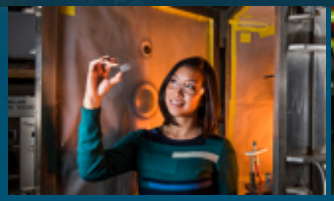
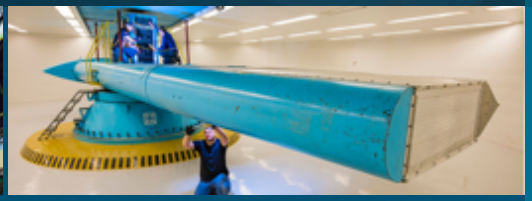
Stanford ME469: Common low-Mach Discretization Approaches





PRESENTED BY

Stefan P. Domino

Computational Thermal and Fluid Mechanics

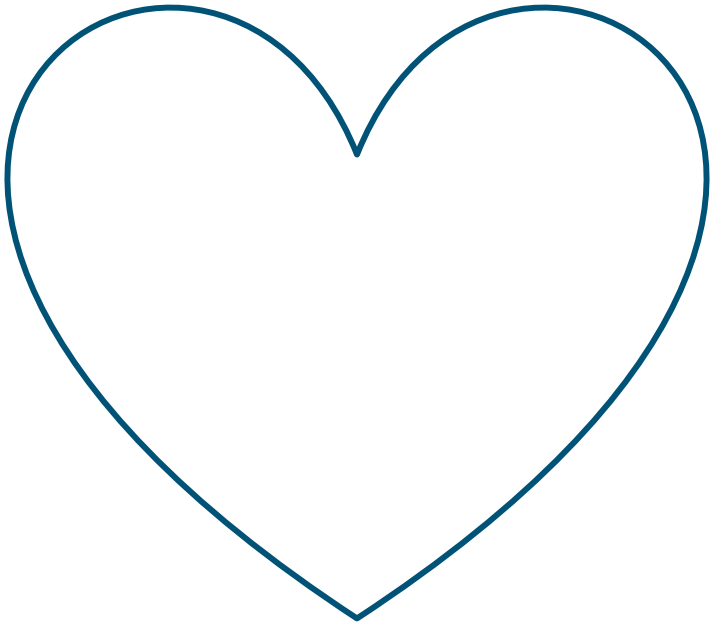
Sandia National Laboratories SAND2018-4536 PE

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Common low-Mach Discretization Approaches: Outline

- The Concept of Meshing
- Why Unstructured?
- Unstructured Element Typoes
- Model PDE for Discretization
- Finite Element Method (FEM)
- Control-Volume Finite Element Method (CVFEM)
- Edge-based Vertex-Centered (EBVC)
- Cell-centered Finite Volume (FV)
- Staggered arrangement
- Conclusions

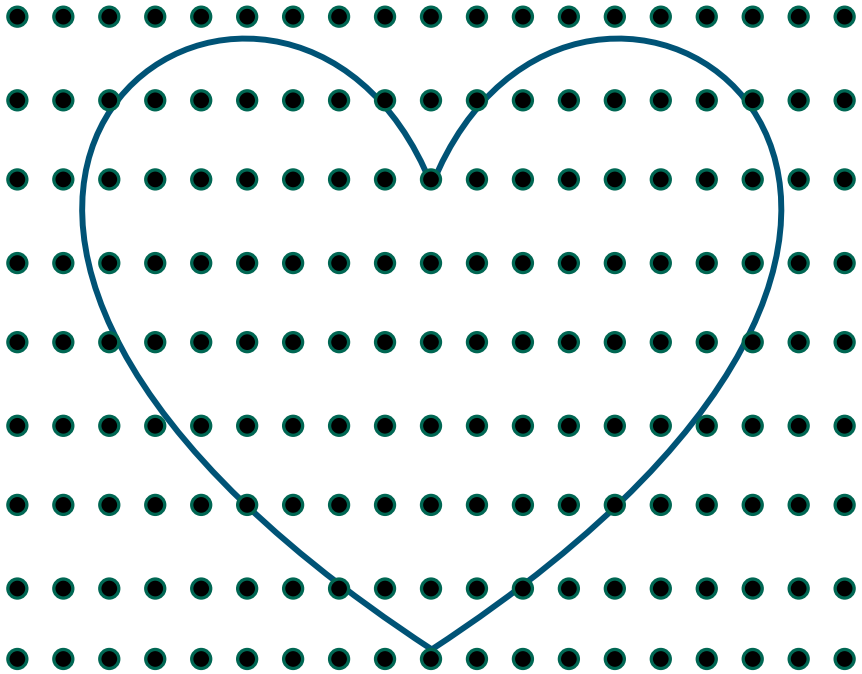
Introducing a Mesh over Heart Domain, Ω



- Notes: Geometry is:

1. Complex
2. Curved
3. Sharp

Introducing a [Finite Difference] Mesh over Heart Domain, Ω

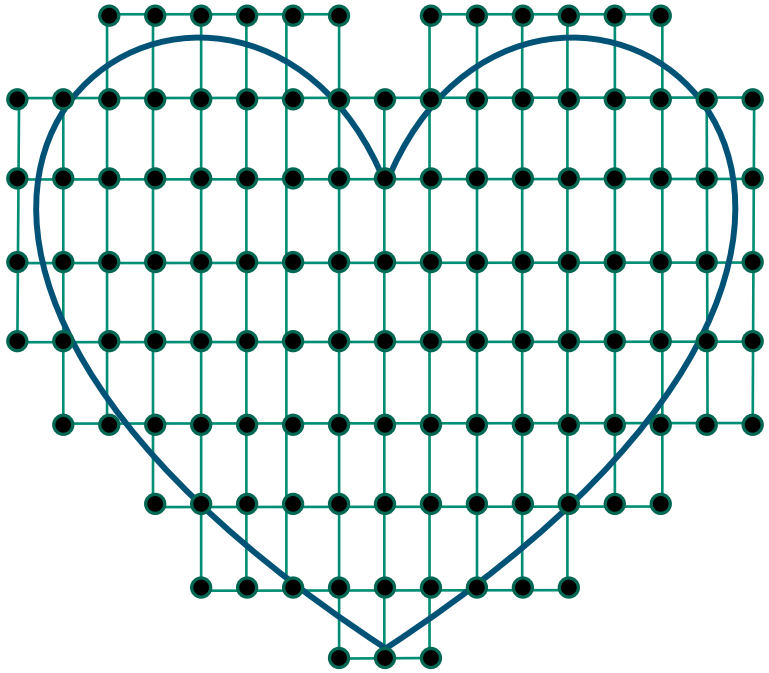


? What breaks..

- Notes: Geometry is:

1. Complex
2. Curved
3. Sharp

Introducing a [Structured Mesh] over Heart Domain, Ω;

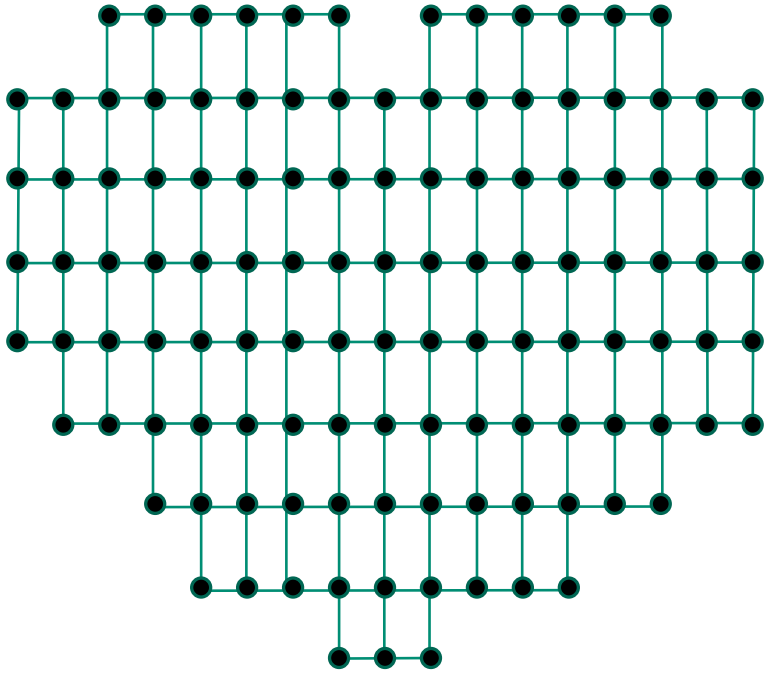


? What breaks..

- Notes: Geometry is:

1. Complex
2. Curved
3. Sharp

Introducing a [Structured Mesh] over Heart Domain, Ω;



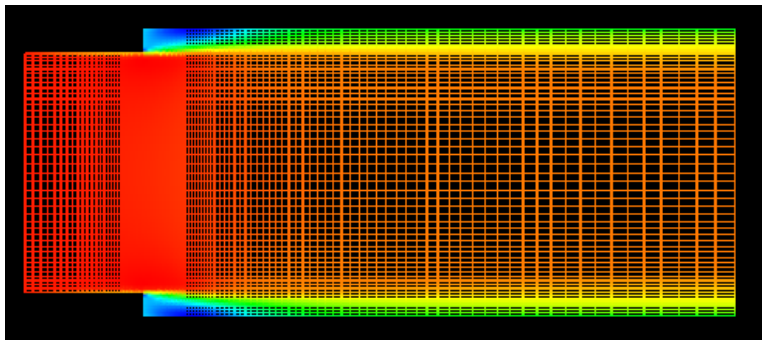
? What breaks..

- Notes: Geometry is:

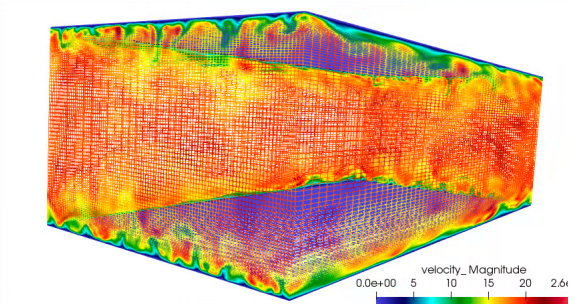
1. Complex
2. Curved
3. Sharp

Structured vs Unstructured

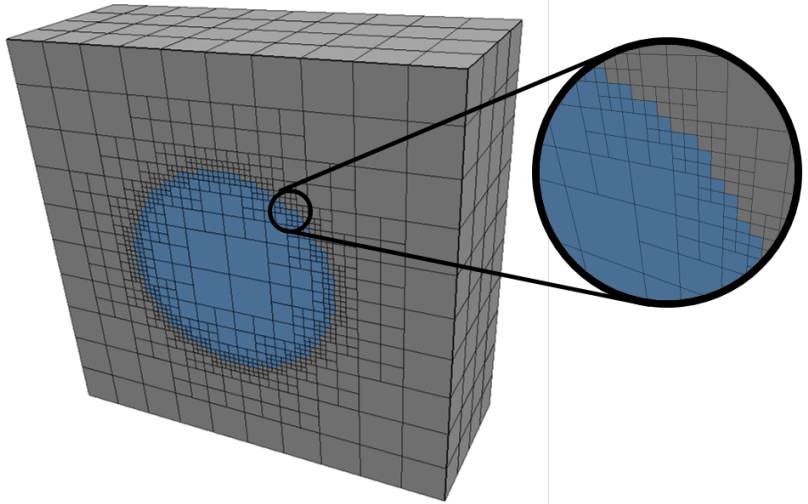
- Many times, the canonical flows of interest are in simplified geometries that allow for cartesian meshes – with "stair-stepping"

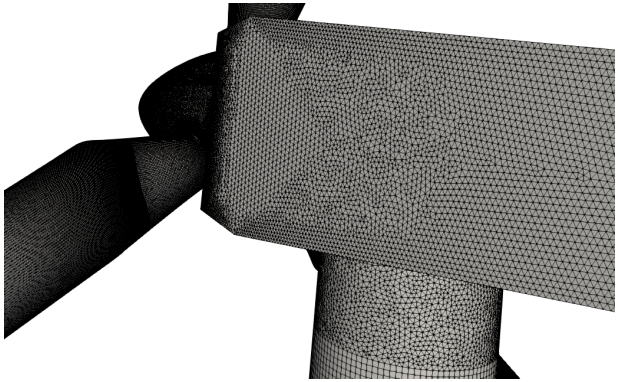


RANS-based backward facing step (Domino, 2012)



Re_τ 395 plane-channel (Jofre, Domino, Iaccarino, 2018)



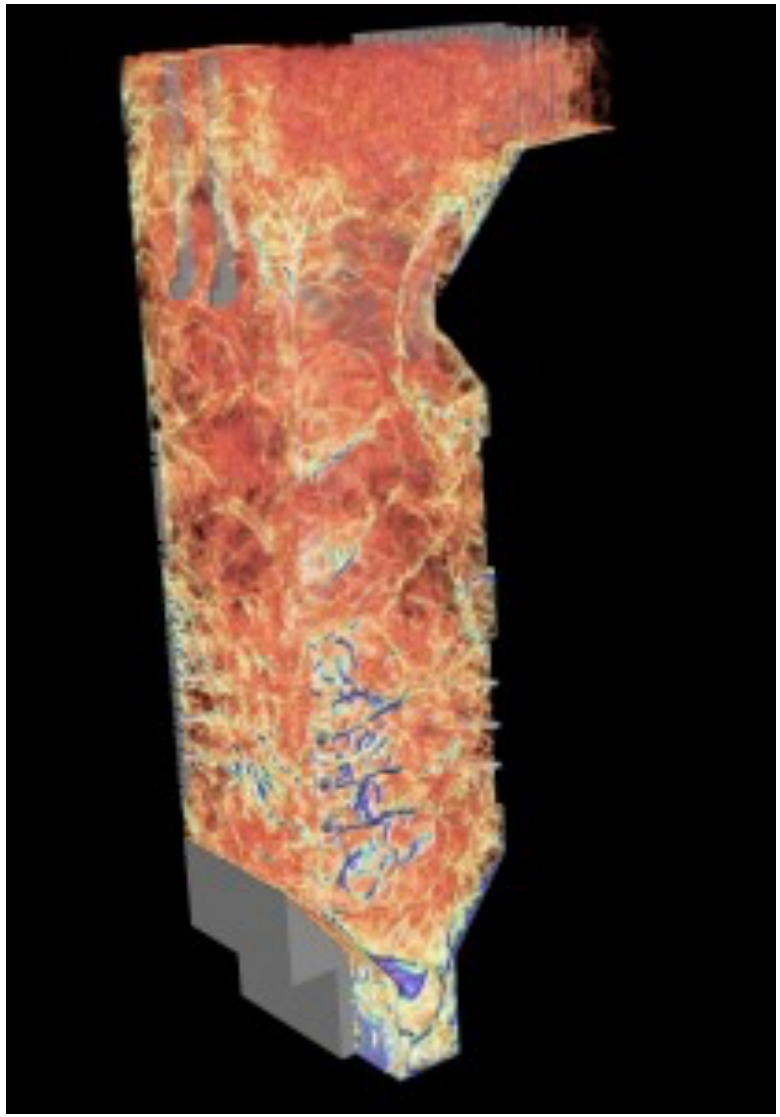


Often times, not!

https://www.itascainternational.com/software/introduction-to-meshing

In Fairness….

