An Explicit Description of Implementation of 4D, H(div)-conforming Simplicial Finite Elements in MFEM

Patrick Saber

Advised by David M. Williams, Associate Professor
The Pennsylvania State University
Department of Mechanical Engineering

Overview

Overview of the Presentation

I. Motivation
II. Mathematical Preliminaries
III. Implementation Details
IV. Order of Accuracy Results (Grad-div problem)

I. Motivation

Motivation

4D Formulation of Inhomogeneous Maxwell's Equations

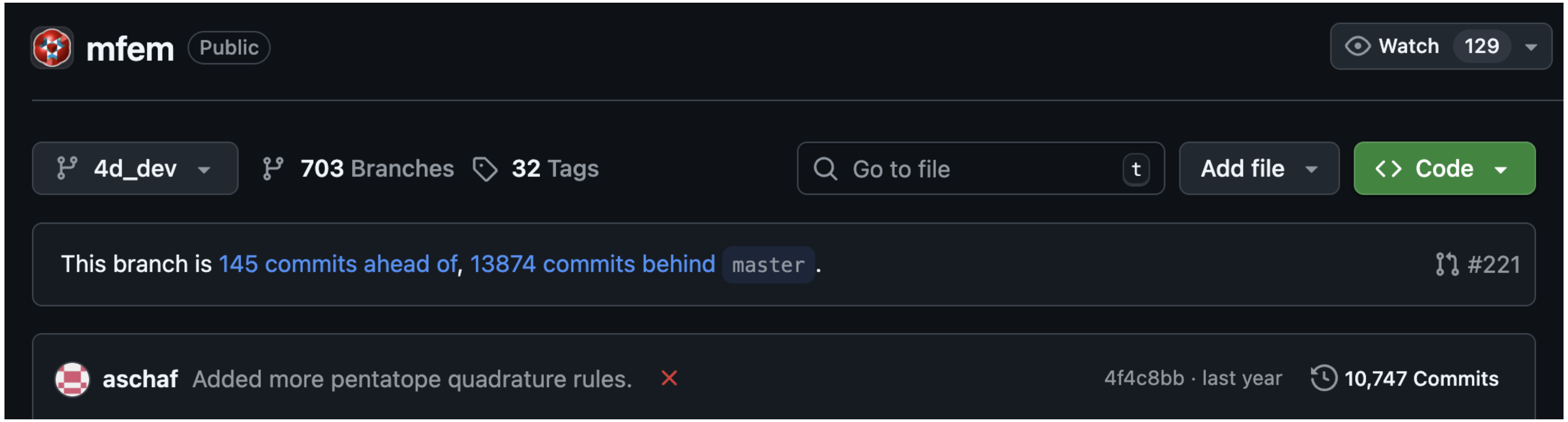
$$F = \frac{1}{2}\begin{bmatrix} 0 & -cB_x & -cB_y & -cB_z \\ cB_x & 0 & -E_z & E_y \\ cB_y & E_z & 0 & -E_x \\ cB_z & -E_y & E_x & 0 \end{bmatrix}, \quad H = \frac{1}{2}\begin{bmatrix} 0 & -cE_x & -cE_y & -cE_z \\ cE_x & 0 & B_z & -B_y \\ cE_y & -B_z & 0 & B_x \\ cE_z & B_y & -B_x & 0 \end{bmatrix}, \quad G = -\begin{bmatrix} \rho \\ j_x \\ j_y \\ j_z \end{bmatrix}$$

- Final equations

$$\boxed{\operatorname{curl}(F) = 4\pi G, \qquad \operatorname{curl}(H) = 0, \qquad \operatorname{div}(G) = 0}$$

Motivation

- **Objective:** high-order, conforming finite elements for Maxwell's equations
- **Problem:** existing 4D branch of MFEM primarily contains low-order finite elements



II. Mathematical Preliminaries

Vector Spaces and Derivative Operators in 4D

4D Derivative Operators

$$\mathrm{grad},\ \mathrm{skwGrad},\ \mathrm{curl},\ \mathrm{div}.$$

Infinite-Dimensional Sobolev Spaces

$$H\left(\mathrm{grad},\Omega,\mathbb{R}\right)=\left\{u\in L^{2}\left(\Omega,\mathbb{R}\right):\mathrm{grad}\,u\in L^{2}\left(\Omega,\mathbb{R}^{4}\right)\right\},$$
$$H\left(\mathrm{skwGrad},\Omega,\mathbb{R}^{4}\right)=\left\{E\in L^{2}\left(\Omega,\mathbb{R}^{4}\right):\mathrm{skwGrad}\,E\in L^{2}\left(\Omega,\mathbb{K}\right)\right\},$$
$$H\left(\mathrm{curl},\Omega,\mathbb{K}\right)=\left\{F\in L^{2}\left(\Omega,\mathbb{K}\right):\mathrm{curl}\,F\in L^{2}\left(\Omega,\mathbb{R}^{4}\right)\right\},$$
$$H\left(\mathrm{div},\Omega,\mathbb{R}^{4}\right)=\left\{G\in L^{2}\left(\Omega,\mathbb{R}^{4}\right):\mathrm{div}\,G\in L^{2}\left(\Omega,\mathbb{R}\right)\right\},$$

