients \( c \).
\[
\int_{\Omega} \phi_{con}^{T} \phi_{dof} \, d\Omega \, c = \int_{\Omega} \phi_{E4} \phi_{con} \, d\Omega
\] (2.71)

and use \(c\) to create \(b_{E4}\) shown below.

\[
b_{E4} = \int_{\Omega} (\phi_{E4} + \phi_{dof}^{T} \, c) \, f \, d\Omega
\] (2.72)

To solve for the coefficients \(u\) solve the system of equations shown below.

\[
\begin{bmatrix}
u_{E4} \\
u_{E5} \\
u_{E6}
\end{bmatrix} = \begin{bmatrix}
\int (\phi_{E4} + \phi_{dof}^{T} \, c) \phi_{E4} & \int (\phi_{E4} + \phi_{dof}^{T} \, c) \phi_{E5} & \int (\phi_{E4} + \phi_{dof}^{T} \, c) \phi_{E6} \\
\int (\phi_{E5} + \phi_{dof}^{T} \, c) \phi_{E4} & \int (\phi_{E5} + \phi_{dof}^{T} \, c) \phi_{E5} & \int (\phi_{E5} + \phi_{dof}^{T} \, c) \phi_{E6} \\
\int (\phi_{E6} + \phi_{dof}^{T} \, c) \phi_{E4} & \int (\phi_{E6} + \phi_{dof}^{T} \, c) \phi_{E5} & \int (\phi_{E6} + \phi_{dof}^{T} \, c) \phi_{E6}
\end{bmatrix}^{-1} \begin{bmatrix}
b_{E4} \\
b_{E5} \\
b_{E6}
\end{bmatrix}
\]

This can be repeated for the remaining vertices. Because each edge has two vertices there will be two solutions to each edge. The choice has been made to average these two solutions which both should be \(p - 1\) accurate approximations.

### 2.7 Accuracy of \(p - 1\) Mass Matrix Inversion

Now that a method has been developed to find an approximate local mass matrix inversion, the accuracy must be validated. To do this, the approximate mass matrix inversion process will be used to solve for the coefficient vector \(u\) to represent the function \(f\) using the basis \(\phi\) as shown below.

\[
\phi^{T}u = f
\] (2.73)

Multiplying by a test function and integrating gives

\[
\int_{\Omega} \phi \phi^{T} u \, d\Omega = \int_{\Omega} \phi f \, d\Omega
\] (2.74)

where the left hand side is the mass matrix times the coefficient vector \(u\). This yields a simple test case in which the mass matrix must be inverted. To test the accuracy of the approximate inversion scheme

\[
u = M_{\text{approx}}^{-1} \int_{\Omega} \phi f \, d\Omega
\] (2.75)
will be solved for $u$ where $M_{\text{approx}}^{-1}$ is the approximate inversion of the mass matrix and $f$ is the forcing function given by

$$f = \sin(\pi(x + 0.5)) \quad (2.76)$$

If the local mass matrix inversion process is $p-1$ accurate, the error in the approximation to $u$ should decrease as $h^p$ where $h$ is the spatial resolution of the tetrahedral mesh. The true mass matrix inversion would converge as $h^{p+1}$ [19].

The error measured is the $L_2$ error norm given by $L_2 = \|\phi^T u - f\|_2$. The geometry will be a simple cube of size $(-0.5 \leq x, y, z \leq 0.5)$ with an unstructured tetrahedral mesh. This test will be done on four meshes shown in figure 2.13 of different resolution for $p = 2$ and 3. Figure 2.14 shows the $L_2$ error vs mesh size for a $p = 2$ basis and figure 2.15 for the $p = 3$ basis. The prediction of order of accuracy has been met for all: for $p = 2$ the one step approximate inversion yields a power fit of 2.26 and the true mass matrix inversion equals 3.04. For $p = 3$ the approximate power fit is 3.13 and true is 4.50.

