Interoperable Components for Parallel Mesh Generation and Adaptation

M.S. Shephard and Cameron Smith, Scientific Computation Research Center, Rensselaer Polytechnic Institute

Mark W. Beall and Saurabh Tendulkar, Simmetrix, Inc.

Outline

- Overview of simulation automation component
- Parallel mesh generation
- Parallel mesh adaptation
- Tools for scalable automated simulation
- Scalable solver
- Parallel adaptive research applications
- Developing parallel adaptive industrial applications

Simmetrix Inc.
Enabling Simulation-Based Design

SCOREC

Interoperable Components for Simulation

Increasingly, engineering simulations consider multiple physics models that may act over multiple physical scales

Good CAE tools available for most pieces, however

- Multiple individual tools need to be applied
- Tools not integrated
- Analysis cores run in parallel, but the others steps serial
- Thousands of person-hours invested in validating analysis tools – not reasonable to simply replace those tools

Interoperable components to support simulation definition and automated analysis model generation provide critical core

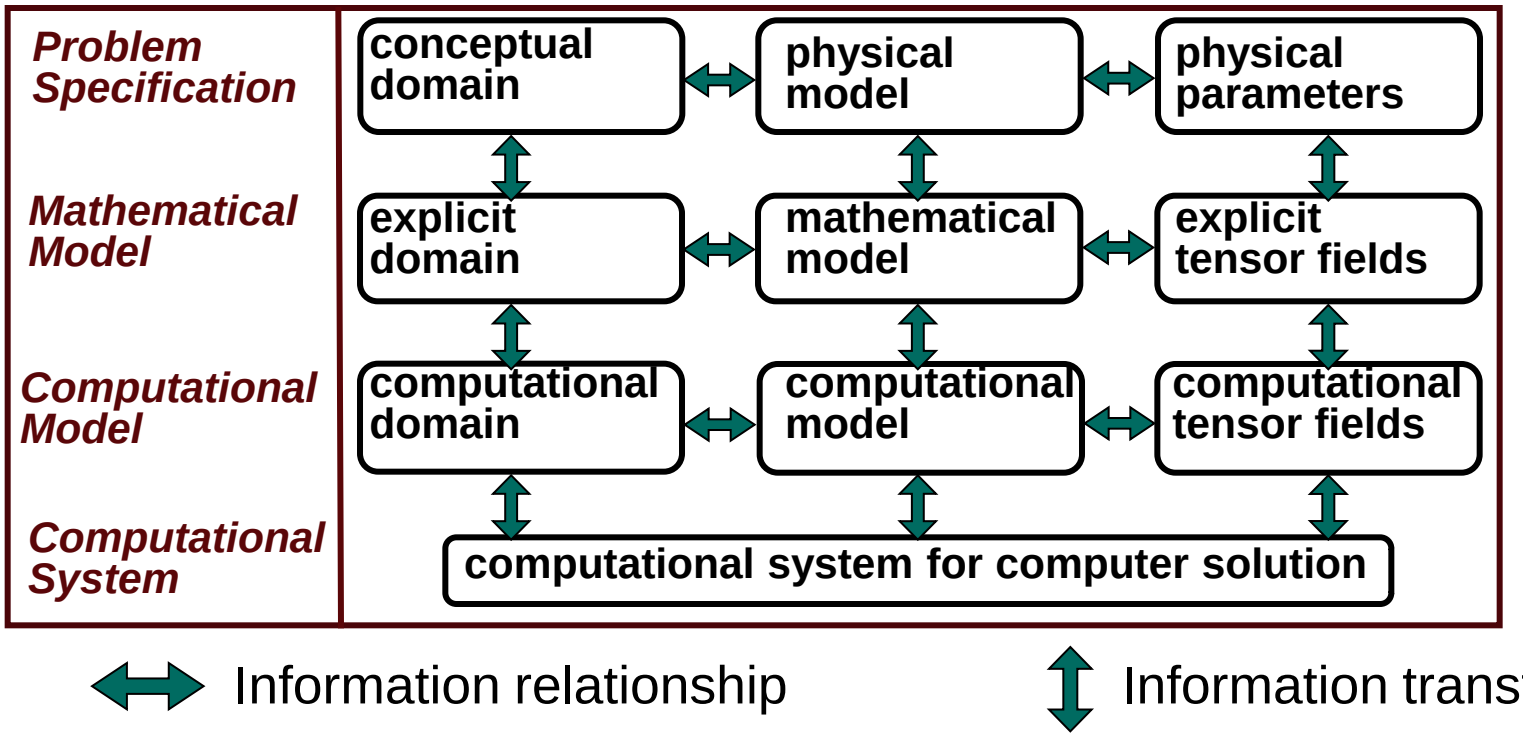
- Need to build on an appropriate abstraction of the information components and fundamental analysis transformation steps
- Must design all components to operate in parallel to deal with the future size of simulation problems and computing systems

Interoperable Components for Simulation

Information sets in the simulation of physical systems

- Domains
- Physical/mathematical models
- Fields defining the model parameters over the domain

Steps single scale engineering simulation (including the evolution of the simulation information)