Finite Element Spaces in 4D

Finite-Dimensional Subspaces

$$V_k\Lambda^0(\mathfrak{T}^4) := P^k(\mathfrak{T}^4),$$

H(grad)-conforming

$$V_k\Lambda^1(\mathfrak{T}^4) := (P^{k-1}(\mathfrak{T}^4))^4 \oplus \left\{p \in (\tilde{P}^k(\mathfrak{T}^4))^4 | p \cdot x = 0\right\},$$

H(skwGrad)-conforming

$$V_k\Lambda^2(\mathfrak{T}^4) := \mathcal{L}\left((P^{k-1}(\mathfrak{T}^4))^6\right) \oplus \left\{B \in \mathcal{L}((\tilde{P}^k(\mathfrak{T}^4))^6) | Bx = 0\right\}.$$

H(curl)-conforming

$$V_k\Lambda^3(\mathfrak{T}^4) := (P^{k-1}(\mathfrak{T}^4))^4 \oplus \tilde{P}^{k-1}(\mathfrak{T}^4)x,$$

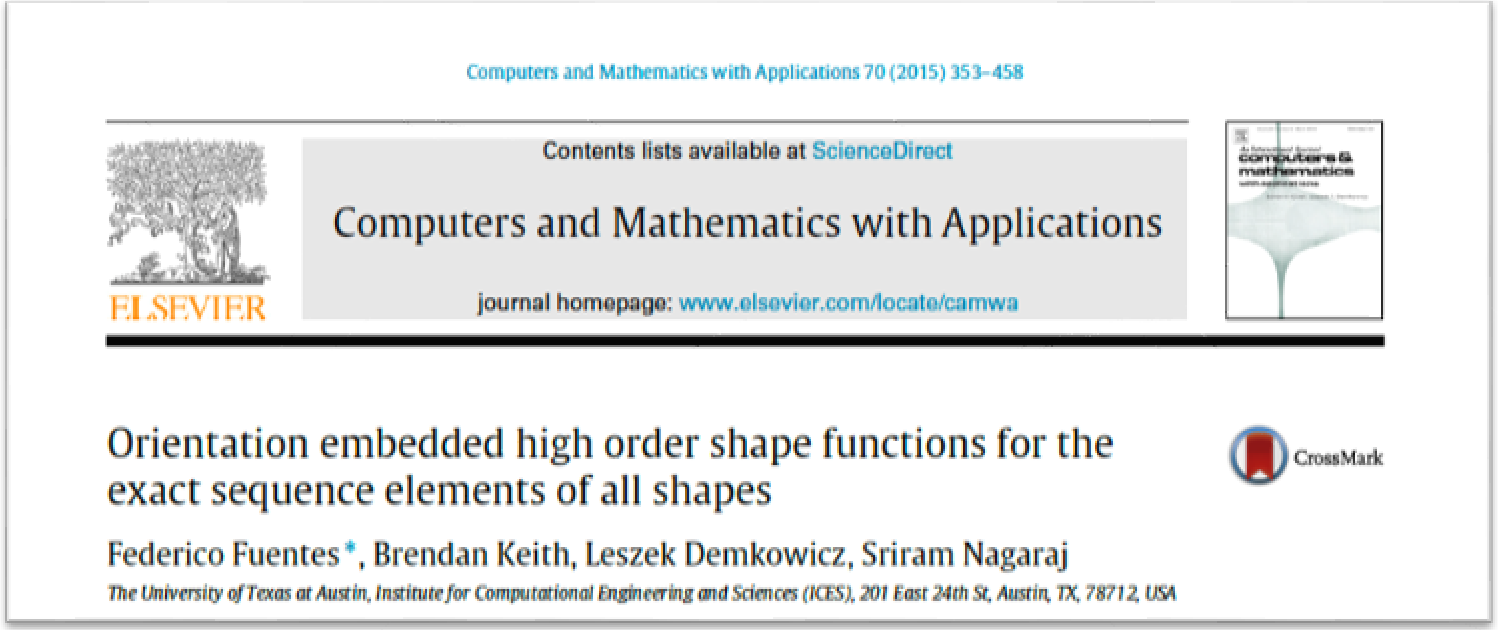
H(div)-conforming

$$V_k\Lambda^4(\mathfrak{T}^4) := P^{k-1}(\mathfrak{T}^4).$$

L2-conforming

Inspiration

- Our 4D approach is inspired by the approach of Fuentes et al. in 3D
- F. Fuentes, B. Keith, L. Demkowicz, and S. Nagaraj, “Orientation embedded high order shape functions for the exact sequence elements of all shapes,” *Computers and Mathematics with Applications*, (2015)