Figure 2.13: Mesh used in resolution studies
2.8 Iterative Performance of \( p - 1 \) Mass Matrix Inversion

Besides enabling explicit hp-FEM simulations, another use of the \( p - 1 \) mass matrix inversion is as an iterative algorithm for inverting the full mass matrix. By using the \( p - 1 \) inversion scheme iteratively, one can converge to the true full mass matrix inversion. This process is shown below.

\[
\Delta u = -M_{\text{approx}}^{-1} \int_\Omega \phi (\phi^T u - f) \, d\Omega
\]  

(2.77)

Where \( M_{\text{approx}}^{-1} \) is the approximate \( p - 1 \) mass matrix inversion and \( \Delta u \) is the change in the vector \( u \) which tends to zero as the scheme converges. To test the iterative convergence
the damping coefficient will be found for $p = 1, 2, \text{ and } 3$. The damping coefficient is the ratio of the residual of one step to the residual from the previous where the residual is defined as the norm of $\Delta u$. The forcing function is shown in equation 2.76 and the geometry in this case will be a simple cube of size $(-0.5 \leq x, y, z \leq 0.5)$ with a finely resolved unstructured tetrahedral mesh. Figure 2.16 shows the residual vs iteration for $p = 1, 2, 3$. The figure shows that there is rapid convergence in the beginning because the inversion is $p - 1$ accurate. The damping coefficients are taken from asymptotic section of the plot. The damping coefficients for $p = 1, 2, 3$ are 0.78, 0.79, and 0.88 respectively. As $p$ increases the damping coefficient increases. Meaning that for high order although the method is more accurate but it does not convergence as rapidly.

![Figure 2.16: Residual vs Iteration for $p = 1,2,3$](image)

### 2.9 Comparison to Fekete Points

As mentioned in the introduction, there are other methods for approximately inverting the mass matrix. In this section, we compare our approach to an approach that uses a nodal basis defined with Fekete points [12]. The Fekete points are the optimal nodes for a nodal basis because they have the smallest known Lebesque constant. Also, for quadrilaterals and hexahedrals, Fekete points are the GL points. The Fekete nodal basis for $p = 2$ includes all
of the vertex nodes and a node in the center of each edge. A Lagrangian polynomial basis is created using these ten nodes. For \( p = 3 \) there are the four vertex nodes and the two nodes on an edge. These are the GL nodes in 1D. There is also a node in the center of each face, for a total of 20 nodes.

A diagonal mass lumping inversion scheme is used to invert the mass matrix. Mass lumping is implemented by replacing the mass matrix with a diagonal matrix. The diagonal entries are formed by summing the terms in the entire row of the mass matrix. We have implemented and tested this scheme, and found that it is divergent when used as an iterative scheme. In order to be able to make a comparison, the iterative scheme was under-relaxed by a factor \( \omega \) which was chosen to make the scheme stable. Figure 2.17 shows the Fekete points and the \( p - 1 \) local mass matrix inversion. For the Fekete points there is good convergence in the beginning but the asymptotic convergence is so slow at the end that this method not practical. The damping coefficient for the asymptotic region is 0.99 for both \( p = 2, 3 \).

![Iteration vs Residual](image)

Figure 2.17: Residual vs Iteration for \( p = 2,3 \)
2.10 Conclusion

In this chapter the tetrahedral basis was defined and a method for locally inverting the Mass matrix was created. For $p = 3$ the edge modes were not obtained in the same fashion as in triangles because for tetrahedrals the edge modes are over-constrained. To remove constraints, edge modes were solved for by creating a ball of tetrahedrals sharing a common vertex. The edges inside this ball were made orthogonal to the shell of the ball and a small mass matrix is inverted to solve for edge modes. Although an inversion is needed it is not dependent on the size of the mesh, it only depends on the local connectivity of the mesh. This method is $p - 1$ accurate and converges more rapidly compared to other methods such as mass-lumping with a Fekete nodal basis. The vertex ball method has only been implemented for $p = 3$ and it is unclear how to implement for higher order.
Chapter 3

Unsteady Heat Equation

3.1 Introduction

In this chapter, we implement the unsteady heat equation to show how the mass matrix inversion techniques from the previous chapter can be used for unsteady simulations. This chapter contains the manipulations needed for a finite element formulation of the heat equation and the verification of the computational scheme in both time and space. The local mass matrix inversion process will be used in the solution of the unsteady heat equation.