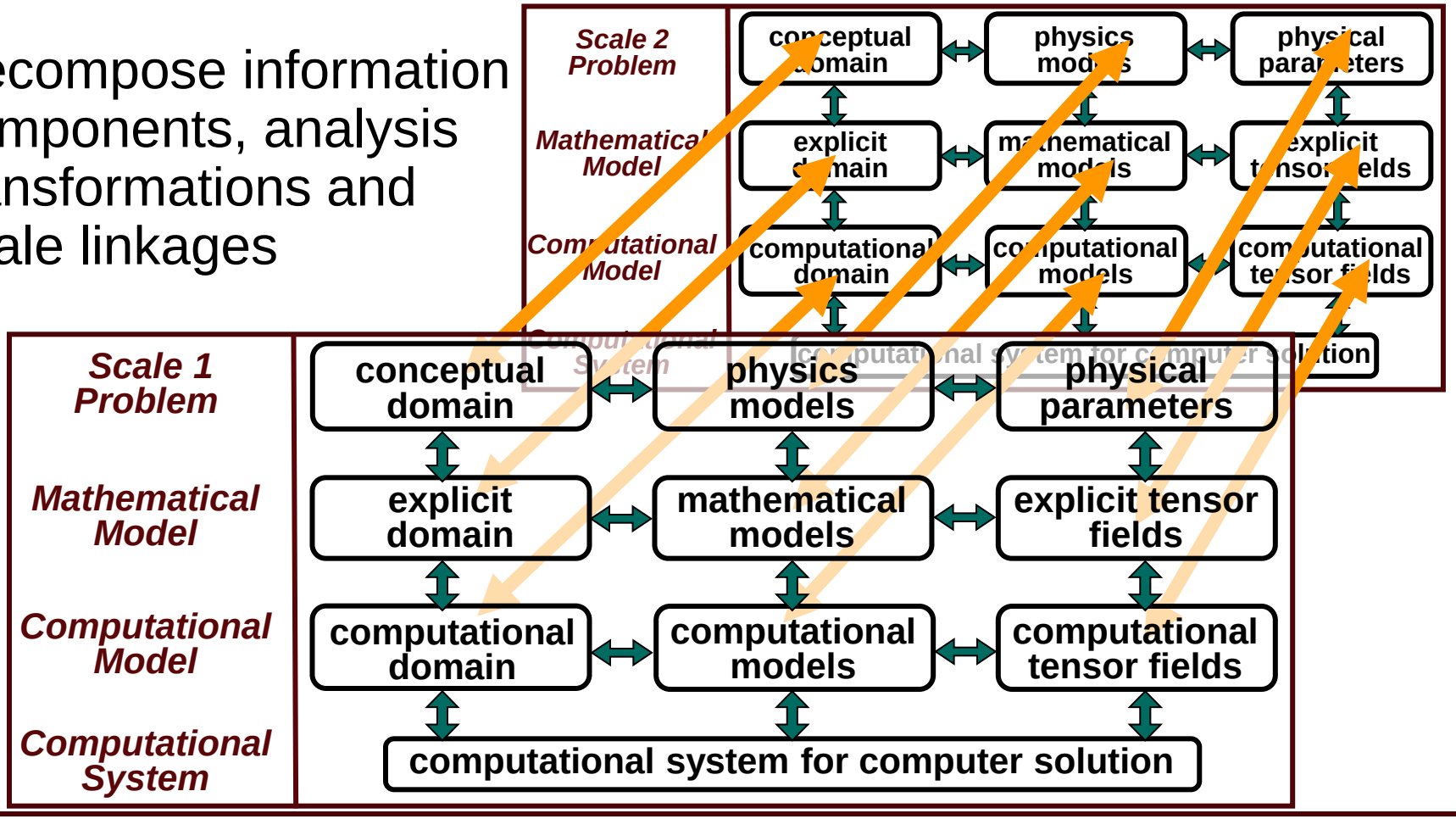


Interoperable Components for Simulation

Multiscale scale simulations introduce any needed scale transformation between information sets at each step

Two scale problem specification

Decompose information components, analysis transformations and scale linkages



Interoperable Components for Simulation

Several efforts underway in the development of components to support the relationships and transformations

Current presentation focuses on the following aspects

- Mesh-based macroscale simulations solving PDE's
- Supporting of simulation automation starting from general problem definitions
- Supporting all components on massively parallel computers
- Bringing parallel adaptive mesh-based simulation procedures to both government labs and industry

Simulation automation components used in examples are various combinations of

- Simmetrix' Simulation Modeling Suite Components
- RPI SCOREC components developed as part of the DOE ITAPS SciDAC center

Automated Adaptive Simulation in Parallel

Components Needed

- Domain representation
  - Mathematical domain in terms of boundary representation (CAD) or image data
  - Discretization of mathematical domain in terms of a mesh
- Fields
  - Tensor parameters needed to quantify the mathematical models in the analysis procedures
  - Input and output tensor fields as discretized on the mesh
- Parallel mesh generation and adaptation
  - Form the system of equations
  - Solve the system of equations
- Scalable mesh-based analysis procedure

All components must operate in parallel

- Preferred parallel structure to support all operations – Distributed Mesh
  - Coordinate interactions of mesh distributed across machine
  - Support movement of entities between parts
  - Interactive with parallel dynamic load balancing procedures

Domain Representation

Explicit Domain Definition Sources

- Geometric Modeling systems – non-manifold boundary representations
- Faceted model representation – mesh models
- Discrete voxel level information – image data

Computational Domain – Mesh

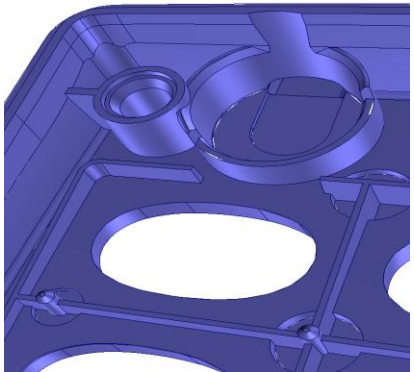
- Also best represented in a compact form of boundary representation

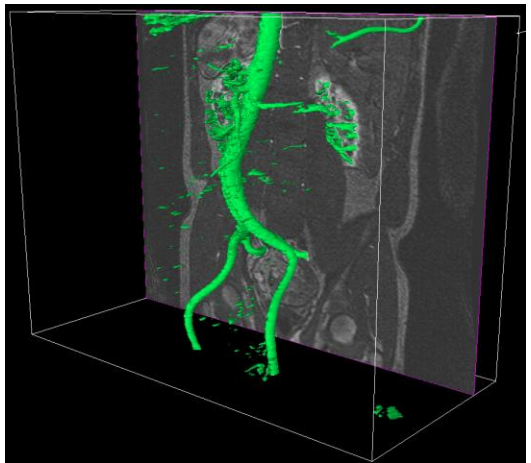
Need to associate

- Mesh to explicit domain definition
- Field information to explicit domain and mesh

Use of boundary representation effectively supports the needs

- Full non-manifold boundary representation for explicit domain
- Compact boundary representation for mesh