The Carbon-Capture Multidisciplinary Simulation Center



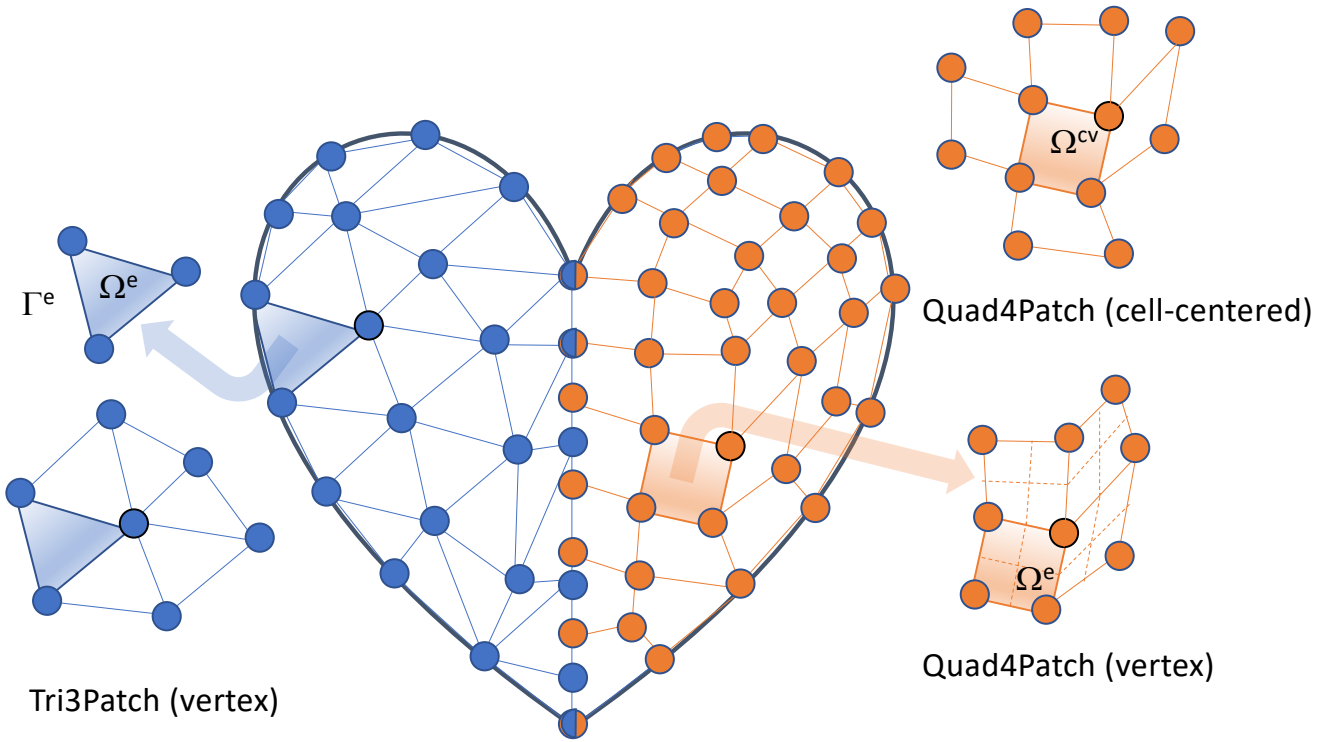
15MW coal-fired boiler volume rendered image of large (90 μm) particles

Staggered schemes have been demonstrated to support complex applications

Cut-cells and embedded approaches help

http://ccmsc.utah.edu/about.html

Introducing a Mesh over Heart Domain, Ω



- Elements of size 4 (Quad4) or 3 (Tri3) have been introduced
- Exterior domain is faceted
- Non-conformal interface between the Tri3 and Quad4 block
- Two types of connectivities have been presented: node:element and element:face:element
- Ω^e vs Ω^{cv}

Integration Over the Domain

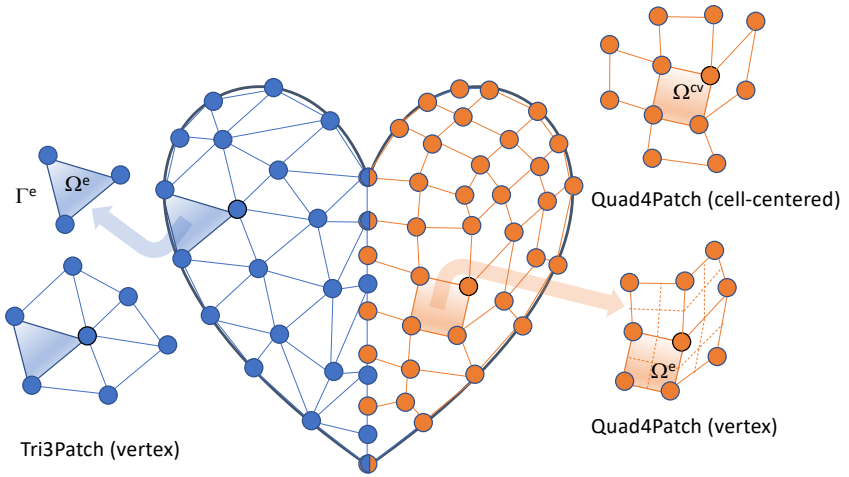
- Consider a simple model equation with the heart domain in mind:

$$\frac{\partial F_j}{\partial x_j} = S,$$

Where F_j is a flux and S is a source term

- Integrating over the domain, Ω:

$$\int_\Omega \frac{\partial F_j}{\partial x_j} dV = \int_\Omega S dV.$$



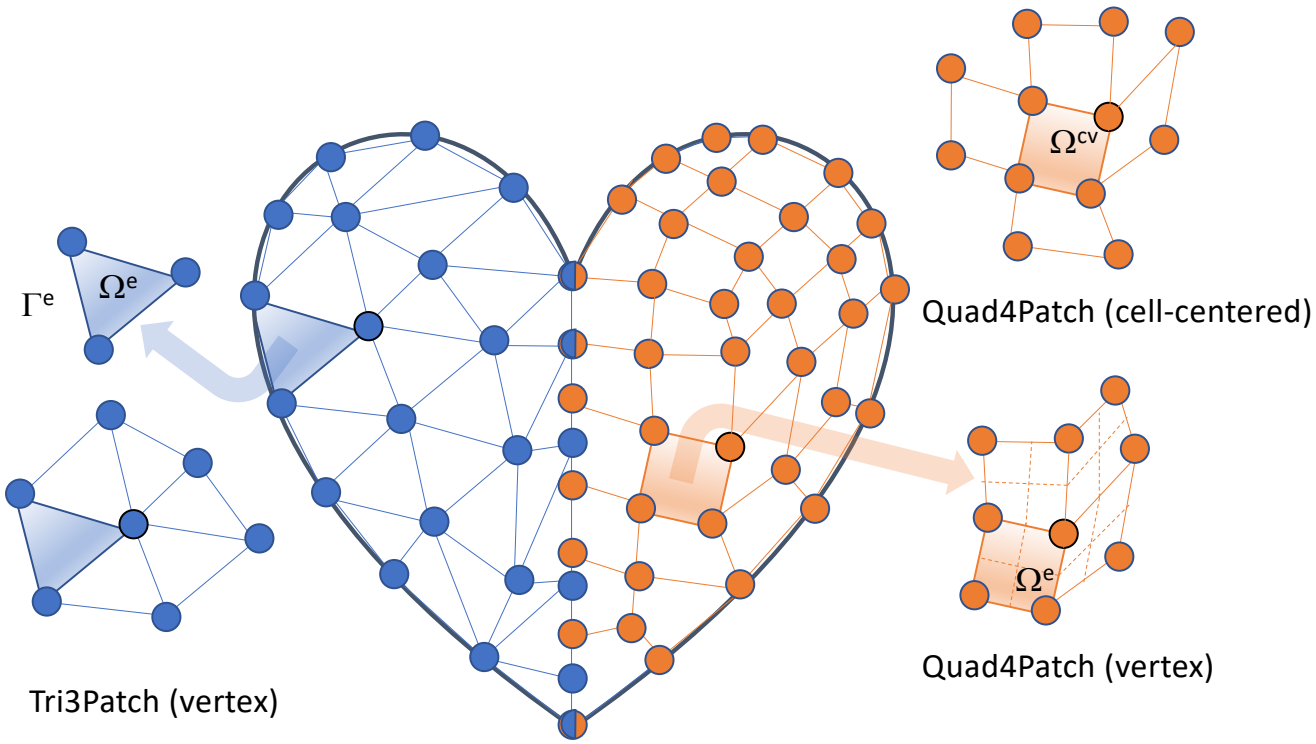
- Without loss of generality, let us define a set of subdomains, Ω_k:

$$\sum_k \int_{\Omega^k} \frac{\partial F_j}{\partial x_j} dV = \sum_k \int_{\Omega_k} S dV,$$

Note: Often the formality of Σ_k and Ω_k is dropped and is implied over the integration

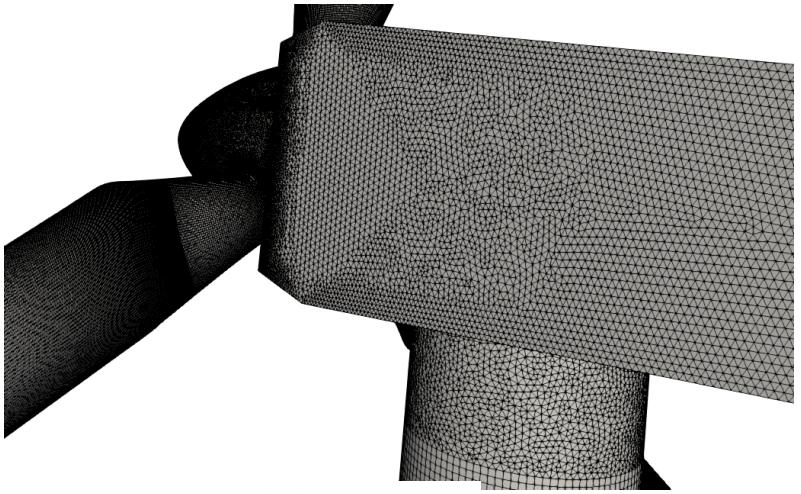
Mesh-based Methods Basic Concept: Sub-divide domain into *Finite Elements and Numerically Integrate*

- Define basis within the domain: $\phi(x,y) \approx \phi^e(x,y) = \sum_j^{nodes} \phi_j^e N_j^e(x,y)$
- While restricting to a Lagrange nodal basis, $N_j^e(x_k, y_k) = \delta_{j,k}$

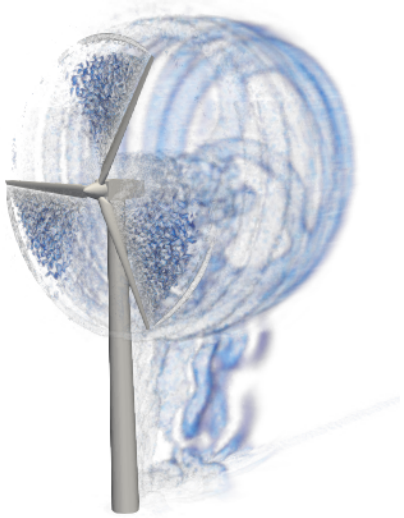


Reality: Meshing time for complex applications remains a significant bottleneck!

- Many applications of interest to SNL contain complex geometries
- low-Mach fluids users interested in high-quality simulation results tend towards hexahedral-based topologies (if possible)
- However, if a scheme is "design-order" accurate, any topology may suffice as it is simply a matter of mesh size and efficiency – not unlike the active discussion on low- vs higher-order
- Sometimes, the penetration of a low-Mach fluids physics addition in common analysis is high as the meshing can be prohibitively complex



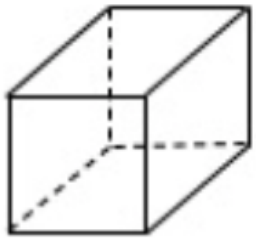
Very complex world - stair-stepped!

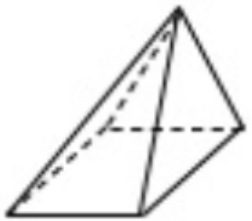


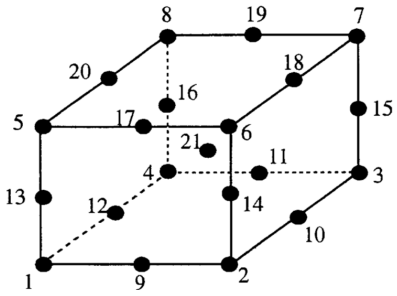
UUR Example: Vestas V27 225 kw hybrid low-order hex/tet/pyr/wedge

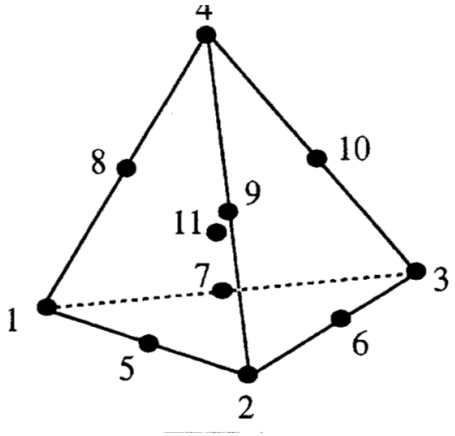
Examples of Various Topologies

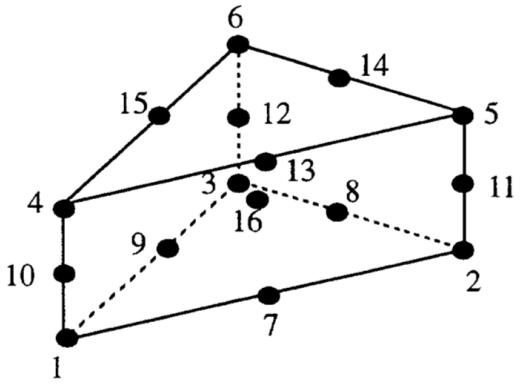
Hex8 Tet4 Pyramid5 Wedge6 Arbitrary











Higher-order promoted elements (Hex27, Tet10, Wedge16, Hex64, etc.)