Computers and Mathematics with Applications 70 (2015) 353–458

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa

ELSEVIER

Orientation embedded high order shape functions for the exact sequence elements of all shapes

CrossMark

Federico Fuentes*, Brendan Keith, Leszek Demkowicz, Sriram Nagaraj

The University of Texas at Austin, Institute for Computational Engineering and Sciences (ICES), 201 East 24th St, Austin, TX, 78712, USA

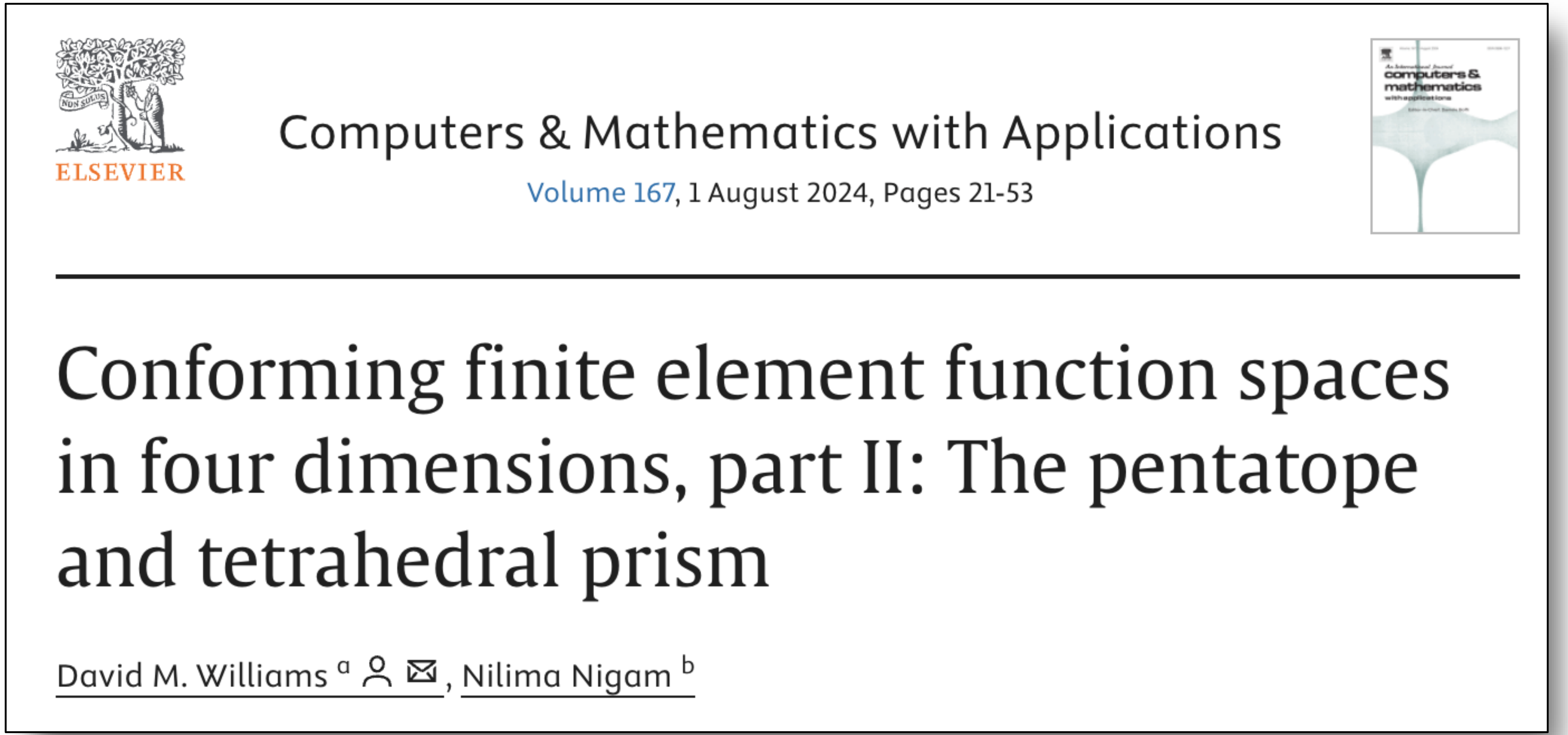
PennState
College of Engineering

Conforming Shape Functions in 3D

- Advantages of the Fuentes et al. approach
  - Intuitive construction based on barycentric coordinates
  - Easily generalizable to higher dimensions
  - Flexibility for representing the basis functions of multiple derivative operators with relatively small modifications to the formulation
  - Hierarchical structure
  - Previously implemented in MFEM for square pyramids in 3D
- Extended to 4D simplices by Nigam and Williams