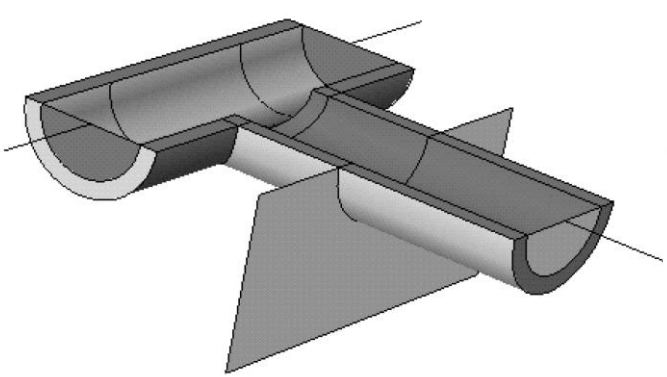


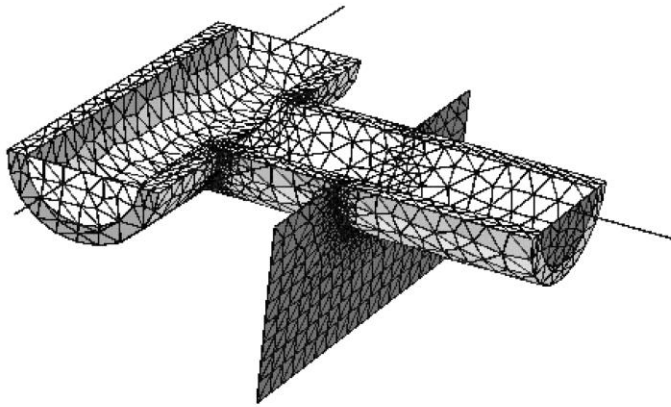


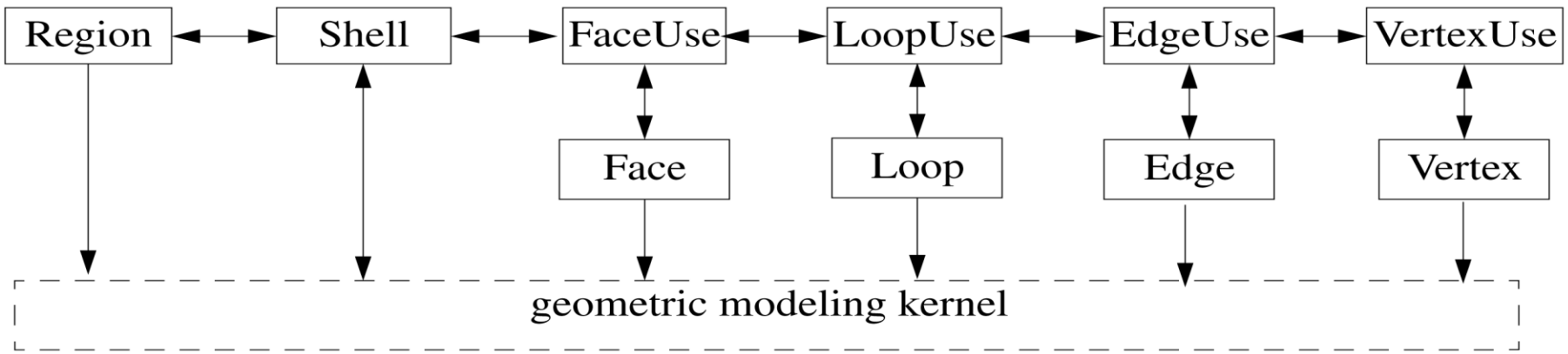
Use of Unified Boundary Representation for Domain

Supports

- General non-manifold model
- Implementation supports obtaining geometry from multiple sources
  - Multiple cade models
  - Cad plus faceted
  - Etc.





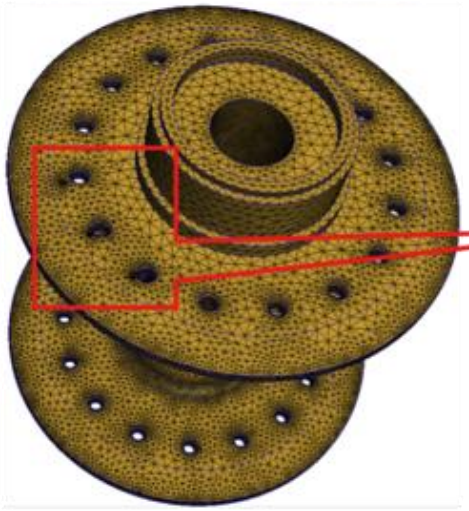


Use of Unified Boundary Representation for Domain

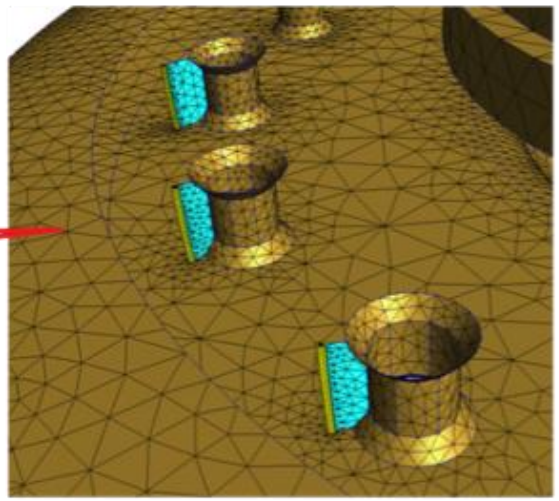
Example – CAD with cracks forming as defined by the simulation

Parasolid model used as an input for crack propagation simulation

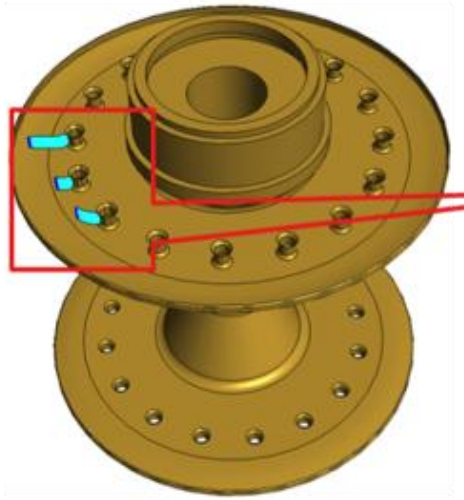
Associated mesh with three cracks defined at the bolt holes



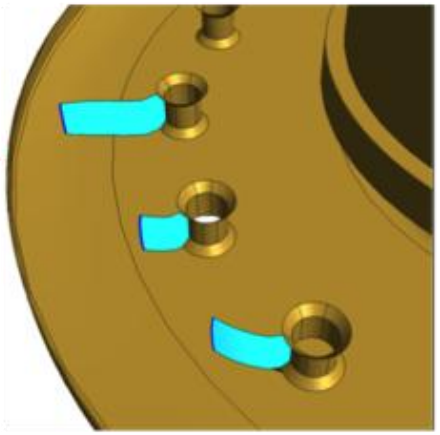
Initial definition (front face hidden for crack visualization) of the three cracks



Geometry representation of the crack propagation paths



Close-up view of the final crack surfaces generated during automatic out-of-plane crack propagation study