Fundamentals of Discretization (Review)

- Given a partial differential equation (PDE) and associated volumetric form:

$$\rho C_p \frac{\partial T}{\partial t} - \frac{\partial}{\partial x_j}\lambda\frac{\partial T}{\partial x_j} = 0 \qquad \int \rho C_p \frac{\partial T}{\partial t} dV - \int \frac{\partial}{\partial x_j}\lambda\frac{\partial T}{\partial x_j} dV = 0$$

- Applying Gauss Divergence provides the standard finite volume form:

$$\int \frac{\partial q_j}{\partial x_j} dV = \int q_j n_j \, dS \qquad \int \rho C_p \frac{\partial T}{\partial t} dV - \int \lambda\frac{\partial T}{\partial x_j} n_j \, dS = 0$$

- We can also multiple PDE by an arbitrary test function, w, and integrate over a volume,

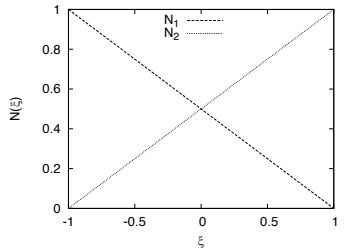
$$\int w\rho C_p \frac{\partial T}{\partial t} dV - \int w\frac{\partial}{\partial x_j}\lambda\frac{\partial T}{\partial x_j} dV = 0$$

Next, integrate by parts and apply Gauss-Divergence:

$$\int w\rho C_p \frac{\partial T}{\partial t} dV + \int \frac{\partial w}{\partial x_j}\lambda\frac{\partial T}{\partial x_j} dV - \int w\lambda\frac{\partial T}{\partial x_j} dS = 0$$

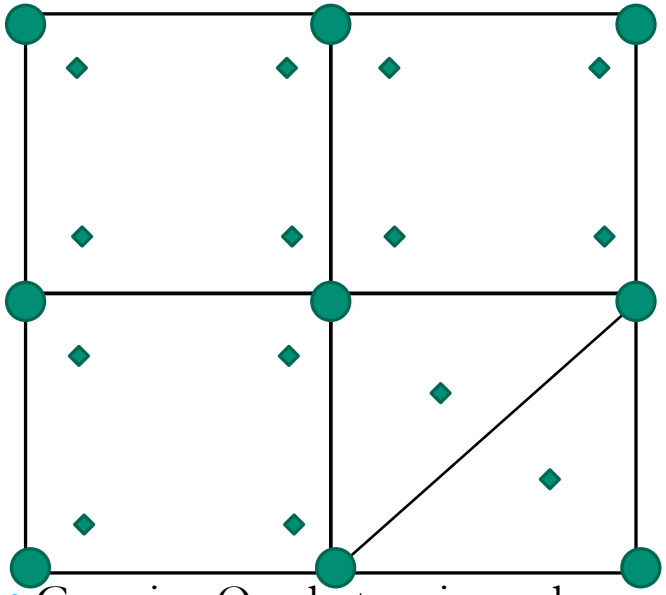
Sometimes, you see T as T^h - the trial space

The Finite Element Method (FEM)



- Consider an alternative approach in which a finite element method (FEM) is employed
- Define an underlying nodal basis with the element:

$$T(x_k) \approx \sum_{i=1}^{npe} N_i(x_k) T_i \qquad \frac{\partial T(x_k)}{\partial x_j} \approx \sum_{i=1}^{npe} \frac{\partial N_i(x_k)}{\partial x_j} T_i$$



- Gaussian Quadrature is used
- On a -1:1 range, +/- sqrt(3)/3

- Consider a simple heat conduction model PDE,

$$\rho C_p \frac{\partial T}{\partial t} - \frac{\partial}{\partial x_j} \lambda \frac{\partial T}{\partial x_j} = 0$$

- Integrate using a test function,

$$\int w \rho C_p \frac{\partial T}{\partial t} dV - \int w \frac{\partial}{\partial x_j} \lambda \frac{\partial T}{\partial x_j} dV = 0$$

- Integration-by-parts (with G-D) provides:

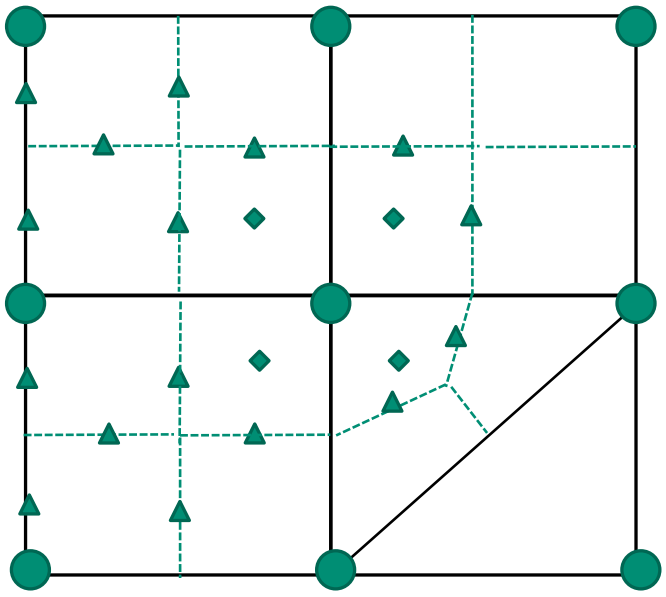
$$\int w \rho C_p \frac{\partial T}{\partial t} dV + \int \frac{\partial w}{\partial x_j} \lambda \frac{\partial T}{\partial x_j} dV - \int w \lambda \frac{\partial T}{\partial x_j} n_j dS = 0$$

- Iterate element quadrature points
- Note that N can be arbitrary in order (shown here for a linear)

The Hybrid Control-Volume Finite Element Method (CVFEM)

- A combination between the edge-based vertex-centered and FEM is the method known as Control Volume Finite Element ()
- A dual mesh is constructed to obtain flux and volume quadrature locations
- As with FEM, a basis is defined:

$$T(x_k) \approx \sum_{i=1}^{npe} N_i(x_k) T_i \qquad \frac{\partial T(x_k)}{\partial x_j} \approx \sum_{i=1}^{npe} \frac{\partial N_i(x_k)}{\partial x_j} T_i$$



Dual-volume definition

- Integration-by-parts over test function w:

$$\int w\rho C_p \frac{\partial T}{\partial t} dV + \int \frac{\partial w}{\partial x_j} \lambda \frac{\partial T}{\partial x_j} dV - \int w\, \lambda \frac{\partial T}{\partial x_j} n_j dS = 0$$

- However, define a test function, w, as a piece-wise constant function (Heavyside) to be 1 inside the dual volume and 0 outside. Gradient is a Dirac-delta function:

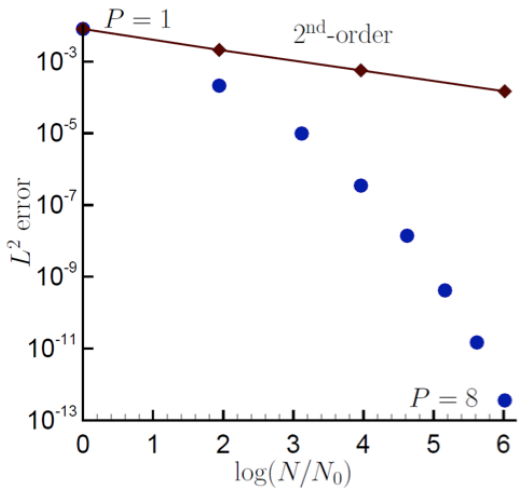
$$\frac{\partial w}{\partial x_j} = -n_j \delta(x_j - xIP_j)$$

- Leading to:

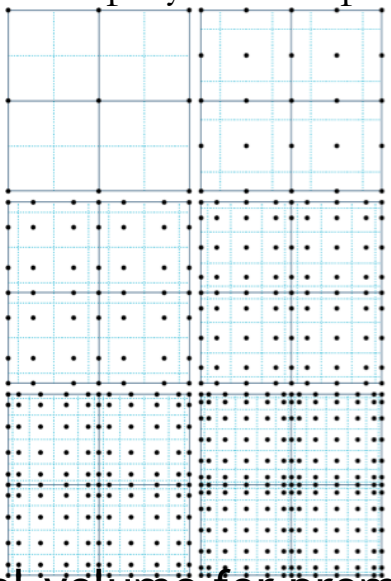
$$\int \rho C_p \frac{\partial T}{\partial t} dV - \int \lambda \frac{\partial T}{\partial x_j} n_j\, dS = 0$$

Control-Volume Finite Element Method Attributes

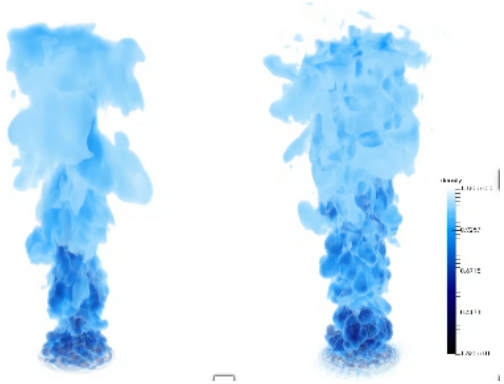
- The CVFEM method, therefore, is a finite volume scheme that is locally conservative, i.e., momentum leaving one dual volume face enters the adjacent dual volume
- However, the gradient operator, like its FEM counterpart is absent of any error due to non-orthogonality
- Since the test-function, w, is different from the underlying basis representation, this method can be considered a Petrov-Galerkin method
- The method can also be promoted in polynomial space



Spectral convergence

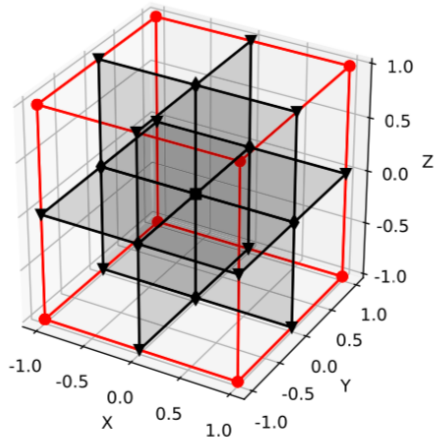


Dual-volume for promoted quad4

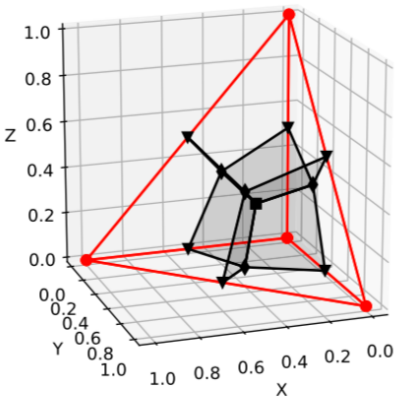


P=1 (left) and P=4 (right)
Helium plume (VR-density)

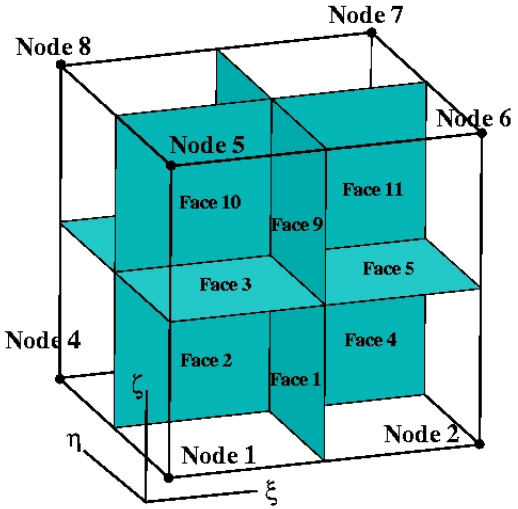
Dual Volume Definitions for Hybrid Meshes

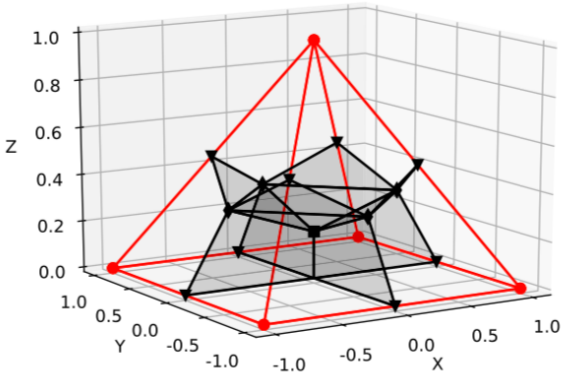


(a) Hexahedral topology (Hex8).

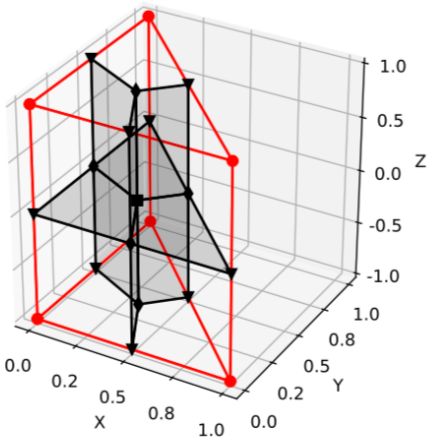


(b) Tetrahedral topology (Tet4).





(c) Pyramid topology (Pyramid5).



(d) Wedge topology (Wedge6).

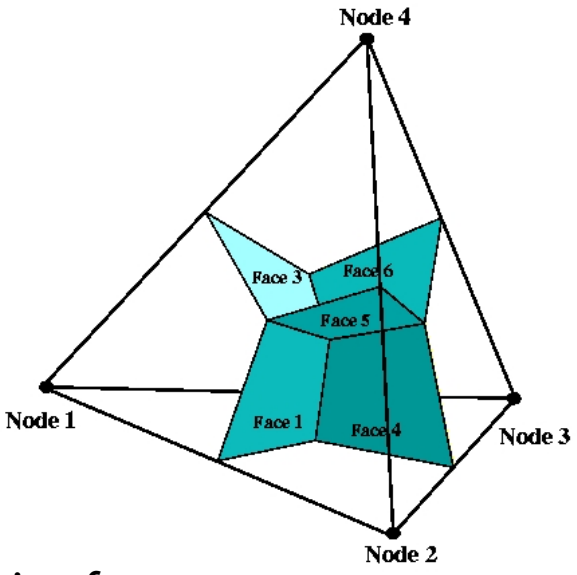
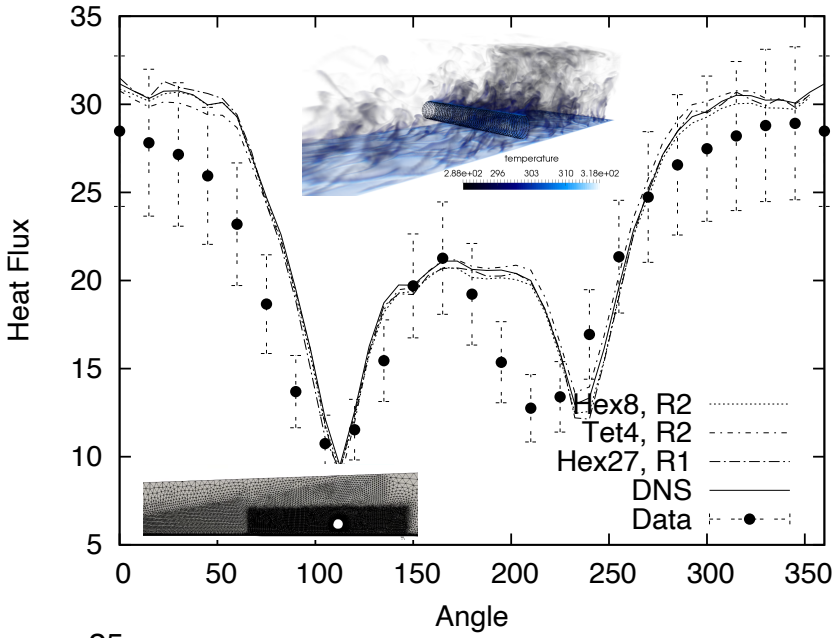


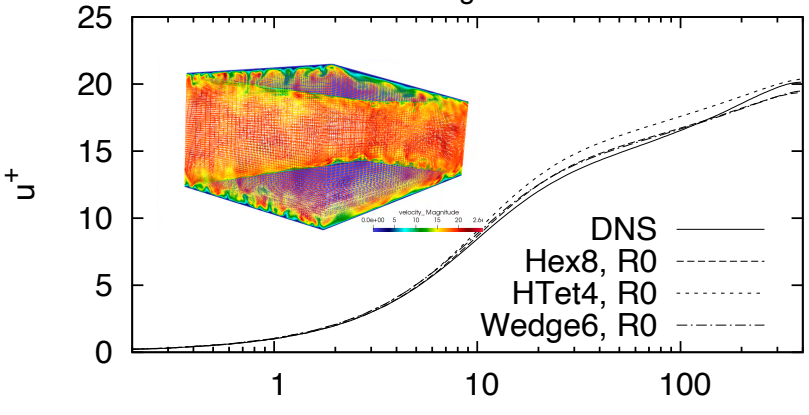
Fig. 1. CVFEM element and dual-volume definition for the low-order topologies.