Conforming Shape Functions in 4D

- D. Williams, N. Nigam, “Conforming Finite Element Function Spaces in Four Dimensions, Part II: The Pentatope and Tetrahedral Prism,” *Computers and Mathematics with Applications*, (2024)



ELSEVIER

Computers & Mathematics with Applications

Volume 167, 1 August 2024, Pages 21-53

Conforming finite element function spaces in four dimensions, part II: The pentatope and tetrahedral prism

David M. Williams [a], Nilima Nigam [b]

H(div)-Conforming Shape Functions

- Legendre and Jacobi Polynomial Construction

Bubble Functions

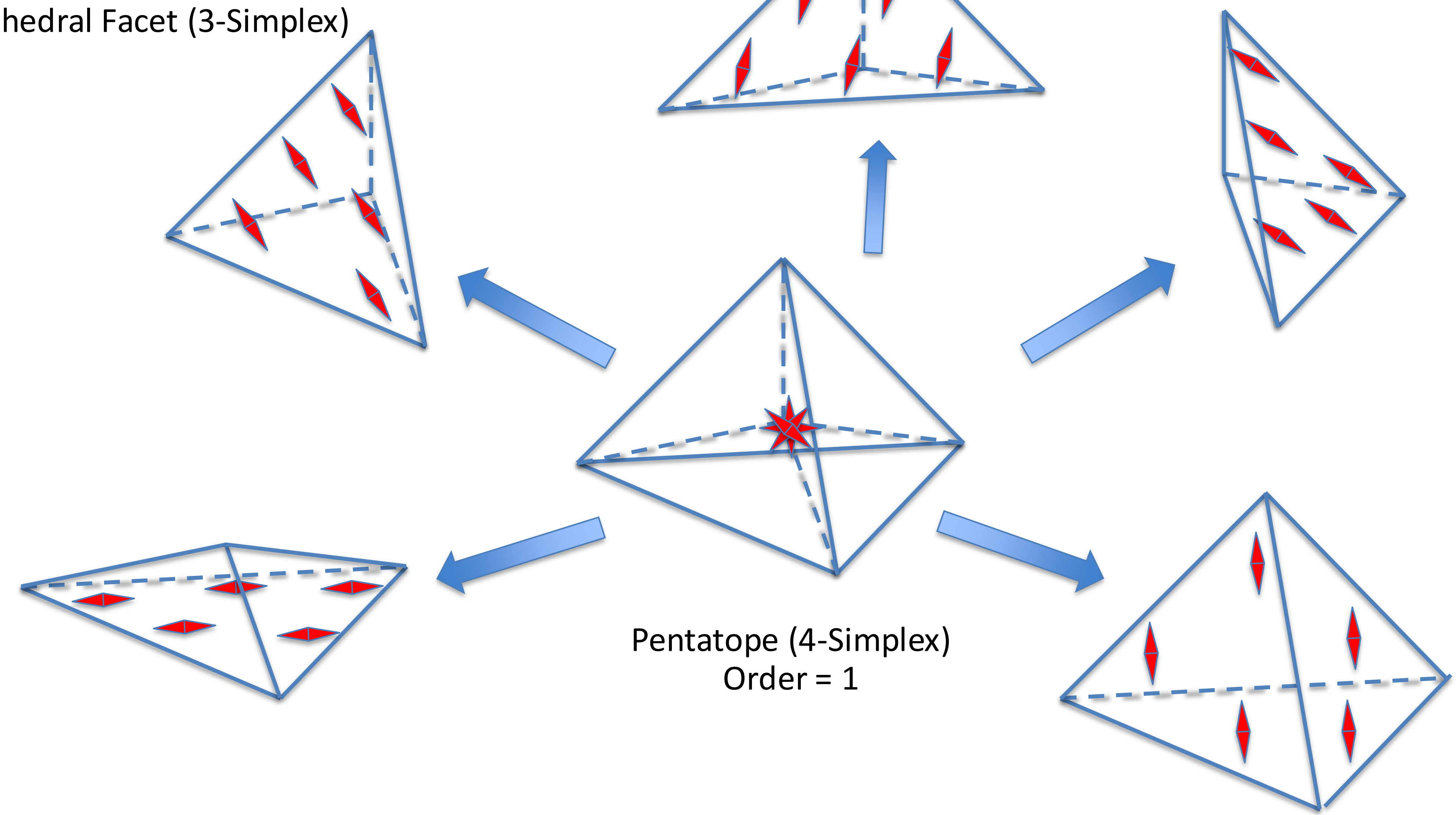
$$\psi^{r}_{ij\ell m}\left(\vec{\lambda}_{abcde}(x)\right) =$$
$$P_i\left(\lambda_b;\lambda_a+\lambda_b\right)P_j^{2i+1}\left(\lambda_c;\lambda_a+\lambda_b+\lambda_c\right)P_\ell^{2(i+j+1)}\left(\lambda_d;\lambda_a+\lambda_b+\lambda_c+\lambda_d\right)$$
$$\cdot L_m^{2(i+j+\ell)+3}\left(\lambda_e\right)\Big[\lambda_a\left(\nabla\lambda_b\times\nabla\lambda_c\times\nabla\lambda_d\right)-\lambda_b\left(\nabla\lambda_c\times\nabla\lambda_d\times\nabla\lambda_a\right)$$
$$+\lambda_c\left(\nabla\lambda_d\times\nabla\lambda_a\times\nabla\lambda_b\right)-\lambda_d\left(\nabla\lambda_a\times\nabla\lambda_b\times\nabla\lambda_c\right)\Big],$$