3.2 Formulation

The unsteady heat equation, shown below, is used to study the transient conduction of heat through solids. The solution is important to many engineering problems, such as heat transfer in integrated circuits.

\[
\frac{\partial u}{\partial t} - k\nabla^2 u = f
\]  

(3.1)

Temperature is represented by the variable \( u \) and is a function of space and time, \( f \) is a forcing function, and \( k \) is a material constant.
3.2.1 Variational Form

The heat equation must undergo a few manipulations before it is in the form needed for a finite element formulation. Equation 3.2 shows the unsteady heat equation expanded into its 3D form.

\[
\frac{\partial u}{\partial t} - k \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right) = f(x, y, z, t) \tag{3.2}
\]

The heat equation is multiplied by a test function \(v\) and integrated over the domain.

\[
\int_{\Omega} v \left( \frac{\partial u}{\partial t} - k \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right) \right) d\Omega = \int_{\Omega} vf d\Omega \tag{3.3}
\]

Through integration by parts the left hand side becomes the symmetric bifunctional and the right hand side is the linear functional. Integration by parts also creates a term on the right hand side used in prescribing neumann boundary conditions defined by the flux term, \(q_n\), normal to the surface.

\[
\int_{\Omega} \left( v \frac{\partial u}{\partial t} + k \left( \frac{\partial v}{\partial x} \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \frac{\partial u}{\partial y} + \frac{\partial v}{\partial z} \frac{\partial u}{\partial z} \right) \right) d\Omega = \int_{\Omega} vf d\Omega - \oint_{\Gamma} q_n v d\Gamma \tag{3.4}
\]

3.2.2 Time Advancement

The unsteady heat equation requires a time advancement scheme to approximate the time derivative shown in equation 3.4. The term takes the form

\[
M \frac{\partial u}{\partial t} \tag{3.5}
\]

where \(M\) is the mass matrix. The time derivative is approximated using a four-stage diagonally implicit Runge-Kutta method (DIRK) [20]. This first stage is explicit and requires information from the previous time step. Because of this, the scheme is not self-starting. Therefore for the first time-step we use a 3-stage 3rd-order accurate scheme [21]. The other three stages require the solution of an implicit problem similar in form to a backwards-Euler discretization.
To solve the implicit problem, a “dual-time-stepping” approach is used. The dual time stepping approach involves adding pseudo-time derivative terms to the governing equations and then iterating until a steady state is reached in pseudo-time. The approximate mass matrix inversion is used to make the explicit advancements in pseudo time. Steady state in pseudo-time corresponds to convergence to the implicit time step update. The pseudo time step $\Delta t^*$ is set by

$$\frac{1}{\Delta t^*} = \frac{1}{\Delta t} + \frac{k}{h^2}$$  \hfill (3.6)

where $k$ is the material constant, $\Delta t$ is the physical time step, and $h$ is the mesh size of each element as by

$$h = \frac{4V_{tet}}{(p+1)^2A_{max}}$$  \hfill (3.7)

The mesh size $h$ is a function of $V_{tet}$ the volume of the tetrahedral, $p$ the polynomial degree, and $A_{max}$ is area of the largest face on the element.

### 3.2.3 $p$-Multigrid

To accelerate convergence in pseudo-time $p$-multigrid will be used in conjunction with the above described iterative process. A relaxation scheme works by smoothing out high frequency error modes. As the high frequency error modes become smoother, relaxation methods converge slower. Multigrid is a method to combine with a relaxation scheme to accelerate convergence.

There are three main types of multigrid: Algebraic multigrid, geometric multigrid, and $p$-multigrid. Geometric multigrid works by switching between solutions on different meshes of varying resolution. Algebraic multigrid uses subsets of the same mesh to achieve different levels of resolution. A low frequency error mode on a fine grid is a high frequency error mode on a course grid, making a relaxation scheme more effective on the course grid [22]. This same idea works with $p$-multigrid but instead of varying mesh size the order of the scheme changes [7]. A high order basis will have low frequency error modes but when
switched to a low order basis the low frequency error modes become high frequency error modes making the relaxation scheme more effective. Switching between modes of different polynomial degree is simple using the hierarchal basis chosen for this implementation.

$p$-multigrid used in a V-cycle will be tested in this paper. A V-cycle goes from high order, down to low order linear modes (restriction), and back up again to a high order basis (prolongation). In the case of this paper a V-cycle for $p = 3$ would go in this order: $p = 3-2-1-2-3$ and for $p = 2$ it will be: $p = 2 - 1 - 2$, this is repeated until the system has converged on the high order basis. At the lowest level where $p = 1$ a direct solve is used.

### 3.3 Verification

In this section, we begin by examining the temporal accuracy and spatial accuracy of the implementation. We then examine the iterative performance of the new dual-time-stepping algorithm.