Full set of crack fronts generated during automatic crack propagation

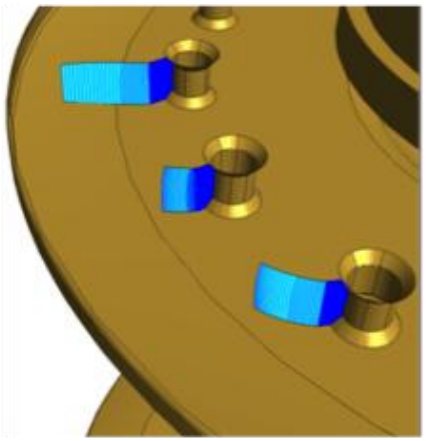
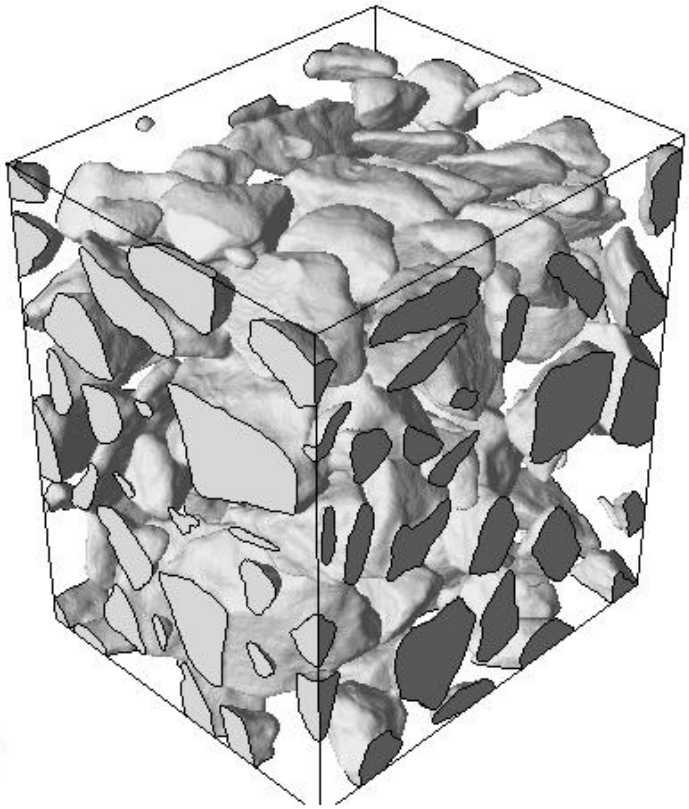
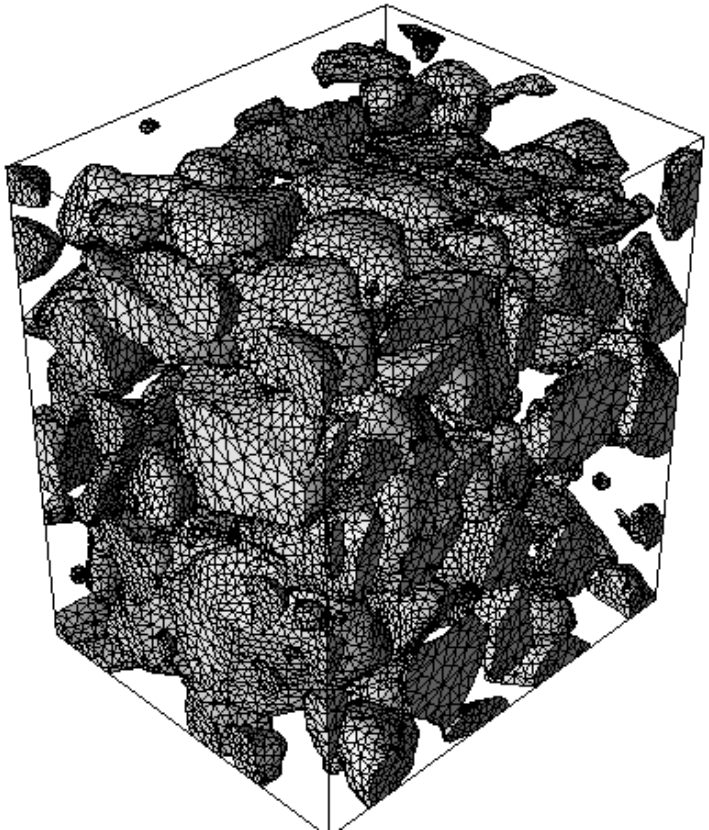


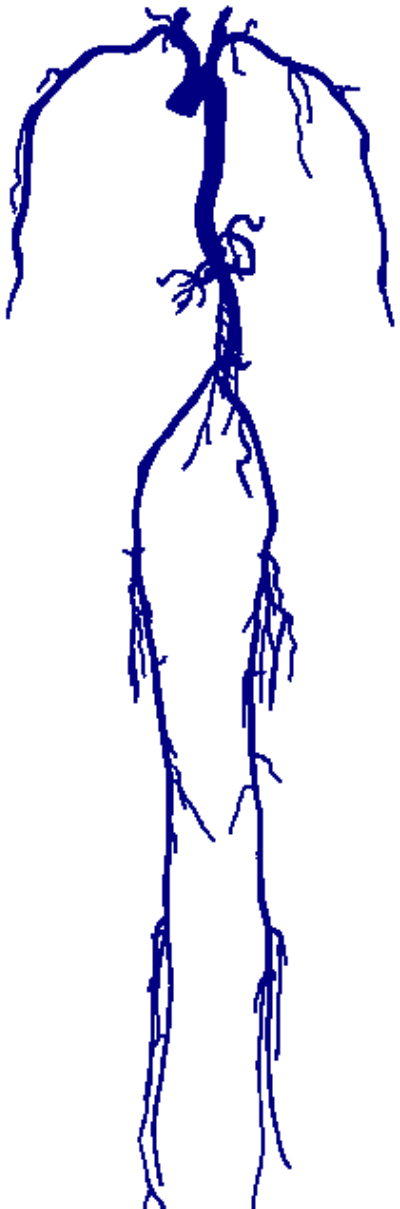
Image Data to Boundary Representation

Image data defined in terms off voxel data

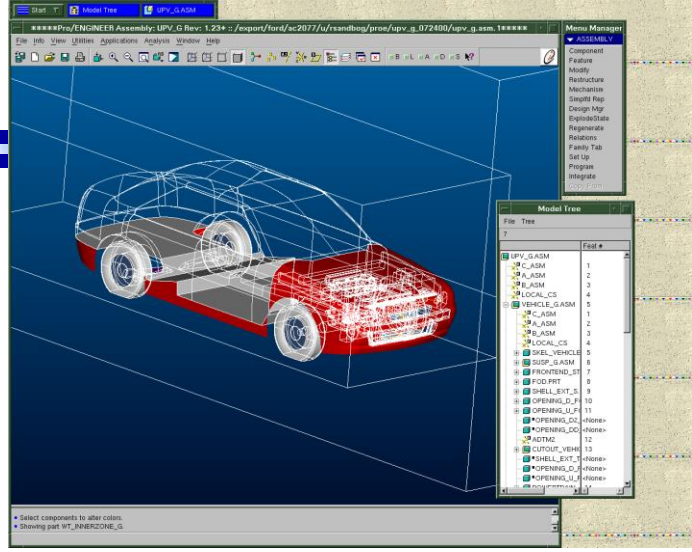
- Can mesh directly
- Constructing model topology provides better model control







Field Representations

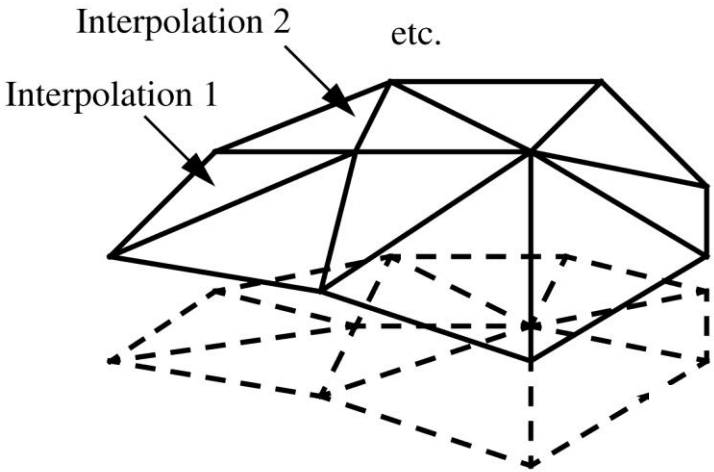


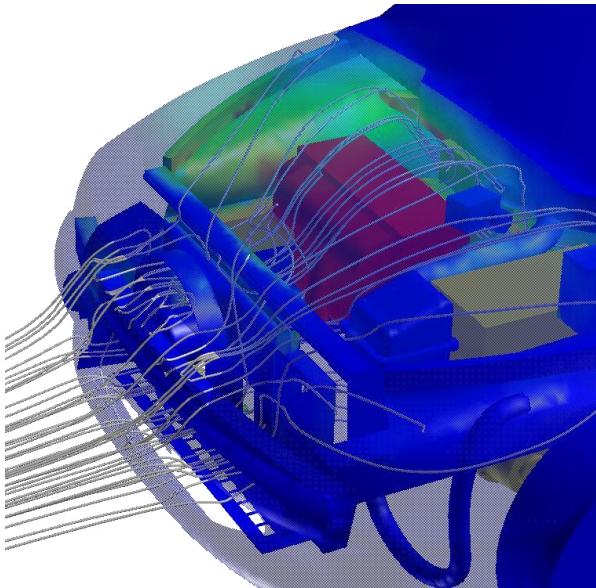
Explicit Tensor Fields

- Define the parameters in the mathematical equations in terms of tensors
- Defined in terms of distributions over entities in the boundary representation