Facet Functions

$$\psi^{\mathcal{F}}_{ij\ell}\left(\vec{\lambda}_{abcd}(x)\right) =$$
$$P_i\left(\lambda_b;\lambda_a+\lambda_b\right)P_j^{2i+1}\left(\lambda_c;\lambda_a+\lambda_b+\lambda_c\right)P_\ell^{2(i+j+1)}\left(\lambda_d;\lambda_a+\lambda_b+\lambda_c+\lambda_d\right)$$
$$\cdot\Big[\lambda_a\left(\nabla\lambda_b\times\nabla\lambda_c\times\nabla\lambda_d\right)-\lambda_b\left(\nabla\lambda_c\times\nabla\lambda_d\times\nabla\lambda_a\right)$$
$$+\lambda_c\left(\nabla\lambda_d\times\nabla\lambda_a\times\nabla\lambda_b\right)-\lambda_d\left(\nabla\lambda_a\times\nabla\lambda_b\times\nabla\lambda_c\right)\Big],$$

III. Implementation Details

Degrees of Freedom



Constructing the Vandermonde Matrix

Construct j[th] vector basis function with k = 0 to 4 components, ψ_j^k evaluated at points $\boldsymbol{r}_i$

Contract each function with the appropriate normal vector n_k

This operation returns a generalized Vandermonde matrix $\mathcal{V}_{ij}$

$$\widehat{\mathcal{V}}_{ijk} = \psi_j^k(\boldsymbol{r}_i),$$
$$\mathcal{V}_{ij} = \widehat{\mathcal{V}}_{ijk} n_k.$$

Orientation Matching Across Elements

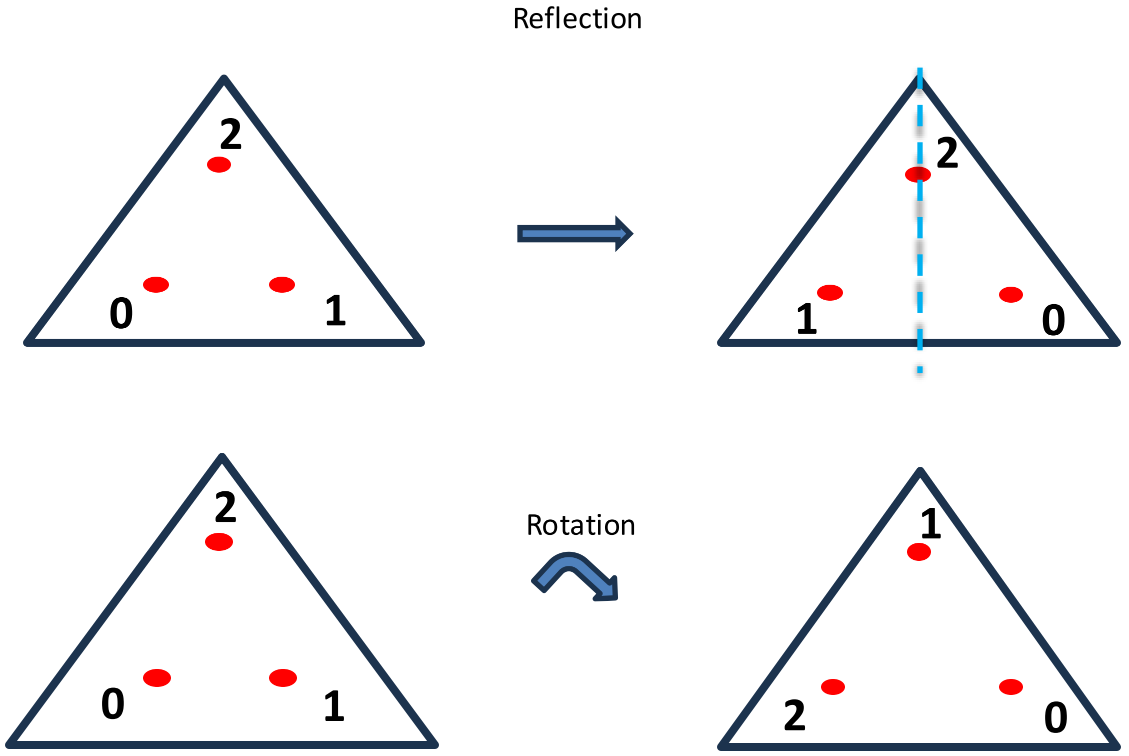
- In 3D each tetrahedral has 4 triangular faces that need to match the orientation of a neighboring tetrahedral face
- Triangles have 6 unique orientations produced by reflections or rotations of the base geometry
- In 4D each Pentatope has 5 tetrahedral facets that need to match the orientation of a neighboring Pentatopic facet.
- Tetrahedra have 24 unique orientations produced by reflections or rotations of the base geometry