#### 3.3.1 Temporal Accuracy

To test the temporal accuracy of the scheme; the method of known solutions will be used. We seek a solution $u$ that is quadratic in space and decays exponentially in time

$$u = \frac{1}{\lambda} e^{-\lambda t} \left( 25 - x^2 \right)$$

where $\lambda = 5$. Substituting this into equation 3.2 and solving for the forcing function $f$ gives

$$f = -e^{-\lambda t} \left( 25 - x^2 - \frac{2k}{\lambda} \right)$$

where $k = 1$. The analytic solution is second order in space and so a basis with $p = 2$ will be used. This will ensure that any numerical errors are from the DIRK scheme and not the spatial error from the basis because a second order basis can perfectly represent any second order polynomial. An unstructured mesh is used on a cube with dimensions $(-0.5 \leq x, y, z \leq 0.5)$. For $x = -0.5, 0.5$ and for all $y, z$ Dirichlet boundary conditions are
used with \( u = 0 \). For the remaining boundaries Neuman boundary conditions are used with 
\( \frac{\partial u}{\partial n} = 0 \). Four cases were tested with \( \Delta t^{-1} \) equal to 10, 20, 40, and 80 for a total of 10, 20, 40, and 80 timesteps respectively. The results are shown in figure 3.1. An exponential curve was used to fit the \( L_2 \) error data and an order of accuracy of 2.819 was observed. As predicted, the order of accuracy of the DIRK scheme is about third order.

![Figure 3.1: \( L_2 \) Error vs Time Step](image)

### 3.3.2 Spatial Accuracy

To test the spatial accuracy of the scheme, steady state solutions are calculated. Polynomial order of 1, 2, and 3 were used in conjunction with four meshes of different resolution as shown previously in figure 2.13. The geometry is a cube with dimensions \((-0.5 \leq x, y, z \leq 0.5)\). The method of known solutions is used where the solution \( u \) is given below

\[
 u = \cos(\pi x)\cos(\pi y)\cos(\pi z)
\]  

(3.10)

Substituting this into equation 3.2 and solving for the forcing function \( f \)

\[
 f = 3\pi^2k\cos(\pi x)\cos(\pi y)\cos(\pi z)
\]  

(3.11)

where \( k = 1 \). Dirichlet boundary conditions on all faces of the cube are set using the analytical solution.
Figure 3.2 shows the $L_2$ error norms for each level of $p$ and mesh length $h$. As the mesh size decreases the error for each level of $p$ decreases. Also, as $p$ increases the error for each level of mesh decreases. This shows that a high order basis with a course mesh has an equivalent accuracy to a lower order basis with a fine mesh. A power fit for $p = 1, 2, 3$ is 1.933, 2.938, and 4.082 respectively, it matches the predicted fit of $h^{p+1}$ [19].

![Figure 3.2: $L_2$ Error vs Mesh Size for $p = 1, 2, 3$](image)

### 3.3.3 Iterative Convergence

To test the iterative convergence a series of cases will be run varying mesh size, time step, and polynomial degree. The geometry is a cube with dimensions $(-0.5 \leq x, y, z \leq 0.5)$. The forcing function is shown in equation 3.9 and initial condition shown in equation 3.8, with Dirichlet boundary conditions on all sides of the cube. When a solution is iterating it initially converges rapidly and then approaches an asymptotic convergence. This asymptotic convergence is measured by the damping coefficient $\mu$, defined as the ratio of the residual from of one step to the previous step. Another way to look at the damping coefficient is number of cycles required to reduce the error one order of magnitude, $\log(0.1)/\log(\mu)$.

The first test will be on a fine mesh and will show cycles/decade for time steps of $\Delta t^{-1} = 0, 40, 400$ and $p = 1, 2, 3$. As shown in figure 3.3 as the time step is increased the
cycles/decade increases and as $p$ is increased the cycles/decade also increases.

Figure 3.3: cycle/decade vs $\Delta t^{-1}$ for $p = 1, 2, 3$ on fine mesh

The next test will be for a fixed time step of $\Delta t^{-1} = 40$ and compares cycles/decade versus mesh size for $p = 1, 2, 3$. As shown in figure 3.4 as the mesh is refined the cycles/decade increase and as $p$ increases the cycles/decade increase as well.