Computational Tensor Fields

- Defined in terms of distributions over mesh entities of regions faces, edges and vertices





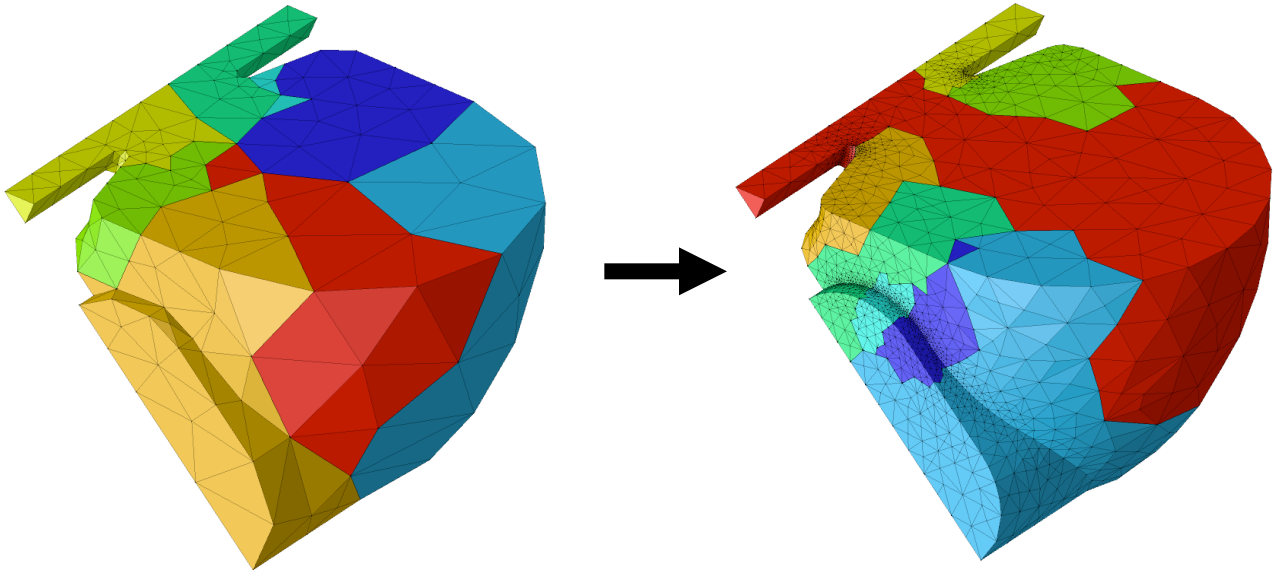
Parallelization of Components

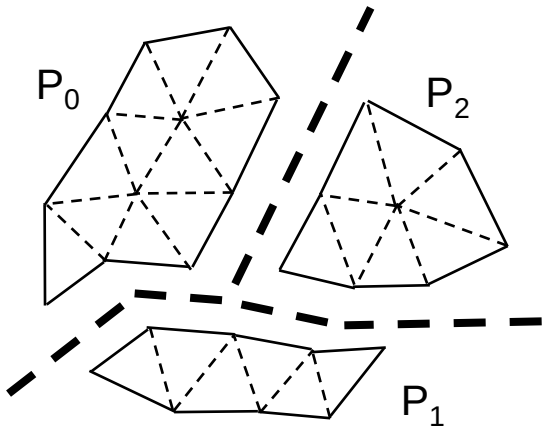
Parallel versions of multiple structures likely

Maximizing the use of one structure that most effectively supports the parallel execution is critical

Use partitioning of mesh into a set of parts that are

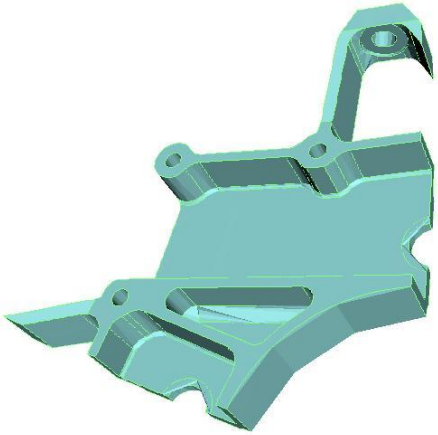
- Capable of providing mesh adjacencies across parts boundaries
- Ability to migrate selected mesh entities between parts
- Support fast dynamic load balancing



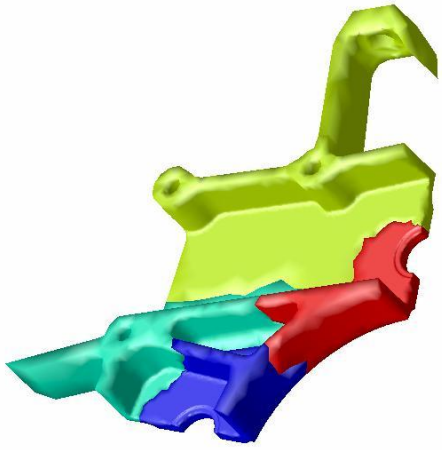


Partition Model for the Distributed Mesh

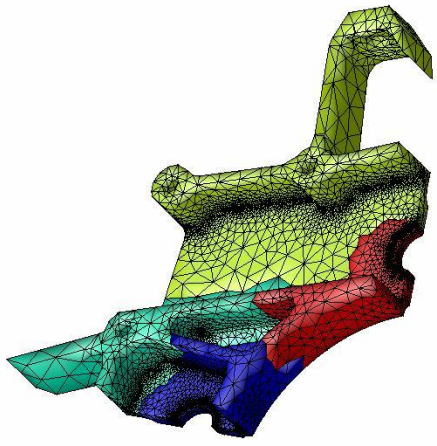
- On part mesh treated like a serial mesh with boundaries
- Some of the entities are on part boundary
- Partition model tracks part boundaries
  - Represent mesh partitioning in topology
  - Support mesh-level inter-partition communications
- An intermediary model located between the geometric model and mesh
- Partition model entities representing the collection of mesh entities that lie on part boundaries are tracked