Orientation Matching Across Elements

Transformations of Degrees of Freedom

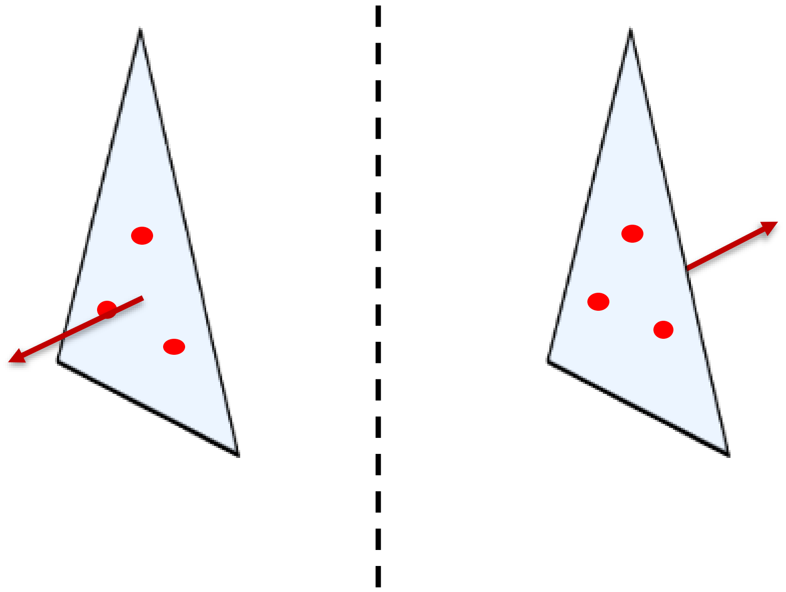
Identity Configuration

Orientation



Normal Vector Sign Flip

Reflection



Orientation Matching Across Elements

```
for (int j = 0; j <= p; j++)
{
   for (int i = 0; i + j <= p; i++)
   {
      int o = TriDof - ((pp2 - j)*(pp1 - j))/2 + i;
      int k = p - j - i;
      TriDofOrd[0][o] = o;  // (0,1,2)
      TriDofOrd[1][o] = -1-(TriDof-((pp2-j)*(pp1-j))/2+k);  // (1,0,2)
      TriDofOrd[2][o] =     TriDof-((pp2-i)*(pp1-i))/2+k;   // (2,0,1)
      TriDofOrd[3][o] = -1-(TriDof-((pp2-k)*(pp1-k))/2+i);  // (2,1,0)
      TriDofOrd[4][o] =     TriDof-((pp2-k)*(pp1-k))/2+j;   // (1,2,0)
      TriDofOrd[5][o] = -1-(TriDof-((pp2-i)*(pp1-i))/2+j);  // (0,2,1)
      if (!signs)
      {
         for (int kk = 1; kk < 6; kk += 2)
         {
            TriDofOrd[kk][o] = -1 - TriDofOrd[kk][o];
         }
      }
   }
}
```