The previous two figures show that as a mesh becomes more resolved, or the polynomial degree is increased, the convergence rate decreases. To overcome this $p$-multigrid will be used. The first case is shown in figure 3.5 for $p = 2$ and in figure 3.6 for $p = 3$ both with the same fine mesh and for $\Delta t^{-1} = 0, 40, 400$. The first iteration is after one v-cycle has been completed which is why the residual is initially reduced drastically. With $p$-multigrid it takes far less total iterations to converge than without. As the time step is increased it also only takes a few more iterations to converge. For $p = 2$ to reach a residual norm of $10^{-12}$ it takes 4 iterations with $\Delta t^{-1} = 400$, 5 iterations for $\Delta t^{-1} = 40$, and 7 iterations $\Delta t^{-1} = 0$. For $p = 3$ it takes 5 iterations with $\Delta t^{-1} = 400$, 10 iterations for $\Delta t^{-1} = 40$, and 13 iterations $\Delta t^{-1} = 0$. This shows that convergence is weakly dependent on the polynomial degree.
For a fixed $\Delta t^{-1} = 400$ and $p = 2$ three meshes of different resolution will be compared and are shown in figure 3.7 and figure 3.8 for $p = 3$. The convergence with $p$-multigrid is independent of the mesh size and it takes about one more iteration when going from $p = 2$ to 3.

### 3.4 Conclusion

In this chapter the local mass matrix inversion process was used in the solution of the unsteady heat equation. The time derivative was approximated using a third order accurate DIRK scheme and to solve this implicit problem a dual time stepping approach was taken. The spatial accuracy was verified in a mesh resolution study where the error has the power fit of $h^{p+1}$. The iterative convergence without $p$-multigrid is very dependent on the mesh, time step, and polynomial degree. For this method to be feasible when applied to real problems it needs to converge rapidly for large meshes, and for any $p$ and time step. To accelerate the convergence $p$-multigrid was used. It was shown that $p$-multigrid is mesh independent and weakly dependent on time step and polynomial degree.
Figure 3.5: $p = 2$ $p$-MG Residual vs Iteration for $\Delta t^{-1} = 0, 40, 400$ on fine mesh

Figure 3.6: $p = 3$ $p$-MG Residual vs Iteration for $\Delta t^{-1} = 0, 40, 400$ on fine mesh

Figure 3.7: $p = 2$ $p$-MG Residual vs Iteration for three meshes at $\Delta t^{-1} = 400$
Figure 3.8: $p = 3$ $p$-MG Residual vs Iteration for three meshes at $\Delta t^{-1} = 400$
In this work, a high order finite element method on tetrahedrals was implemented. This method uses a local mass matrix inversion similar to what has been developed on triangles. The local inversion is not done on a single tetrahedral but instead it uses a small cluster of tetrahedrals with a common vertex. This ball of tetrahedrals creates a small coupled mass matrix that requires direct inversion. This inversion is not dependent on mesh size, only on local mesh connectivity. The local mass matrix inversion is $p - 1$ accurate and when used iteratively converges very rapidly. So far only $p = 3$ has been implemented and it is fourth order accurate. This method converges more rapidly than a nodal basis created by the Fekete points and a mass lumping scheme.

The local mass matrix inversion was applied to the unsteady heat equation. The time derivative term was approximated using a third order diagonally implicit Runge-Kutta scheme and dual time stepping. This scheme was proven to be third order accurate in time and fourth order accurate in space. To accelerate convergence $p$-Multigrid was implemented and showed a major improvement in the convergence and reduced the number of iterations.
Bibliography


Chapter 5

Appendix A

5.1 Jacobi Polynomial

The Jacobi Polynomial is described in equation 5.1 where the integer \( n \) is the order of the polynomial, \( \alpha \) and \( \beta \) are inputs giving different orthogonality properties shown in equation 5.2, and \( \Gamma \) is the Gamma function.

\[
P_n^{(\alpha,\beta)}(x) = \frac{\Gamma(\alpha + n + 1)}{n!\Gamma(\alpha + \beta + n + 1)} \sum_{m=0}^{n} \frac{\Gamma(n + 1)}{\Gamma(m + 1)\Gamma(n - m + 1)} \frac{\Gamma(\alpha + \beta + m + n + 1)}{\Gamma(\alpha + m + 1)} \left(\frac{x - 1}{2}\right)^m
\]

(5.1)

Jacobi Polynomials have the the orthogonality property shown in equation 5.2

\[
\int_{-1}^{1} (1 - x)^\alpha (1 + x)^\beta P_m^{(\alpha,\beta)}(x) P_n^{(\alpha,\beta)}(x) dx = \frac{2^{\alpha + \beta + 1}}{2n + \alpha + \beta + 1} \frac{\Gamma(n + \alpha + 1)\Gamma(n + \beta + 1)}{\Gamma(n + \alpha + \beta + 1)n!} \delta_{mn}
\]

(5.2)