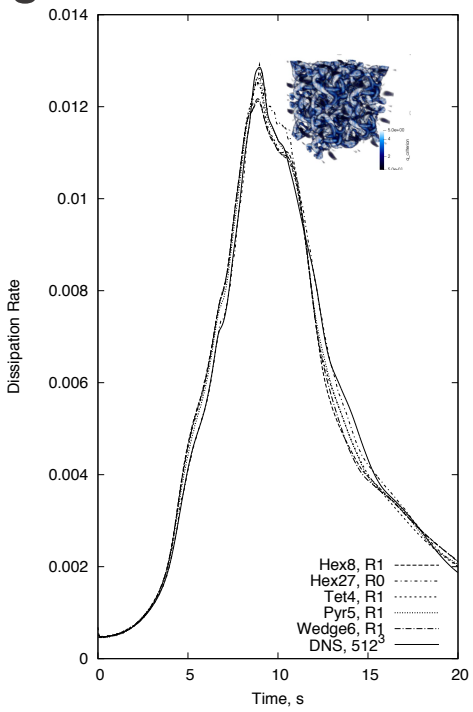
Domino, et. al, “An assessment of atypical mesh topologies for low-Mach large-eddy simulation” 2019

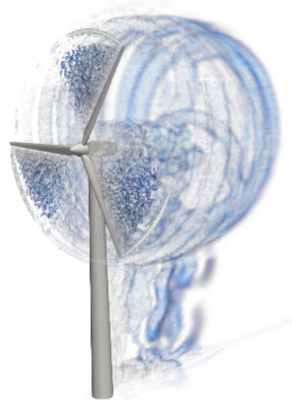
Recent Generalized Unstructured Findings

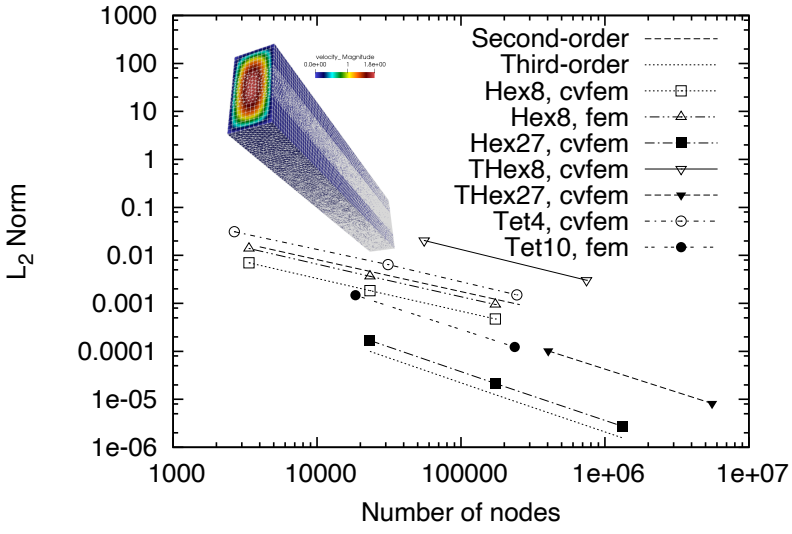
- Domino, et. al, "An assessment of atypical mesh topologies for low-Mach large-eddy simulation" 2019





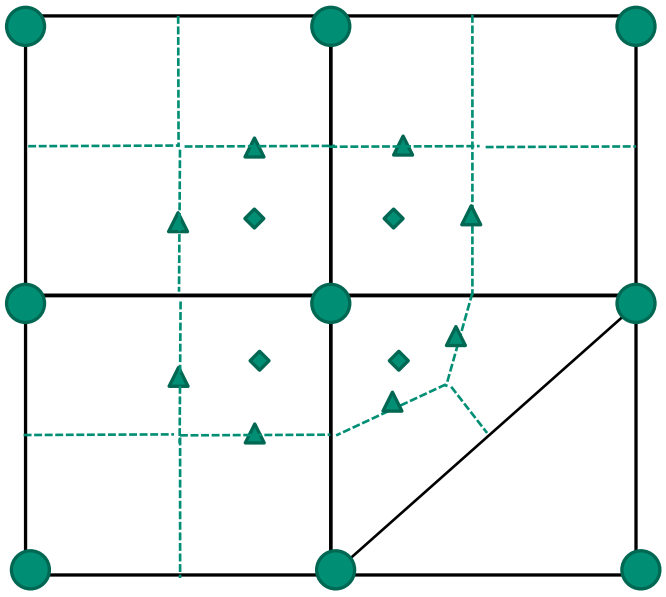






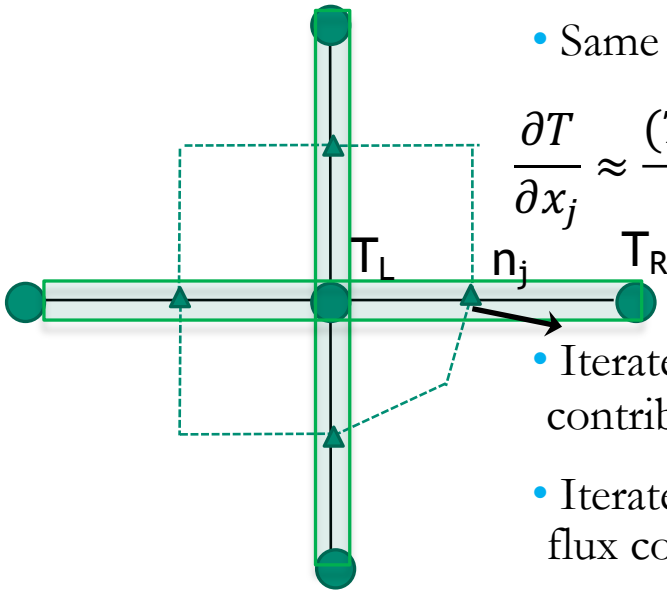
Equal Order Interpolation Edge-Based Vertex-Centered (EBVC) Finite Volume

- All primitives are collocated at the vertices of the elements with equal-order interpolation
- A dual mesh is constructed to obtain flux and volume quadrature locations
- Classic two-state, "L" and "R" approach provides spatially second-order accuracy



Dual-volume definition

- Surface quadrature point (area summed to edge)
- Volume quadrature point (sub-vol summed to node)



Edge-based stencil

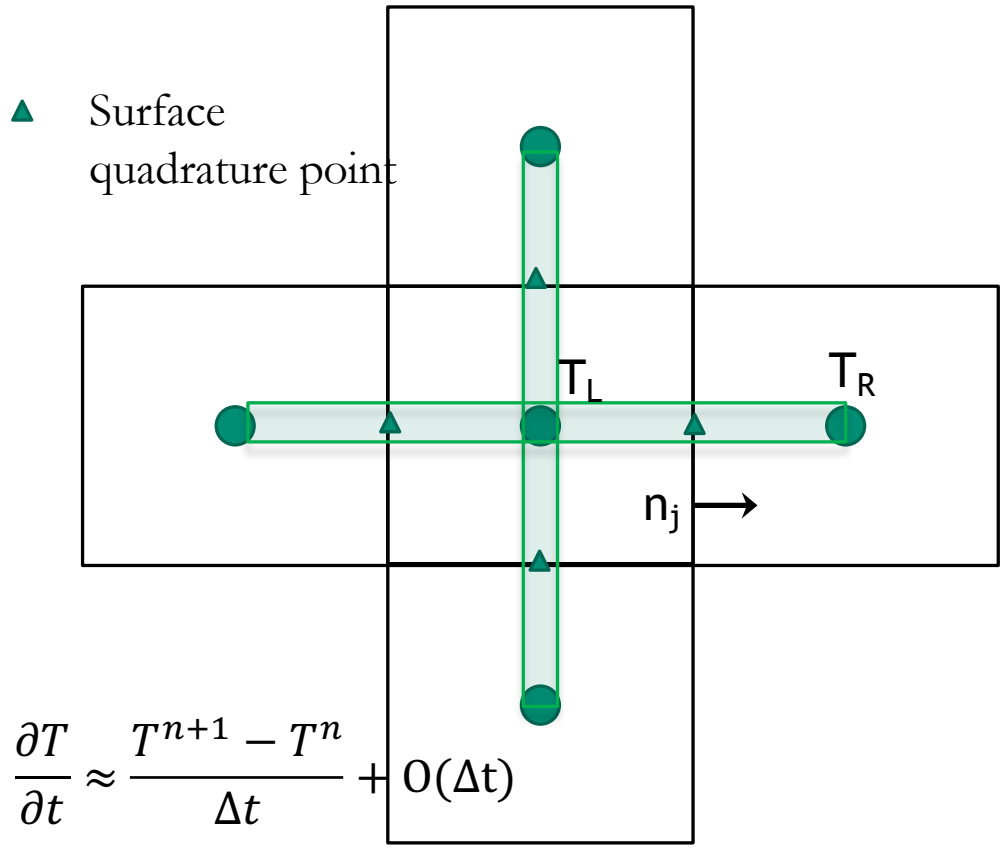
- Same grad-op as CC:

$$\frac{\partial T}{\partial x_j} \approx \frac{(T_R - T_L)n_j}{\|\Delta x_j\|} + O(\Delta x)$$

- Iterate nodes for volume contributions
- Iterate edges for surface flux contribution

Equal-Order Interpolation Cell-Centered (CC) Finite Volume

- All primitives are collocated at the cell-center of the element with equal-order interpolation
- Classic two-state, "L" and "R" approach provides spatially second-order accuracy



- Consider a simple heat conduction model PDE,

$$\rho C_p \frac{\partial T}{\partial t} - \frac{\partial}{\partial x_j} \lambda \frac{\partial T}{\partial x_j} = 0$$

- Integrate over a control volume and use Gauss-Divergence,

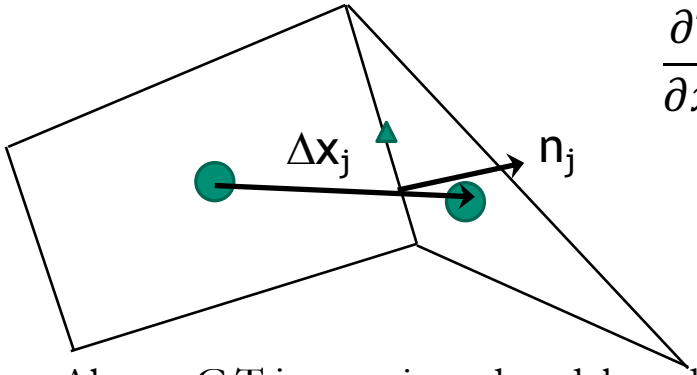
$$\int \rho C_p \frac{\partial T}{\partial t} dV - \int \lambda \frac{\partial T}{\partial x_j} n_j \, dS = 0$$

with: $$\frac{\partial T}{\partial x_j} \approx \frac{(T_R - T_L) n_j}{\|\Delta x_j\|} + O(\Delta x)$$

- Iterate element cell-centers for volume contributions (time/source)
- Iterate element faces for surface flux contribution

Typical Failings for Two-State Discretization Methods

- With two points, only a linear basis can be used.
- Therefore, unstructured CC and EBVC are limited to second-order spatial accuracy
- Non-orthogonality is problematic for gradient-operator



$$\frac{\partial T}{\partial x_j} = G_j T + \left[(T_R - T_L) - G_k T \Delta x_k\right] \frac{A_j}{A_l \Delta x_l}$$

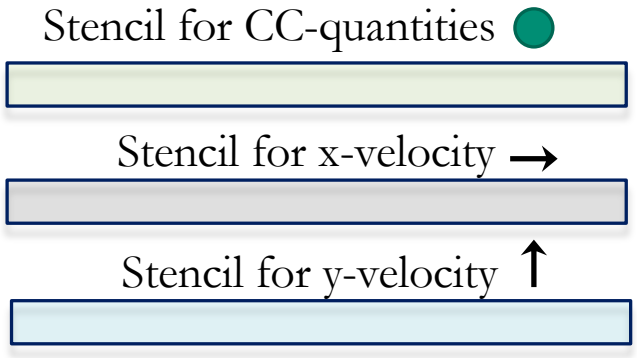
With area vector defined by: $A_j = n_j dS$

- Above, $G_j T$ is a projected nodal gradient at the cell-center, or vertex center:
- Non-orthogonality is simply defined as the mis-alignment of the distance vector between the two "L" and "R" states and the surface normal

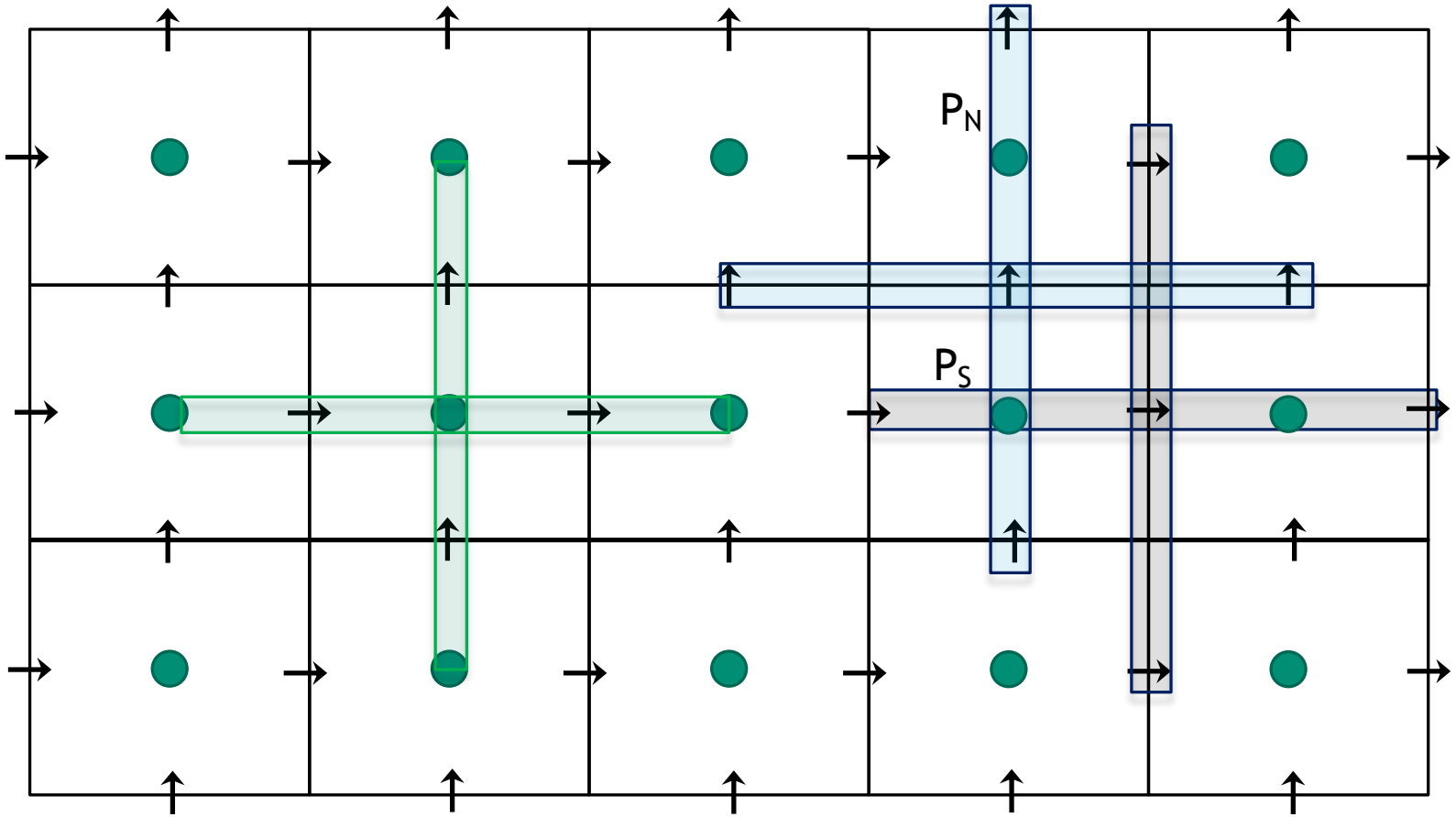
$$G_j T = \frac{\int T A_j}{\int dV}$$

- Both edge- and cell centered-based schemes show degraded accuracy on typical production meshes
- Several non-orthogonality approaches are available, for the best source, see Jasak
  - Jasak, "Error analysis and error estimation for the finite volume method with applications to fluid flow", Imperial College Dissertation, 1996