Degree of Freedom Indexes		
0	1	2
1	0	2
2	0	1
2	1	0
1	2	0
0	2	1

Degree of Freedom Indexes		
0	1	2
-2	-1	-3
2	0	1
-3	-2	-1
1	2	0
-1	-3	-2

Orientation Matching Across Elements


```
for (int k=0; k<=p; k++)
{
   for (int j=0; j+k<=p; j++)
   {
      for (int i=0; i+j+k<=p; i++)
      {
          int o = TetDof + TriDof2 - ((pp3-k)*(pp2-k)*(pp1-k))/6 - (pp2-j)*
                  (pp1-j)/2 - k*j + i;
          int l = p-k-j-i;
          TetDofOrd[0][o] = o;
          TetDofOrd[1][o] = -1 - (TetDof + TriDof2 - ((pp3-j)*(pp2-j)*(pp1-j))/6 -
                            (pp2-k)*(pp1-k)/2 - j*k + i);
          TetDofOrd[2][o] =       TetDof + TriDof2 - ((pp3-i)*(pp2-i)*(pp1-i))/6 -
                            (pp2-k)*(pp1-k)/2 - i*k + j;
          TetDofOrd[3][o] =  -1 - (TetDof + TriDof2 - ((pp3-k)*(pp2-k)*(pp1-k))/6 -
                            (pp2-i)*(pp1-i)/2 - k*i + j);
          TetDofOrd[4][o] =        TetDof + TriDof2 - ((pp3-j)*(pp2-j)*(pp1-j))/6 -
                            (pp2-i)*(pp1-i)/2 - j*i + k;
          TetDofOrd[5][o] = -1 - (TetDof + TriDof2 - ((pp3-i)*(pp2-i)*(pp1-i))/6 -
                            (pp2-j)*(pp1-j)/2 - i*j + k);
          TetDofOrd[6][o] =        TetDof + TriDof2 - ((pp3-k)*(pp2-k)*(pp1-k))/6 -
                            (pp2-l)*(pp1-l)/2 - k*l + j;
          TetDofOrd[7][o] = -1 - (TetDof + TriDof2 - ((pp3-l)*(pp2-l)*(pp1-l))/6 -
                            (pp2-k)*(pp1-k)/2 - l*k + j);
          TetDofOrd[8][o] =       TetDof + TriDof2 - ((pp3-l)*(pp2-l)*(pp1-l))/6 -
                            (pp2-j)*(pp1-j)/2 - l*j + k;
          TetDofOrd[9][o] =  -1 - (TetDof + TriDof2 - ((pp3-j)*(pp2-j)*(pp1-j))/6 -
                            (pp2-l)*(pp1-l)/2 - j*l + k);
          TetDofOrd[10][o] =       TetDof + TriDof2 - ((pp3-j)*(pp2-j)*(pp1-j))/6 -
                            (pp2-k)*(pp1-k)/2 - j*k + l;
          TetDofOrd[11][o] =  -1 - (TetDof + TriDof2 - ((pp3-k)*(pp2-k)*(pp1-k))/6 -
                           (pp2-j)*(pp1-j)/2 - k*j + l);
          TetDofOrd[12][o] =        TetDof + TriDof2 - ((pp3-i)*(pp2-i)*(pp1-i))/6 -
                            (pp2-l)*(pp1-l)/2 - i*l + k;
          TetDofOrd[13][o] =  -1 - (TetDof + TriDof2 - ((pp3-l)*(pp2-l)*(pp1-l))/6 -
                            (pp2-i)*(pp1-i)/2 - l*i + k);
          TetDofOrd[14][o] =        TetDof + TriDof2 - ((pp3-k)*(pp2-k)*(pp1-k))/6 -
                           (pp2-i)*(pp1-i)/2 - k*i + l;
          TetDofOrd[15][o] = -1 - (TetDof + TriDof2 - ((pp3-i)*(pp2-i)*(pp1-i))/6 -
                            (pp2-k)*(pp1-k)/2 - i*k + l);
          TetDofOrd[16][o] =       TetDof + TriDof2 - ((pp3-l)*(pp2-l)*(pp1-l))/6 -
                            (pp2-k)*(pp1-k)/2 - l*k + i;
          TetDofOrd[17][o] =  -1 - (TetDof + TriDof2 - ((pp3-k)*(pp2-k)*(pp1-k))/6 -
                           (pp2-l)*(pp1-l)/2 - k*l + i);
          TetDofOrd[18][o] =       TetDof + TriDof2 - ((pp3-i)*(pp2-i)*(pp1-i))/6 -
                            (pp2-j)*(pp1-j)/2 - i*j + l;
          TetDofOrd[19][o] = -1 - (TetDof + TriDof2 - ((pp3-j)*(pp2-j)*(pp1-j))/6 -
                            (pp2-i)*(pp1-i)/2 - j*i + l);
          TetDofOrd[20][o] =       TetDof + TriDof2 - ((pp3-j)*(pp2-j)*(pp1-j))/6 -
                            (pp2-l)*(pp1-l)/2 - j*l + i;
          TetDofOrd[21][o] = -1 - (TetDof + TriDof2 - ((pp3-l)*(pp2-l)*(pp1-l))/6 -
                            (pp2-j)*(pp1-j)/2 - l*j + i);
          TetDofOrd[22][o] =       TetDof + TriDof2 - ((pp3-l)*(pp2-l)*(pp1-l))/6 -
                            (pp2-i)*(pp1-i)/2 - l*i + j;
          TetDofOrd[23][o] = -1 - (TetDof + TriDof2 - ((pp3-i)*(pp2-i)*(pp1-i))/6 -
                            (pp2-l)*(pp1-l)/2 - i*l + j);
```