Geometric model



Partition model

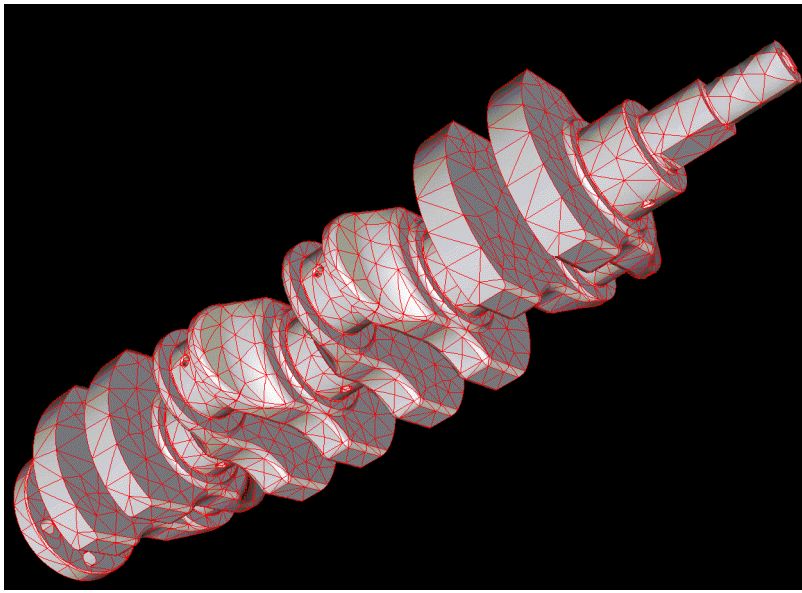


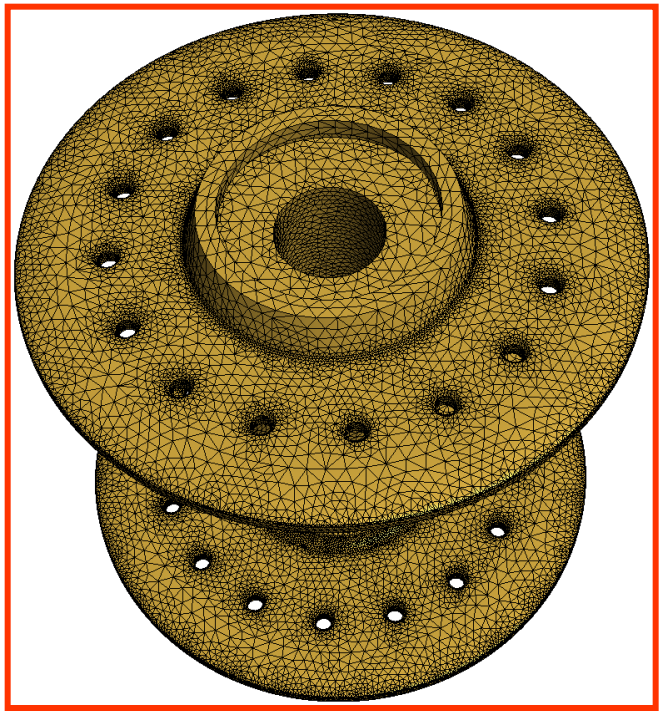
Partitioned mesh

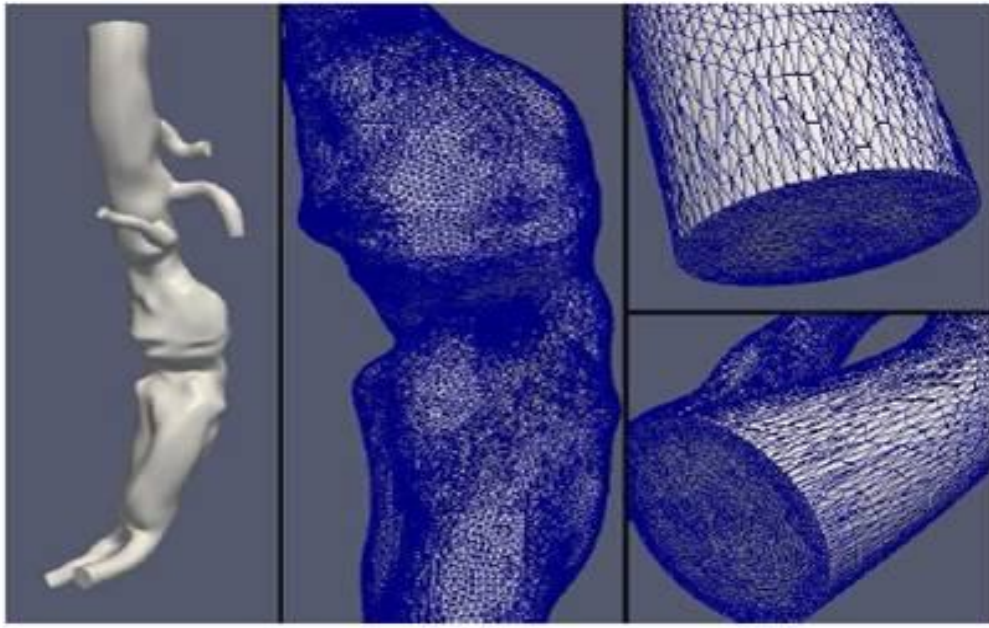
Automatic Mesh Generation from CAD

Key issues

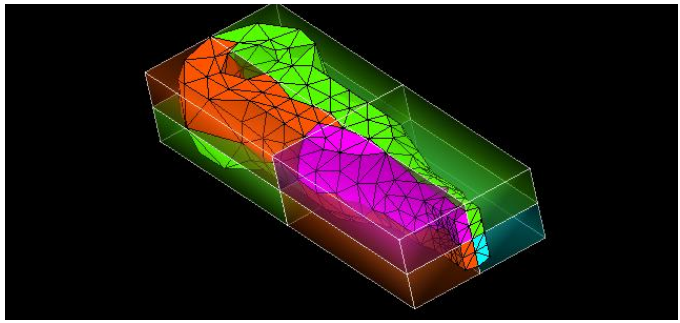
- Mesh representation
- Supporting needed relationship to the geometry
- Ensuring that a valid mesh of the model of interest is generated







Parallel Automatic Mesh Generation



Parallel mesh generation critical for initial meshes with billions of elements

Consider parallel mesh generation

- Computation effort related # of elements, but boundary elements have variable load
- Only structure known at start is the geometric model

All mesh generation steps operate in parallel

Meshes starting from solid model and all steps must be parallel

Structures created by the mesh generator are distributed

- Octree - used for mesh control, localizing searches, interior templates
- Mesh - topological hierarchy distributed