Classic Staggered Finite Volume



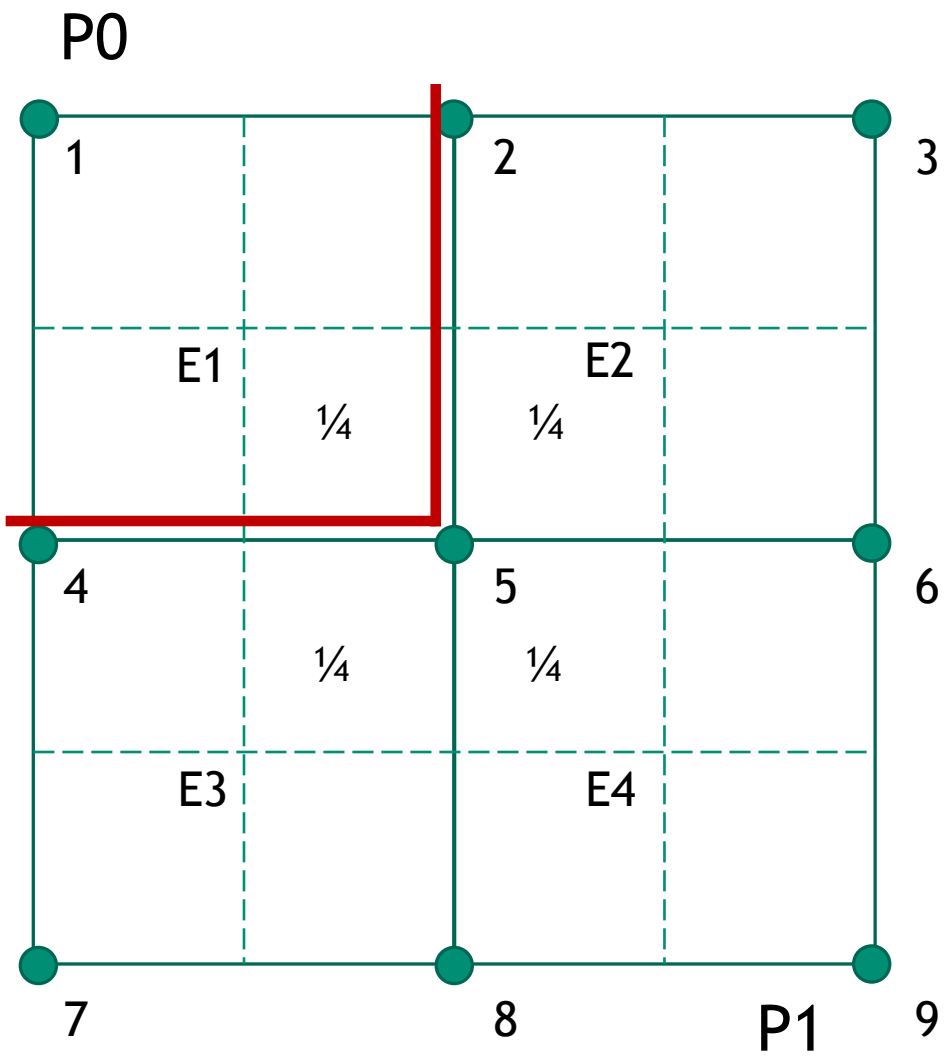
- Velocity degree-of-freedom is staggered relative to pressure and other primitives, e.g., enthalpy, mixture fraction, etc.



Attributes of a Staggered Scheme

- By design, non-orthogonality is absent, however, complex geometry will be stair-stepped
- From a fluids perspective, the operators are ideal, i.e., pressure gradient for momentum is compact, e.g., $(P_E - P_W)\Delta x^{-1}$
  - As will be seen in future lecture topics, the skew-adjoint nature of the Divergence operator, **D**, and Gradient operator, **G**, allows for a Laplace operator, **L** = **DG**
- Can be extended to higher-order
- Frequently, meshing complex geometries can be extremely difficult (consider our V27 example)

A Note on Parallelism… Vertex-based



Sierra Toolkit/Standard Ownership rules:

- P0 locally owns E1 and nodes 1, 2, 4, and 5
- P1 locally owns E2, E3, and E4 and nodes 3, 6, 7, 8, and 9
- P0 and P1 share nodes 2, 4, and 5

Desired Control-Volume Finite Element (CVFEM) Operation:

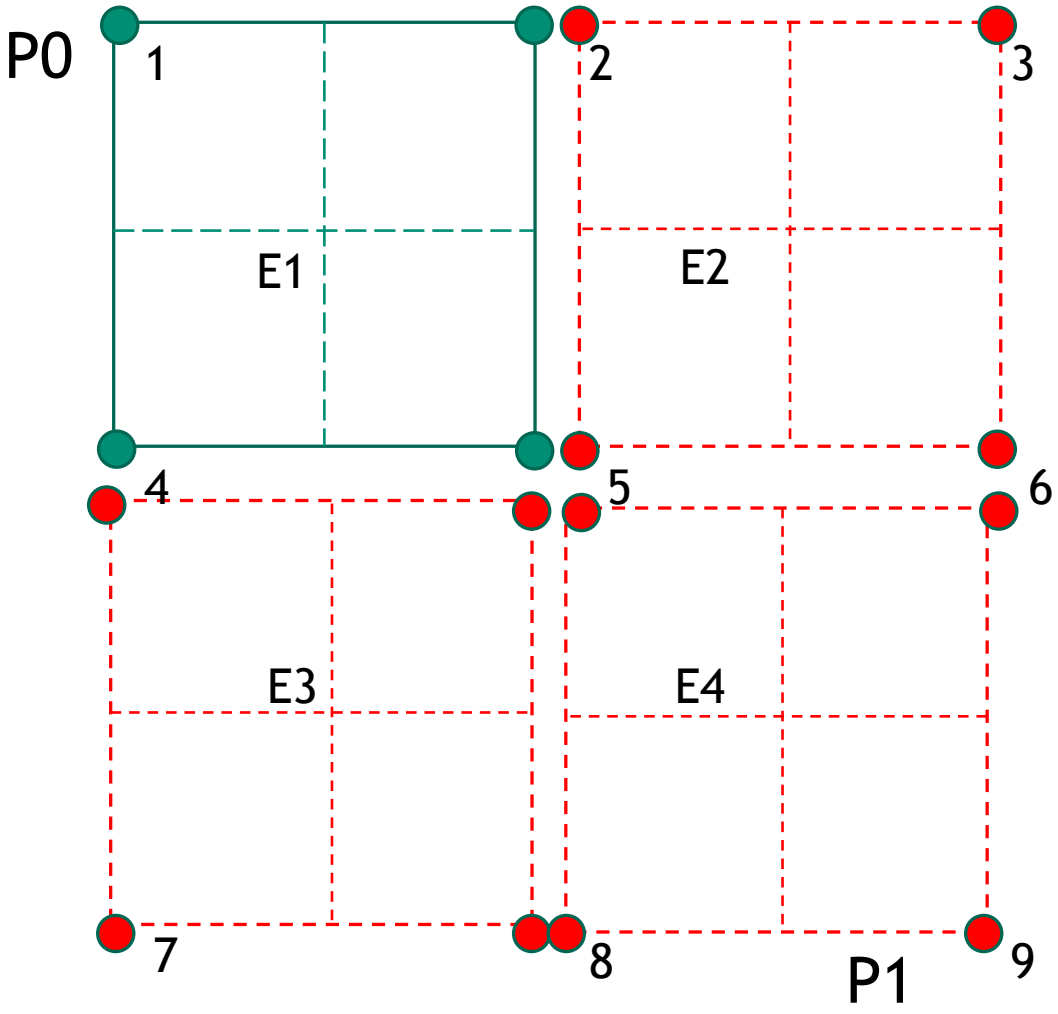
- "nodal volume" assemble for node 5 via iteration of each 1cmx1cm element. Desired nodal volume for node 5 = 1 cm^2 exactly the same on both parallel ranks

Options:

1. Iterate over locally owned elements and locally accumulate nodal volume field followed by a parallel sum and copy owned to shared.

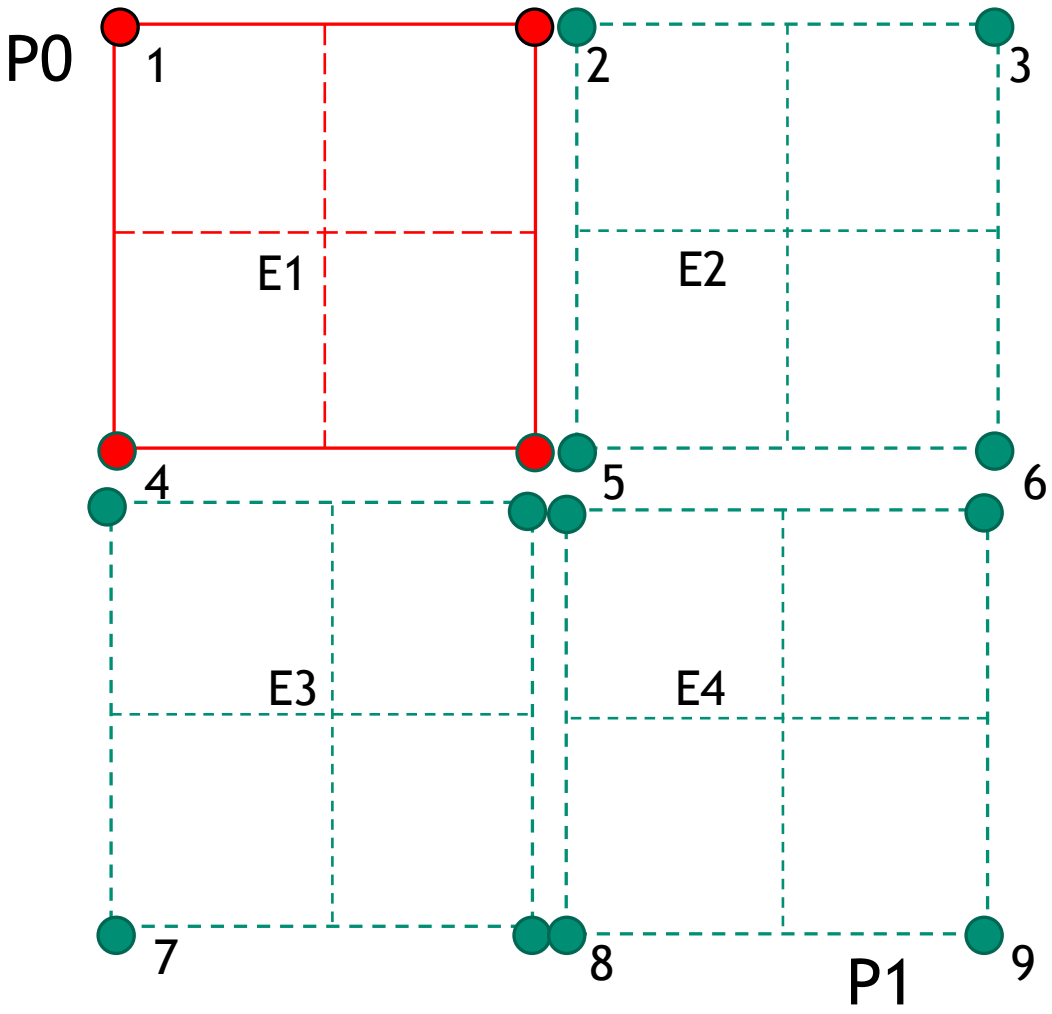
2. Ghost element E2, E3, and E4 to P0 and E1 to P1 (along with coordinates) and iterate locally owned and ghosted elements. Locally accumulate nodal volume field followed by a copy owned to shared.

A Note on Parallelism… Aura-based; View from P0



Ghost P1 -> P0

A Note on Parallelism… Aura-based; View from P1



Ghost P0 -> P1

Model Passive Scalar Transport

- Time/Src/Advection/Diffusion equation for scalar ϕ

$$\frac{\partial \rho \phi}{\partial t} + \frac{\partial \rho u_j \phi}{\partial x_j} - \frac{\partial}{\partial x_j}\left(\frac{\mu}{Sc}\frac{\partial \phi}{\partial x_j}\right) = S^{\phi}$$

$$\int w \left(\frac{\partial \rho \phi}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_j \phi - \frac{\mu}{Sc}\frac{\partial \phi}{\partial x_j}\right) - S^{\phi}\right) dV = 0.$$

- Here, w is the weight function, or an arbitrary function that is differentiable - at least once
  - Any ideas why? What are the properties of the above PDE?
- In this example, let density, velocity, and diffusive flux coefficient be prescribed
- This equation, with a prescribed velocity, can be used to model trace species contaminant transport, smoke, etc.
- The above equation is termed a weighted-residual statement of the original PDE
- Thus far, it is exact

Model Passive Scalar Transport; Heaviside Test Function

- Time/Src/Advection/Diffusion equation for scalar f

$$\int w \left(\frac{\partial \rho \phi}{\partial t} + \frac{\partial}{\partial x_j} \left(\rho u_j \phi - \frac{\mu}{Sc} \frac{\partial \phi}{\partial x_j} \right) - S^{\phi} \right) dV = 0.$$