Degree of Freedom Indexes			
0	1	2	3
0	1	3	2
0	3	1	2
0	3	2	1
2	0	1	3
3	0	1	2

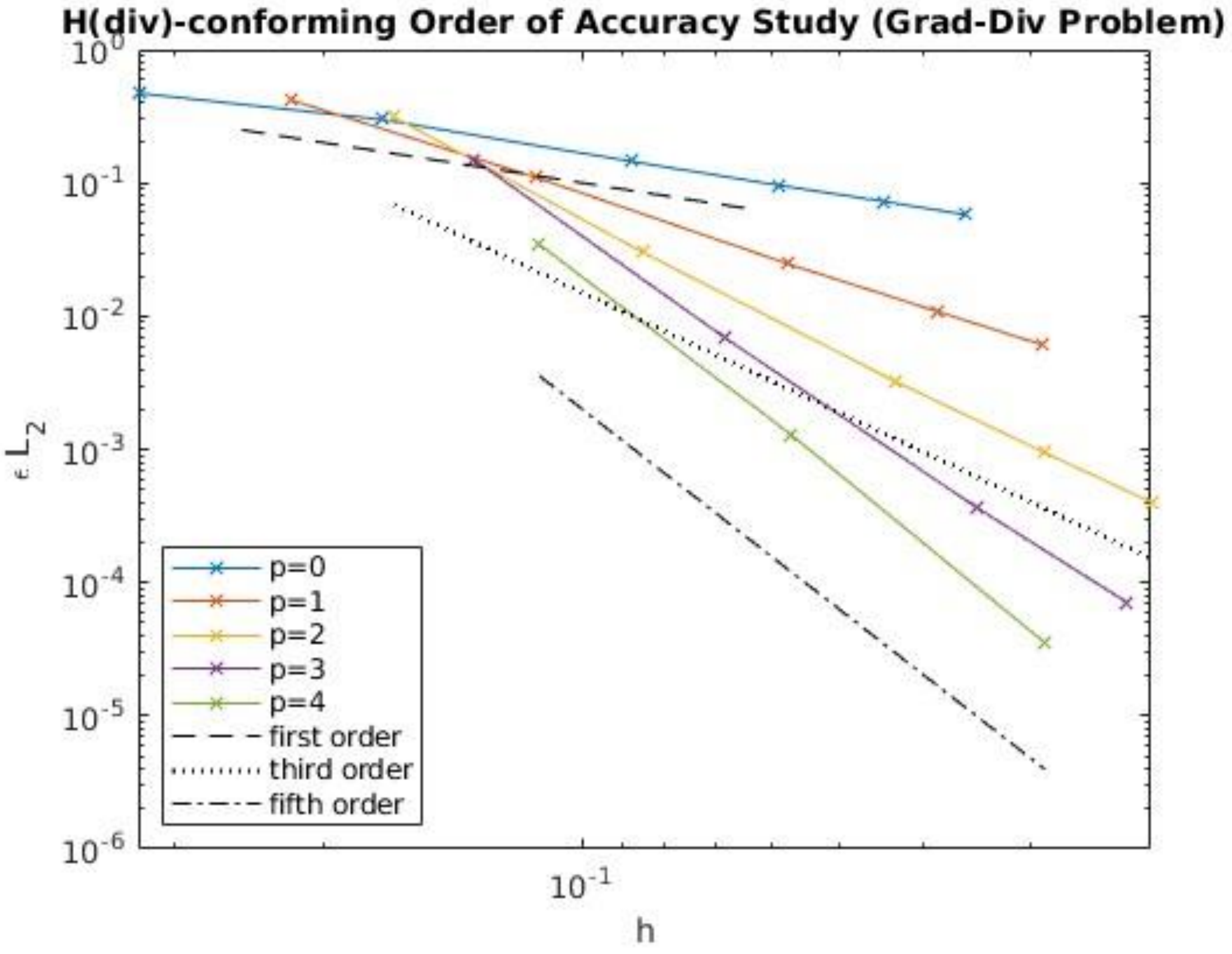
Degree of Freedom Indexes			
0	1	2	3
-1	-2	-4	-3
0	3	1	2
-1	-3	-2	-4
0	2	3	1
-1	-4	-3	-2

** Tables not complete

Order of Accuracy

$$-\nabla(\nabla \cdot \vec{F}) + \vec{F} = \vec{f}$$

$$\vec{F} = \begin{bmatrix} cos(\kappa x)sin(\kappa y)sin(\kappa z)sin(\kappa t) \\ cos(\kappa y)sin(\kappa z)sin(\kappa t)sin(\kappa x) \\ cos(\kappa z)sin(\kappa t)sin(\kappa x)sin(\kappa y) \\ cos(\kappa t)sin(\kappa x)sin(\kappa y)sin(\kappa z) \end{bmatrix}$$



Future Work

- Implement high-order H(skwGrad)-conforming finite elements on the 4-simplex
- Implement high-order H(curl)-conforming finite elements on the 4-simplex

**Theoretical details can be found on MFEM's seminar page in Dr. David Williams seminar talk on: *Finite Element Exterior Calculus in Four-Dimensional Space*