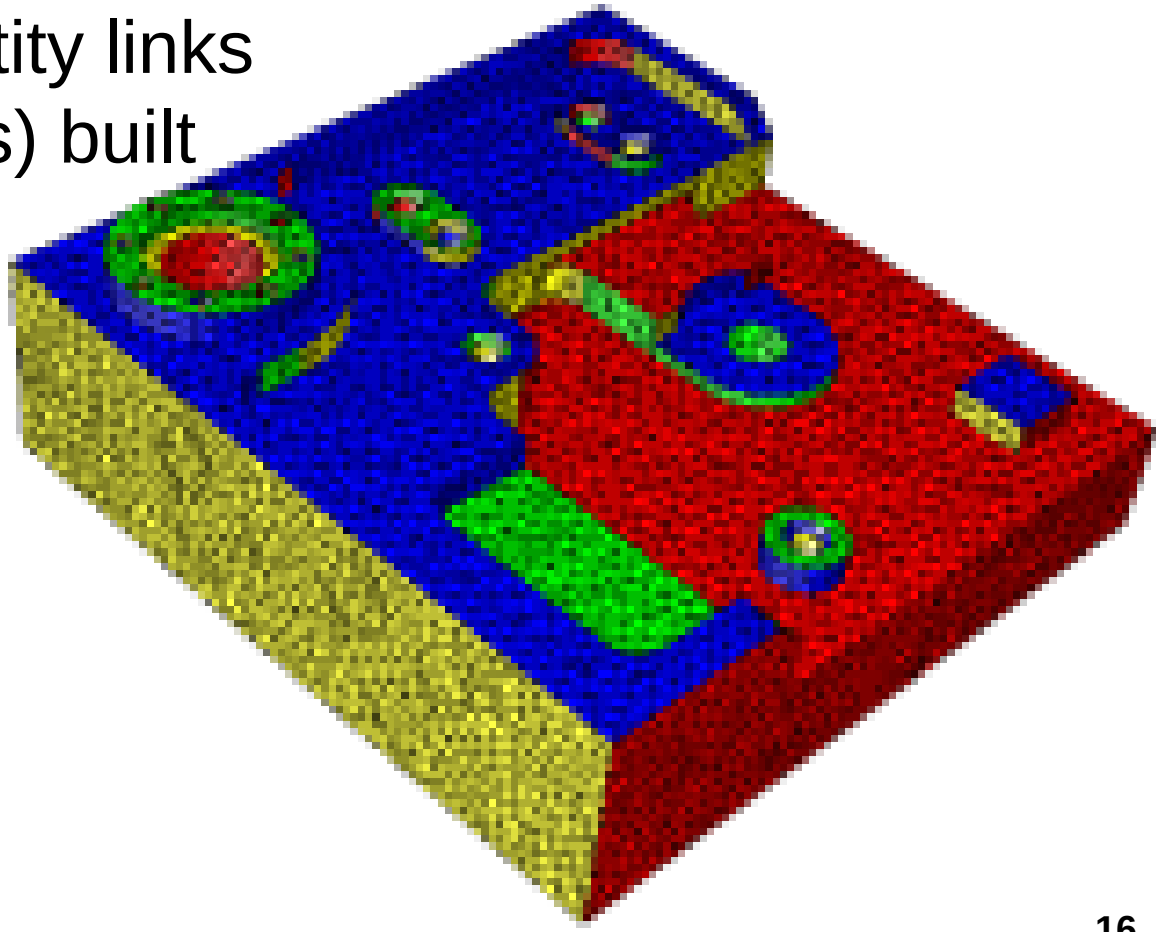
Mesh generation steps

- Surface mesh generation
- Octree refinement
- Template meshing of interior octants
- Meshing boundary octants

Parallel Surface Mesh Generation

Key features

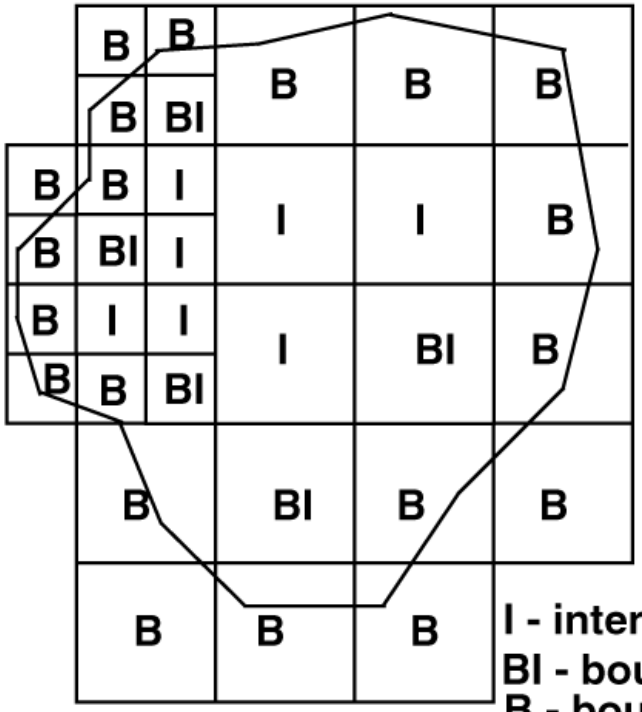
- Surfaces distributed to processors for meshing
- Faces can be split if needed to ensure scalability
- Boundary mesh entity links (edges and vertices) built



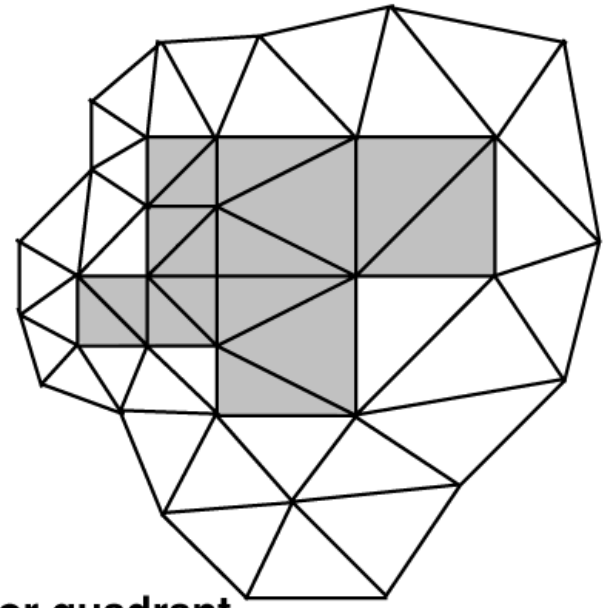
Parallel Volume Meshing

Given a distributed surface mesh, steps include:

- Build distributed tree (may be partly done)
- Classify octants
- Template meshing of interior octants
- Partition boundary octants
- Mesh on processor boundary regions
- Repartition to mesh partition boundaries
  - face
  - edge
  - vertex
- Repartition for next operation (an analysis step)



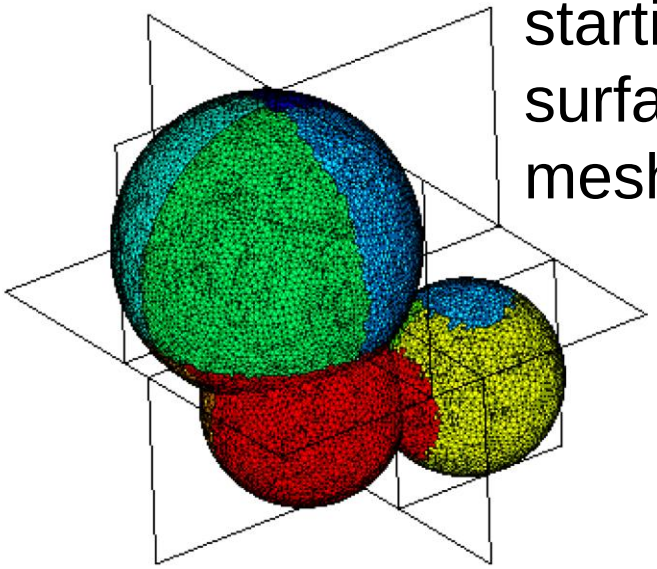
Quadtree with boundary edges



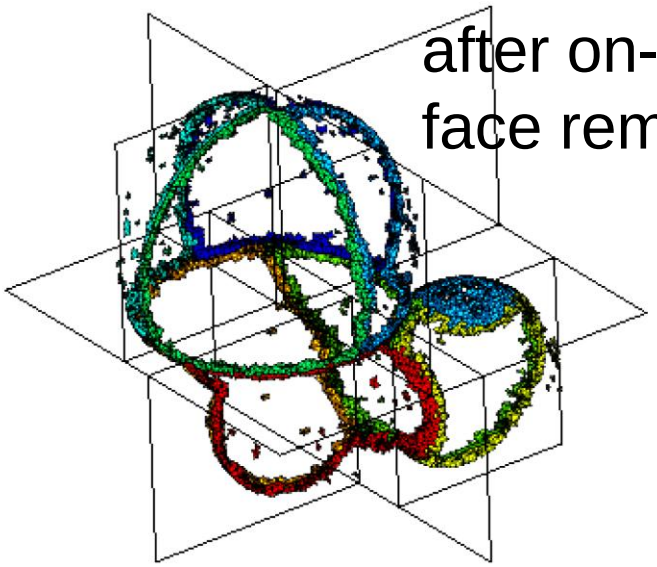
I - interior quadrant
BI - boundary like interior quadrant
B - boundary quadrant