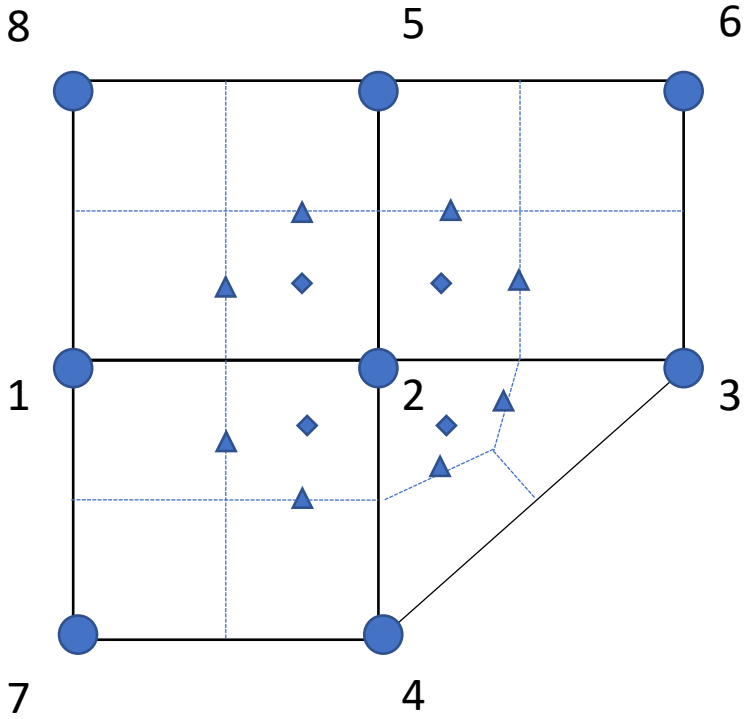
$$\int w \frac{\partial \rho u_j \phi}{\partial x_j} dV = - \int \rho u_j \phi \frac{\partial w}{\partial x_j} dV + \int w \rho u_j \phi n_j dS$$

$$\frac{\partial w}{\partial x_j} = -n_j \delta(x_j - x_j^{ip}),$$

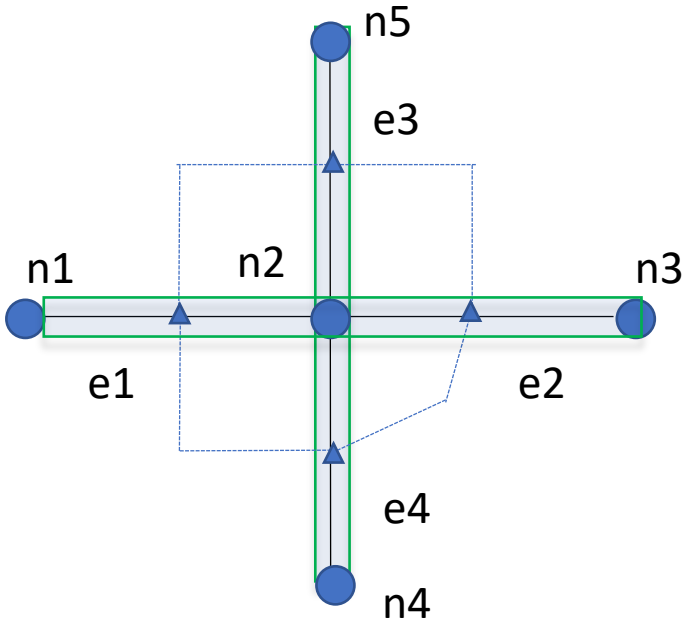
- The CVFEM and EBVC test function meets our requirement in that it is differentiable once
- This definition also allows us to view such methods as Petrov Galerkin schemes

Deep Dive on EBVC

- Recall, EBVC is a discretization scheme that:
  - Iterates over locally-owned nodes for Time/Source/etc. (volumetric-based terms)
  - Iterates over locally-owned edges for Advection/Diffusion/etc. (integrated by parts terms)
  - Below is the patch of elements connected to node 2 (a global matrix row number)
  - Area vector at the edge and dual volume at the node require the node:element connectivity whose quantities are determined in a pre-processing step



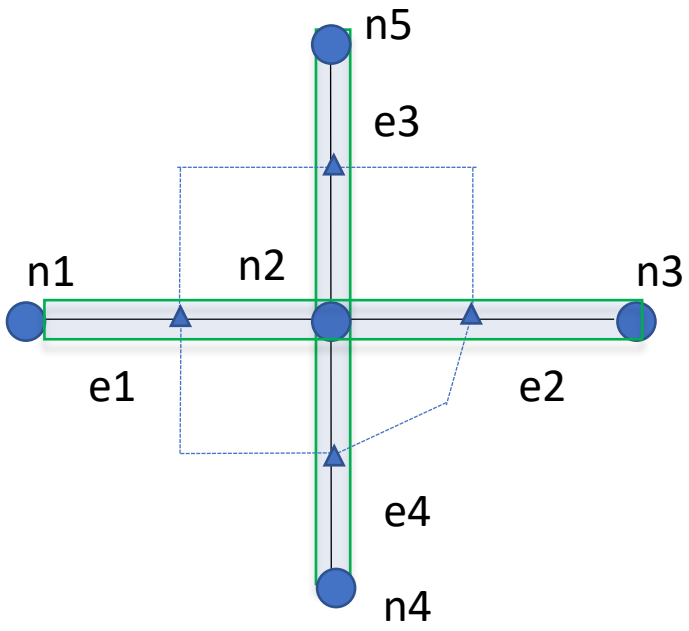
Sample patch of elements

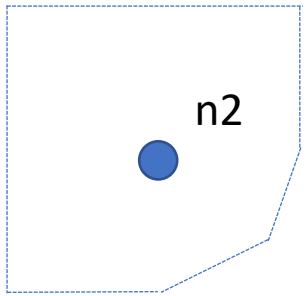


Note: No expectation of nodal ordering..

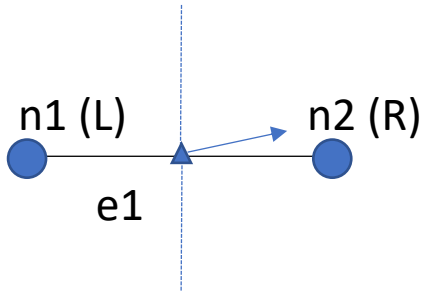
Deep Dive on EBVC: Node and Edge-loops

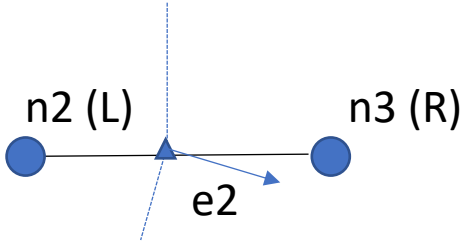
- Recall, EBVC is a discretization scheme that:
  - Iterates over locally-owned nodes for Time/Source/etc. (volumetric-based terms)
  - Iterates over locally-owned edges for Advection/Diffusion/etc. (integrated by parts terms)





$$\int S^{\phi} dV \approx \sum_{nd} S^{\phi}_{nd} V_{nd}.$$

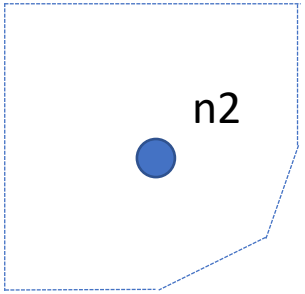




$$\int w \frac{\partial \rho u_j \phi}{\partial x_j} dV \approx \sum_{ip} (\rho u_j)_{ip} \phi_{ip} n_j dS \approx \sum_{ip} \dot{m}_{ip} \phi_{ip}$$

Deep Dive on EBVC: Implicit Time Discretization

- Let us define a general implicit time integrator that is A-stable
  - For y' = k y ; y(0) = 1; y(t) = e^{kt} solution approaches zero as time increases for k < 0
- Backward Euler (two state) and is first-order accurate (A-stable)
- BDF2 (three state) is second-order accurate (A-stable)
- This term is assembled over a nodal iteration



$$\int \frac{\partial \rho \phi}{\partial t} dV \approx \sum_{nd} \frac{(\gamma_1 \rho_{nd}^{n+1} \phi_{nd}^{n+1} + \gamma_2 \rho_{nd}^{n} \phi_{nd}^{n} + \gamma_3 \rho_{nd}^{n-1} \phi_{nd}^{n-1})}{\Delta t} V_{nd}.$$

- For uniform time steps:

$\gamma_1 = 3/2$

$\gamma_2 = -2$

$\gamma_3 = 1/2$

Deep Dive on EBVC: Implicit Time Discretization (Code)

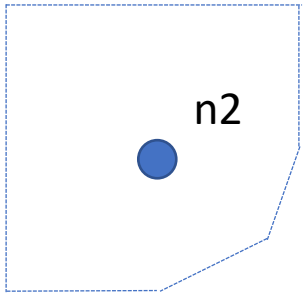
- https://github.com/NaluCFD/Nalu/blob/master/src/ScalarMassBDF2NodeSuppAlg.C

```
//--------------------------------------------------------------------------
//-------- node_execute ----------------------------------------------------
//--------------------------------------------------------------------------
void
ScalarMassBDF2NodeSuppAlg::node_execute(
  double *lhs,
  double *rhs,
  stk::mesh::Entity node)
{
  // deal with lumped mass matrix
  const double qNm1       = *stk::mesh::field_data(*scalarQNm1_, node);
  const double qN         = *stk::mesh::field_data(*scalarQN_, node);
  const double qNp1       = *stk::mesh::field_data(*scalarQNp1_, node);
  const double rhoNm1     = *stk::mesh::field_data(*densityNm1_, node);
  const double rhoN       = *stk::mesh::field_data(*densityN_, node);
  const double rhoNp1     = *stk::mesh::field_data(*densityNp1_, node);
  const double dualVolume = *stk::mesh::field_data(*dualNodalVolume_, node);
  const double lhsTime    = gamma1_*rhoNp1*dualVolume/dt_;
  rhs[0] -= (gamma1_*rhoNp1*qNp1 + gamma2_*qN*rhoN + gamma3_*qNm1*rhoNm1)*dualVolume/dt_;
  lhs[0] += lhsTime;
}
```

Note the implicit term

Deep Dive on EBVC: Source Term Discretization

- Source terms for the edge-based scheme are also assembled over a nodal loop



$$\int S^{\phi} dV \approx \sum_{nd} S^{\phi}_{nd} V_{nd}.$$

- Note that in this scheme, we are using single point quadrature, i.e., the function is evaluated at a single point
- p = 2N -1
  - Where N is the number of integration points and p is the polynomial order
  - For a linear basis, using one-point quadrature is design-order

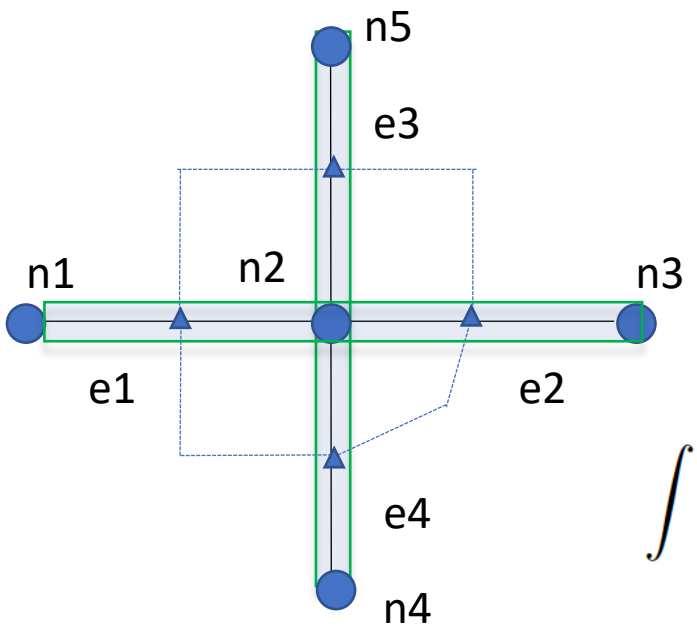
Deep Dive on EBVC: Source Term Discretization (Code)

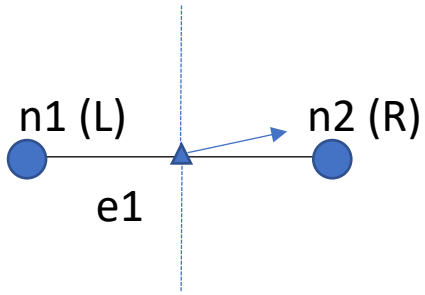
- https://github.com/NaluCFD/Nalu/blob/master/src/user_functions/VariableDensityMixFracSrcNodeSuppAlg.C

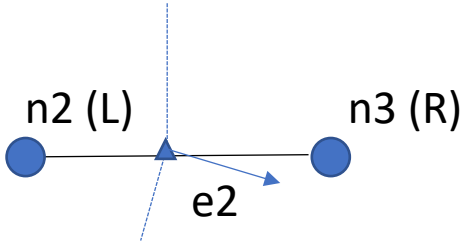
```
//--------------------------------------------------------------------------
//-------- node_execute ----------------------------------------------------
//--------------------------------------------------------------------------
void
VariableDensityMixFracSrcNodeSuppAlg::node_execute(
  double */*lhs*/,
  double *rhs,
  stk::mesh::Entity node)
{
  // deal with lumped mass matrix
  const double *coords = stk::mesh::field_data(*coordinates_, node);
  const double dualVolume = *stk::mesh::field_data(*dualNodalVolume_, node );
  const double x = coords[0];
  const double y = coords[1];
  const double z = coords[2];

  const double src = 0.10e1 * pow(znot_ * cos(amf_ * pi_ * x) * cos(amf_ * pi_ * y) * cos(amf_

  rhs[0] += src*dualVolume;
}
```

Deep Dive on EBVC: Advection Discretization (no stabilization)

- For advection, we have transformed the volume integral to a surface integration
- Therefore, a patch of edges are required for the full assembly at node 2







Note: Advection term must be in divergence form....

$$\int w \frac{\partial \rho u_j \phi}{\partial x_j} dV \approx \sum_{ip} (\rho u_j)_{ip} \phi_{ip} n_j dS \approx \sum_{ip} \dot{m}_{ip} \phi_{ip}$$

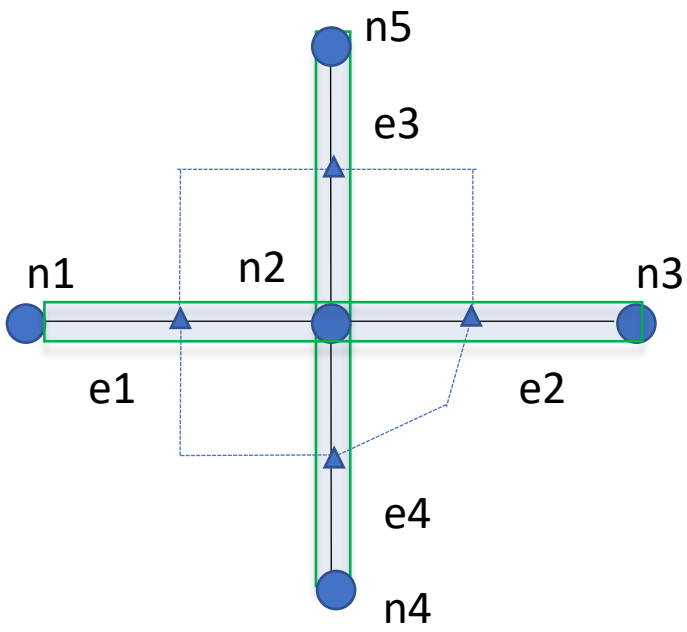
- Recall, that the mass flow rate at an integration point is prescribed
- Moreover, since the integration point is at the edge mid-point, $\phi_{ip} = (\phi^R + \phi^L)/2$
- This is a *central-* or *Galerkin-based* advection operator

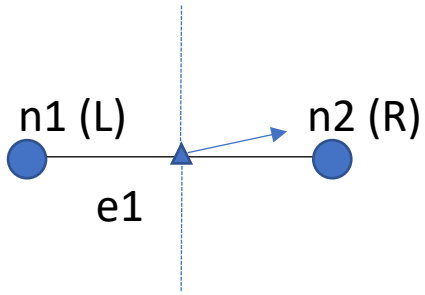
Deep Dive on EBVC: Advection Discretization (Code)

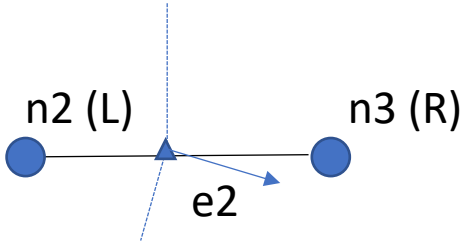
- https://github.com/NaluCFD/Nalu/blob/master/src/AssembleScalarEdgeSolverAlgorithm.C
- This routine includes advection and diffusion

Deep Dive on EBVC: Diffusion Discretization

- For diffusion, we have transformed the volume integral to a surface integration
- Therefore, a patch of edges are required for the full assembly at node 2





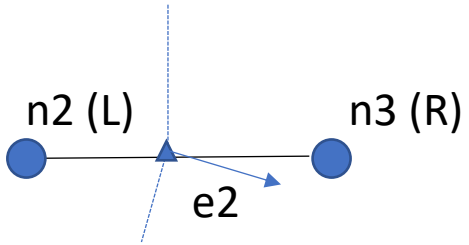


$$\int w \frac{\partial q_j}{\partial x_j} dV \approx -\sum_{ip} \left(\frac{\mu}{Sc}\right)_{ip} \left.\frac{\partial \phi}{\partial x_j}\right|_{ip} n_j dS.$$

Deep Dive on EBVC: Diffusion Discretization

- The edge-based diffusion operator is a bit more complex in that we are computing a gradient at the integration point while using two nodes, the left and right
- In many practical meshes, the distance vector between the edges (dx_j) and the edge area vector will not be perfectly orthogonal
- This non-orthogonality causes inaccuracy when using a simplified approach

$$\frac{\partial \phi_{ip}}{\partial x} = \frac{(\phi_R - \phi_L)}{dx}.$$



- The non-orthogonality approach of Jasek (Ph.D. Imperial College) is a standard cell-centered approach that has been adopted for edge-based schemes
  - CC and EBVC are each two-state schemes and share non-orthogonality issues

$$\left.\frac{\partial \phi}{\partial x_j}\right|_{ip} = \overline{G_j \phi} + \left[(\phi_R - \phi_L) - \overline{G_l \phi} dx_l\right] \frac{A_j}{A_k dx_k}.$$

Deep Dive on EBVC: Diffusion Discretization (code)

- https://github.com/NaluCFD/Nalu/blob/master/src/AssembleScalarEdgeDiffSolverAlgorithm.C

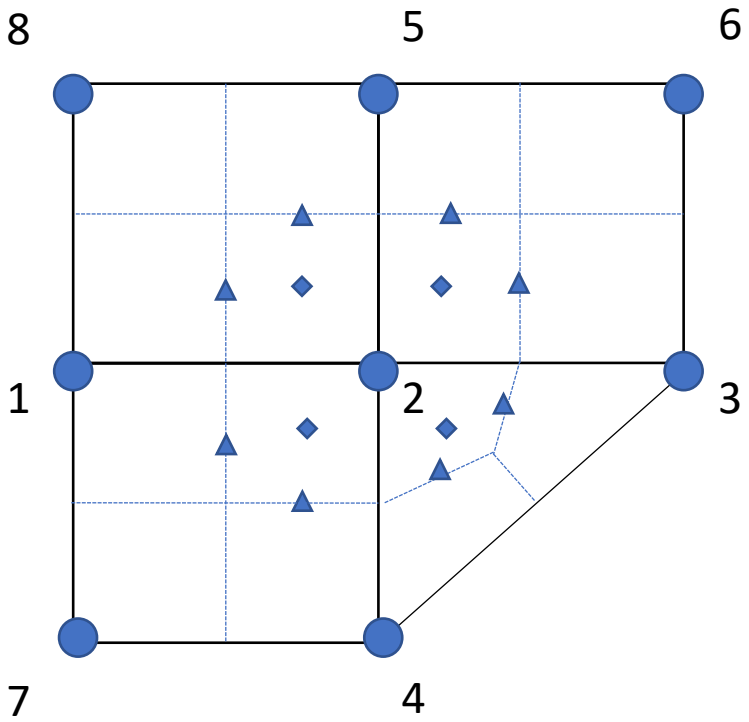
$$-\int w\frac{\mu}{Sc}\frac{\partial\phi}{\partial x_j}dV \approx -\frac{\mu}{Sc}\Big|_{ip}\left[\left(\overline{G_x\phi}A_x+\overline{G_y\phi}A_y\right)+\left(\phi_R-\phi_L\right)\frac{A_xA_x+A_yA_y}{A_x\Delta x_x+A_y\Delta x_y}\right.$$
$$\left.-\left(\overline{G_x\phi}dx+\overline{G_y\phi}dy\right)\frac{A_xA_x+A_yA_y}{A_xdx+A_ydy}\right]. \qquad (21)$$

Deep Dive on EBVC: Example Case

- For a mid-term set of example cases, we have selected a flow past a two-dimensional cylinder at a Reynolds number of 150
- Mesh and input files can be found in the Nalu/examples/street directory
- Familiarize yourself with this physics set via literature surveys
  - As a starting place, see: https://en.wikipedia.org/wiki/Kármán_vortex_street
- In this test case, we have added a passive scalar field whose inflow is zero and wall boundary condition unity. For your final project, feel free to modify inflow and wall parameters as you choose.
- Note that this case uses a conformal hybrid mesh, or a mesh of disparate element topologies, Quad4 and Tri3

Deep Dive on CVFEM

- Recall, CVFEM is a discretization scheme that:
  - Iterates over locally-owned elements for Time/Source/etc. (volumetric-based terms)
  - Iterates over locally-owned elements for Advection/Diffusion/etc. (integrated by parts terms)
  - Below is the patch of elements connected to node 2 (a global matrix row number)

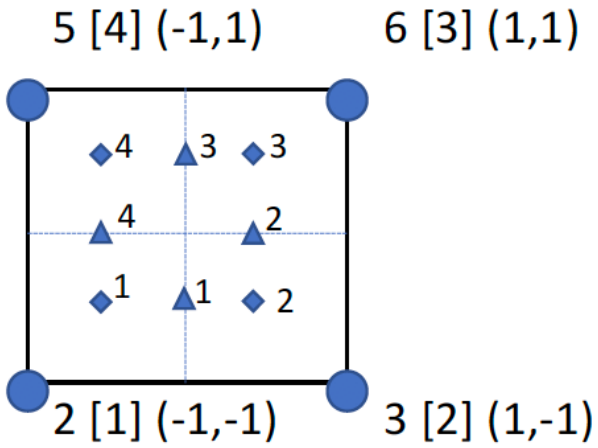


Sample patch of elements

Note: Time and source terms can also be evaluated at nodes without loss of accuracy

Deep Dive on CVFEM: Element-loops

- For each element, recall that a dual volume is nonstructed
- Volume-based contributions are evaluated at the subcontrol volume integration points (diamonds)
- Surface-based contributions are evaluated at the subcontrol surface integration points (triangles)
- We define an isoparametric element than ranges from -1:1 in the ξ– (x-direction) and η– (y-direction) direction



Basis Functions for a Quad4

$$N_1^{ip} = \frac{1}{4}(1-\xi)(1-\eta)$$

$$N_2^{ip} = \frac{1}{4}(1+\xi)(1-\eta)$$

$$N_3^{ip} = \frac{1}{4}(1+\xi)(1+\eta)$$

$$N_4^{ip} = \frac{1}{4}(1-\xi)(1+\eta)$$

For example, the isoparametric coordinate for ip 1 is (0.0, -0.5)

Deep Dive on CVFEM: Implicit Time Discretization

- Backward Euler (two state) and is first-order accurate (A-stable)
- BDF2 (three state) is second-order accurate (A-stable)
- This term is assembled over an element iteration and drives a consistent mass matrix with a full node:element:node connectivity

$$\int \frac{\partial \rho \phi}{\partial t} dV \approx \sum_{scvip} \frac{(\gamma_1 \rho^{n+1}_{scvip} \phi^{n+1}_{scvip} + \gamma_2 \rho^{n}_{scvip} \phi^{n}_{scvip} + \gamma_3 \rho^{n-1}_{scvip} \phi^{n-1}_{scvip})}{\Delta t} V_{scvip},$$

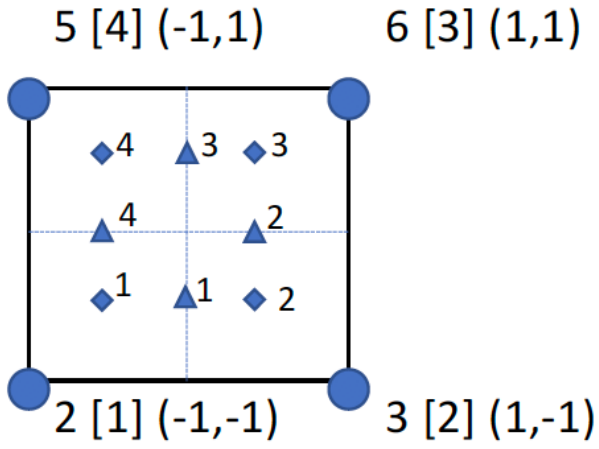
- For uniform time steps:

$\gamma_1 = 3/2$

$\gamma_2 = -2$

$\gamma_3 = 1/2$

$$\phi_{scvip} = \sum_{nd} N^{scvip}_{nd} \phi_{nd}.$$

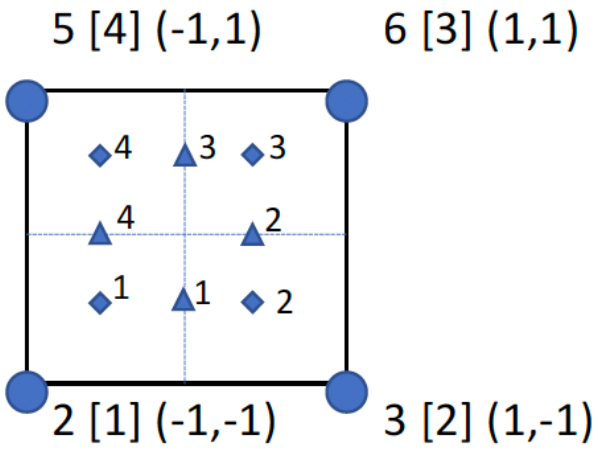


Deep Dive on CVFEM: Implicit Time Discretization (Code)

- https://github.com/NaluCFD/Nalu/blob/master/src/kernel/ScalarMassElemKernel.C

Deep Dive on CVFEM: Source Term Discretization

- Source terms for CVFEM are also assembled over an element or nodal loop
- In some cases, the source term is complex, i.e., includes gradients, which drives either a nodal assembly of these quantities to the nodes or local evaluation



$$\int wS^{\phi}dV \approx \sum_{scvip} S^{\phi}_{scvip} V_{scvip},$$

$$S^{\phi}_{scvip} = \sum_{nd} N^{scvip}_{nd} S^{\phi}_{nd}.$$

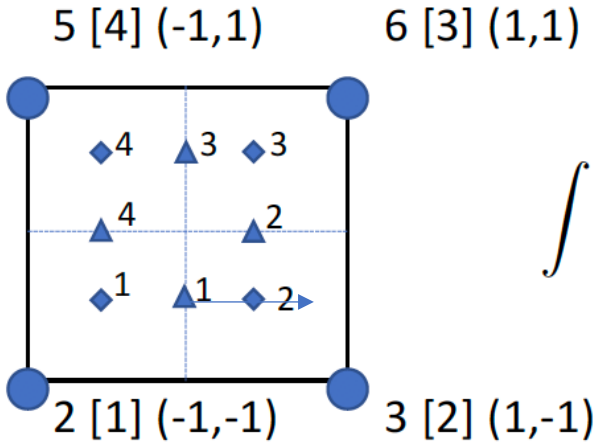
Deep Dive on CVFEM: Source Term Discretization (Code)

- https://github.com/NaluCFD/Nalu/blob/master/src/user_functions/SteadyThermal3dContactSrcElemKernel.C

```
template<typename AlgTraits>
void
SteadyThermal3dContactSrcElemKernel<AlgTraits>::execute(
  SharedMemView<DoubleType**>& /* lhs */,
  SharedMemView<DoubleType *>& rhs,
  ScratchViews<DoubleType>& scratchViews)
{

  // Forcing nDim = 3 instead of using AlgTraits::nDim_ here to avoid compiler
  // warnings when this template is instantiated for 2-D topologies.
  NALU_ALIGNED DoubleType w_scvCoords[3];

  SharedMemView<DoubleType**>& v_coordinates = scratchViews.get_scratch_view_2D(*coordinates_);
  SharedMemView<DoubleType*>& v_scv_volume = scratchViews.get_me_views(CURRENT_COORDINATES).scv_volume;

  // interpolate to ips and evaluate source
  for ( int ip = 0; ip < AlgTraits::numScvIp_; ++ip ) {

    // nearest node to ip
    const int nearestNode = ipNodeMap_[ip];

    // zero out
    for ( int j =0; j < AlgTraits::nDim_; ++j )
      w_scvCoords[j] = 0.0;

    for ( int ic = 0; ic < AlgTraits::nodesPerElement_; ++ic ) {
      const DoubleType r = v_shape_function_(ip,ic);
      for ( int j = 0; j < AlgTraits::nDim_; ++j )
        w_scvCoords[j] += r*v_coordinates(ic,j);
    }

    rhs(nearestNode) += k_/4.0*(2.0*a_*pi_)*(2.0*a_*pi_)*(
      stk::math::cos(2.0*a_*pi_* w_scvCoords[0])
      + stk::math::cos(2.0*a_*pi_* w_scvCoords[1])
      + stk::math::cos(2.0*a_*pi_* w_scvCoords[2]))*v_scv_volume(ip);
  }
}
```

Deep Dive on CVFEM: Advection Discretization (no stabilization)

- For advection, we have transformed the volume integral to a surface integration
- Therefore, a patch of elements are required for the full assembly at node 2



$$\int w \frac{\partial \rho u_j \phi}{\partial x_j} dV \approx \sum_{ip} (\rho u_j)_{ip} \phi_{ip} n_j dS \approx \sum_{ip} \dot{m}_{ip} \phi_{ip}$$

Note: Advection term need not be in divergence form….

- Recall, that the mass flow rate at an integration point is prescribed
- Integration points can also be shifted from the subcontrol surface to the edge mid-poiny (while still using the integration point area vector)
- This is a *central-* or *Galerkin-based* advection operator

Deep Dive on CVFEM: Advection Discretization (Code)

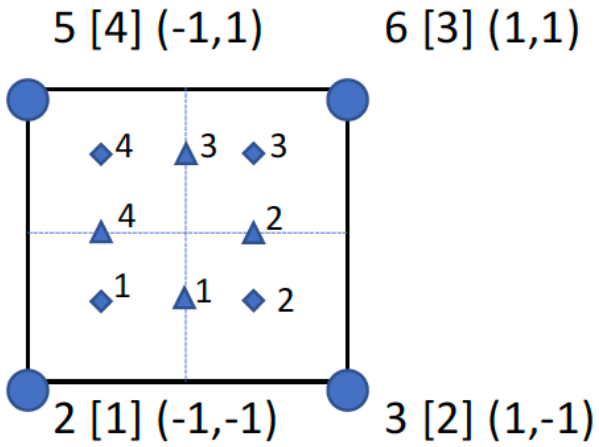
- https://github.com/NaluCFD/Nalu/blob/master/src/kernel/ScalarAdvDiffElemKernel.C
- This routine includes advection and diffusion
- Recall that integration point value is provided by nodal loop over the underlying nodal basis for this element
- Also note that this routine is valid for all types of supported elements – both low- and higher-order

Basis Functions for a Quad4

$$N_1^{ip} = \frac{1}{4}(1-\xi)(1-\eta)$$
$$N_2^{ip} = \frac{1}{4}(1+\xi)(1-\eta)$$
$$N_3^{ip} = \frac{1}{4}(1+\xi)(1+\eta)$$
$$N_4^{ip} = \frac{1}{4}(1-\xi)(1+\eta)$$

Deep Dive on CVFEM: Diffusion Discretization

- For diffusion, we have transformed the volume integral to a surface integration
- Therefore, a patch of elements are required for the full assembly at node 2



$$\begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{bmatrix} \frac{\partial N}{\partial x} \\ \frac{\partial N}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial N}{\partial \xi} \\ \frac{\partial N}{\partial \eta} \end{bmatrix} \quad \mathbf{J}^{-1} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}^{-1}$$

$$\begin{bmatrix} \frac{\partial N}{\partial x} \\ \frac{\partial N}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial N}{\partial \xi} \\ \frac{\partial N}{\partial \eta} \end{bmatrix}$$

$$\int w \frac{\partial q_j}{\partial x_j} dV \approx -\sum_{ip} \frac{\mu}{Sc}_{ip} \frac{\partial \phi}{\partial x_j}_{ip} n_j dS = -\sum_{ip} \frac{\mu}{Sc}_{ip} \sum_{nd} \frac{\partial N_{nd}^{ip}}{\partial x_j} \phi_{nd} A_j^{ip},$$

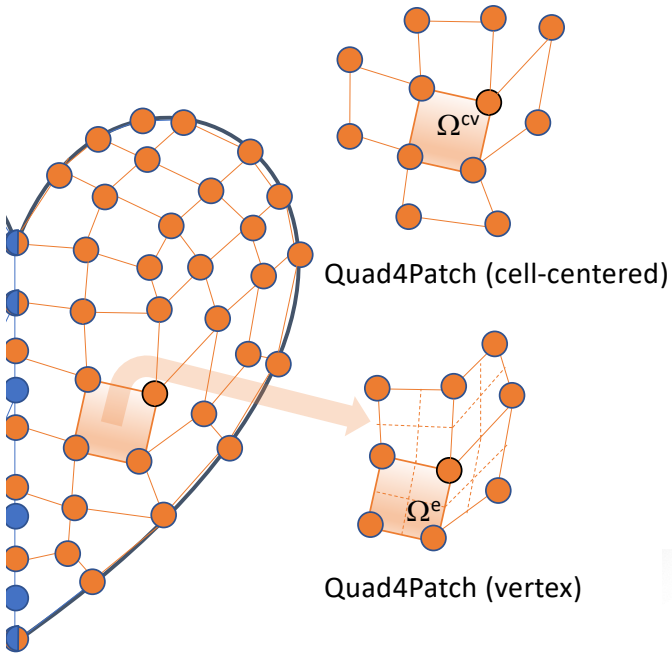
Note that the CVFEM approach is absent any non-orthogonality corrections

Deep Dive on CVFEM: Diffusion Discretization (code)

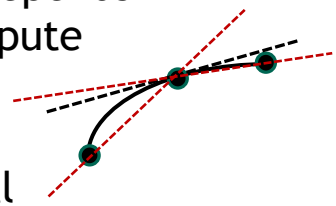
- https://github.com/NaluCFD/Nalu/blob/master/src/kernel/ScalarAdvDiffElemKernel.C

Deep Dive on Nodal Gradient Operator: An Alternative View

- Recall, that the edge-based diffusion operator, and for some choices for the advection operator, a nodal gradient is required
- What is a nodal gradient? As formerly described in the *Computing Gradients* lecture, in a cell-centered context we obtained this via a number of ways, e.g., Green-Gauss, Least Squares, etc.
- Let's take another view…



First, the gradient of a function (other than linear) is discontinuous, i.e., the value at a shared element face depends on which element is used to compute the gradient



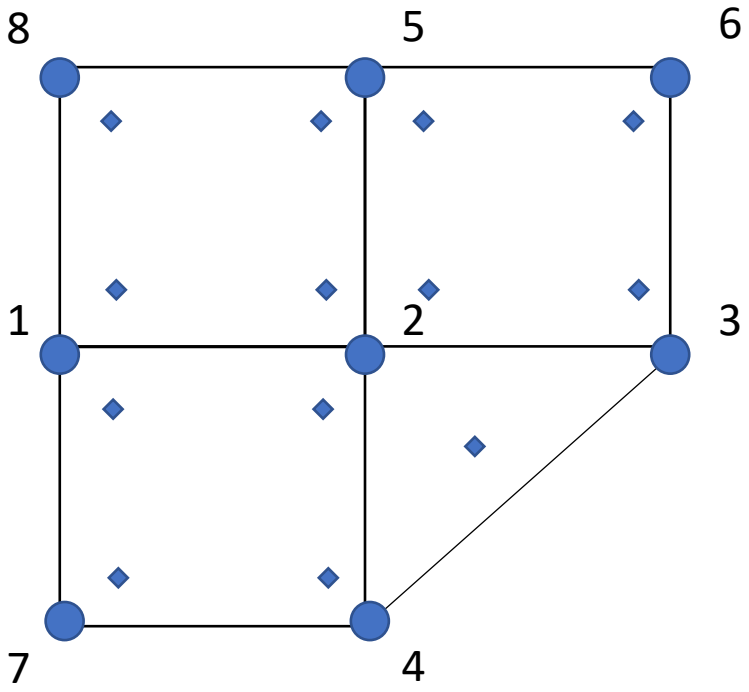
Therefore, we can view the nodal gradient a continuous at the nodes and discontinuous within the elements

Let's minimize this difference: $\frac{1}{2}\left(\frac{\partial \phi}{\partial x_j} - G_j\phi\right)^2$ by solving:

$$\int w G_j \phi dV = \int w \frac{\partial \phi}{\partial x_j} dV \xrightarrow[\text{Lumped-mass}]{} G_j\phi = \frac{\sum_{ip} \phi_{ip} n_j dS}{V}.$$

Deep Dive on FEM

- Recall, FEM is a discretization scheme that:
  - Iterates over locally-owned elements for Time/Source/etc (volumetric-based terms)
  - Iterates over locally-owned elements for Advection/Diffusion/etc. (integrated by parts terms)
  - Below is the patch of elements connected to node 2 (a global matrix row number)

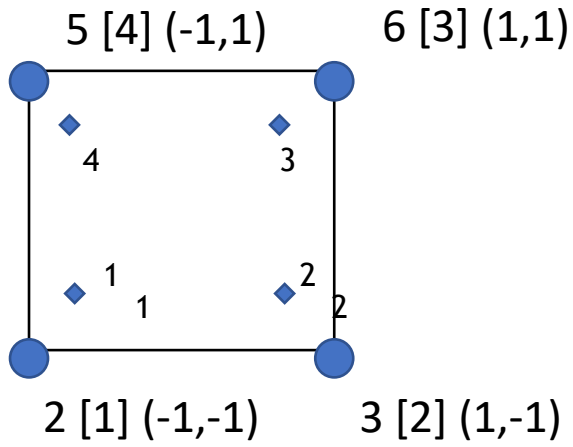


Sample patch of elements

Note: Single integration point location for all terms

Deep Dive on FEM: Element-loops

- For each element, we have a set of integration or quadrature points (no dual-notion)
- We define an isoparametric element than ranges from -1:1 in the ξ– (x-direction) and η– (y-direction) direction
- Gaussian quadrature on a -1:1 range is defined, +/- sqrt(3)/3



Basis Functions for a Quad4

$$N_1^{ip} = \frac{1}{4}(1-\xi)(1-\eta)$$

$$N_2^{ip} = \frac{1}{4}(1+\xi)(1-\eta)$$

$$N_3^{ip} = \frac{1}{4}(1+\xi)(1+\eta)$$

$$N_4^{ip} = \frac{1}{4}(1-\xi)(1+\eta)$$

Deep Dive on FEM: Implicit Time Discretization

- Backward Euler (two state) and is first-order accurate (A-stable)
- BDF2 (three state) is second-order accurate (A-stable)
- This term is assembled over an element iteration and drives a consistent mass matrix with a full node:element:node connectivity

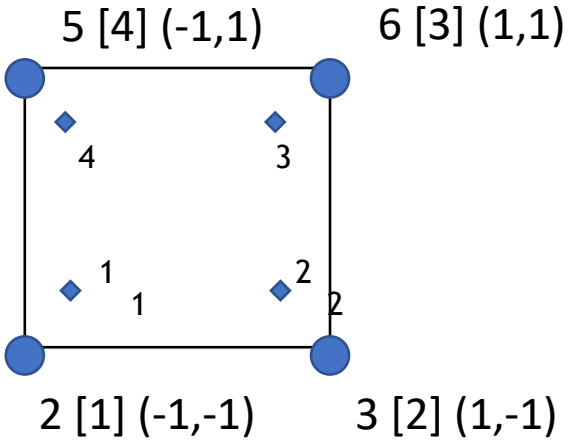
$$\int w \frac{\partial \rho \phi}{\partial t} dV,$$

- For uniform time steps:

$\gamma_1 = 3/2$

$\gamma_2 = -2$

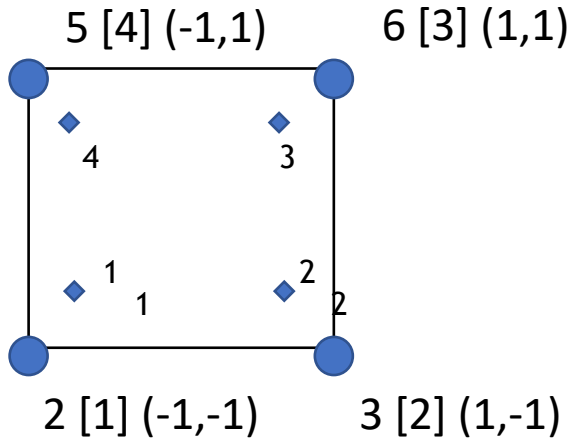
$\gamma_3 = 1/2$



Deep Dive on FEM: Source Term Discretization

- Source terms for FEM are assembled over an element loop
- In some cases, the source term is complex, i.e., includes gradients, which drives either a nodal assembly of these quantities to the nodes or local evaluation

$$\int wSdV.$$



Deep Dive on FEM: Advection Discretization

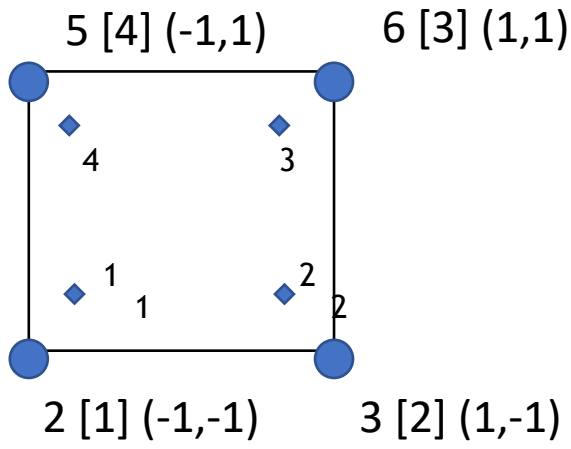
- Advection terms for FEM are assembled over an element loop
- The advection term can be integrated by parts, or not; moreover, the PDE can drive a non-conservative equation form

$$\int w \frac{\partial \rho u_j \phi}{\partial x_j} dV = \int w \rho u_j \phi n_j dS - \int \rho u_j \phi \frac{\partial w}{\partial x_j} dV, \qquad \text{IBP}$$

$$\int w \rho u_j \frac{\partial \phi}{\partial x_j} dV + \int w \phi \frac{\partial \rho u_j}{\partial x_j} dV,$$

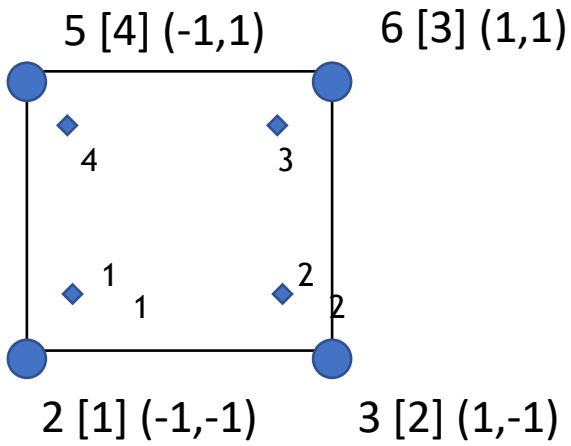
Non-conserved form, like CVFEM, poses no complexity

$$\int w \left(\rho \frac{\partial \phi}{\partial t} + \rho u_j \frac{\partial \phi}{\partial x_j} \right) dV,$$



Deep Dive on FEM: Diffusion Discretization

- For diffusion, we will again use the nodal basis to support a diffusion operator
- Integration-by-parts is desired to remove the second-derivative requirement that would not be possibly for a linear basis



$$\begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{bmatrix} \frac{\partial N}{\partial x} \\ \frac{\partial N}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial N}{\partial \xi} \\ \frac{\partial N}{\partial \eta} \end{bmatrix} \quad \mathbf{J}^{-1} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}^{-1}$$

$$\begin{bmatrix} \frac{\partial N}{\partial x} \\ \frac{\partial N}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial N}{\partial \xi} \\ \frac{\partial N}{\partial \eta} \end{bmatrix}$$

$$\int w \frac{\partial q_j}{\partial x_j} dV = \int w q_j n_j dS - \int q_j \frac{\partial w}{\partial x_j} dV.$$

Note that the FEM approach is absent any non-orthogonality corrections

Deep Dive on FEM: Numerical Integration

- Transform the integral via a change of variables from physical space to isoparametric space

$$\int f dV = \int f(x,y)dxdy = \int \hat{f}(\xi,\eta)|\mathbf{J}|d\xi d\eta,$$

- And apply Gaussian Quadrature over a -1:1 mapping (for a Quad/Hex-based element) by introducing quadrature weights

Deep Dive on FEM: Discretization Kernels (Code)

- Time
  - https://github.com/NaluCFD/Nalu/blob/master/src/kernel/ScalarMassFemKernel.C
- Advection
  - https://github.com/NaluCFD/Nalu/blob/master/src/kernel/ScalarAdvFemKernel.C
- Diffusion
  - https://github.com/NaluCFD/Nalu/blob/master/src/kernel/ScalarDiffFemKernel.C

Common low-Mach Discretization Approaches: Conclusions

- Two-state methods, e.g., cell-centered and EBVC are attractive due to simplicity, however, suffer from non-orthogonality issues in the diffusion operator
- FEM provides a machinery to provide accurate discretizations on non-ideal meshes, however, the same diffusion operator suffers on high-aspect ratio meshes
- CVFEM is a hybrid method that contains the likeable attributes of both FV and FEM (same high-aspect ratio diffusion operator finding)
- Staggered arrangement is well suited for a class of fluid mechanics applications where low-order or simple geometries are found
- Examples for time/source/advection and diffusion provided for EBVC, CVFEM, and FEM
- Recall that Nalu iterates over locally-owned mesh objects as apposed to local+ghosted
  - This is a choice; however, the STK infrastructure allows for both approaches
  - Test yourself on a dual nodal volume assembly?