Unsmoothed mesh

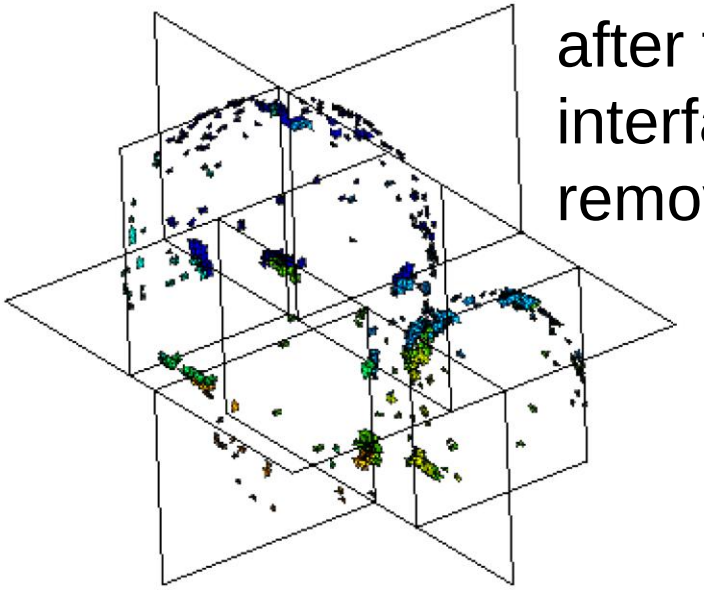
Boundary Octant Meshing Steps



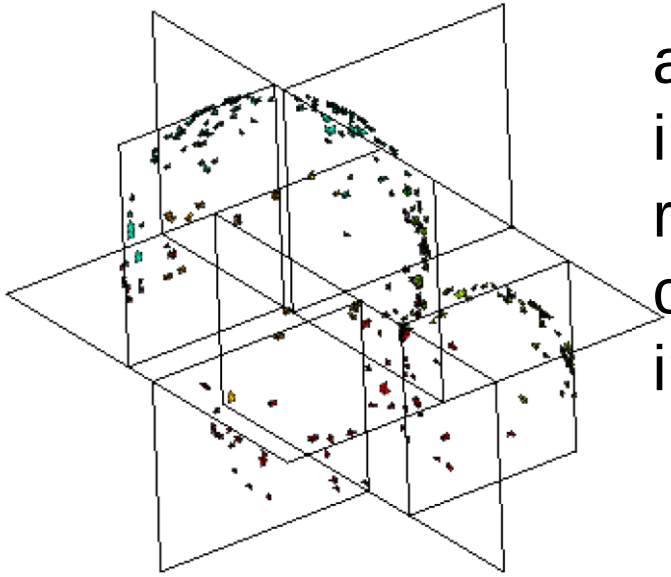
starting surface mesh



after on-processor face removals

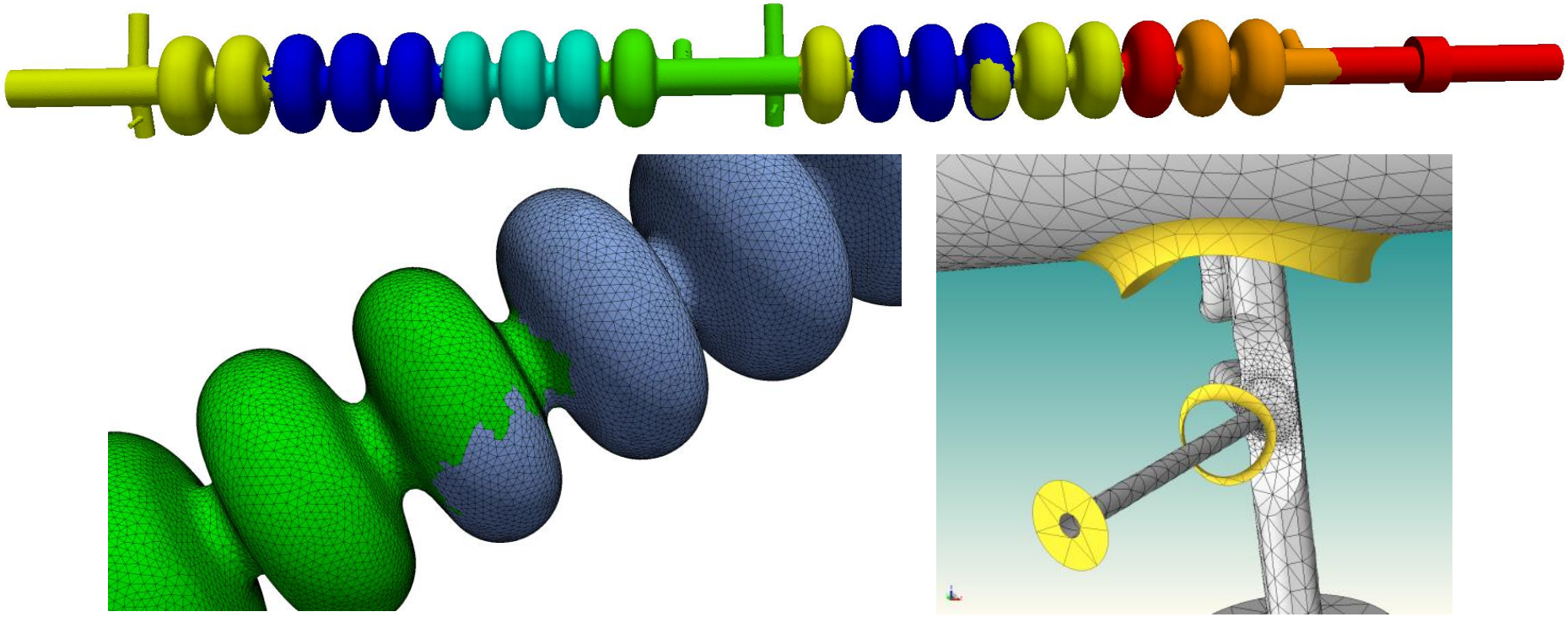


after face interface removals



after edge interface removals - completed in next step

Parallel meshing example



Overview and close-ups of mesh generated on 8 cores

- 180 million element mesh generated in 8 minutes on 64 cores
- Took more time to write files than generate mesh

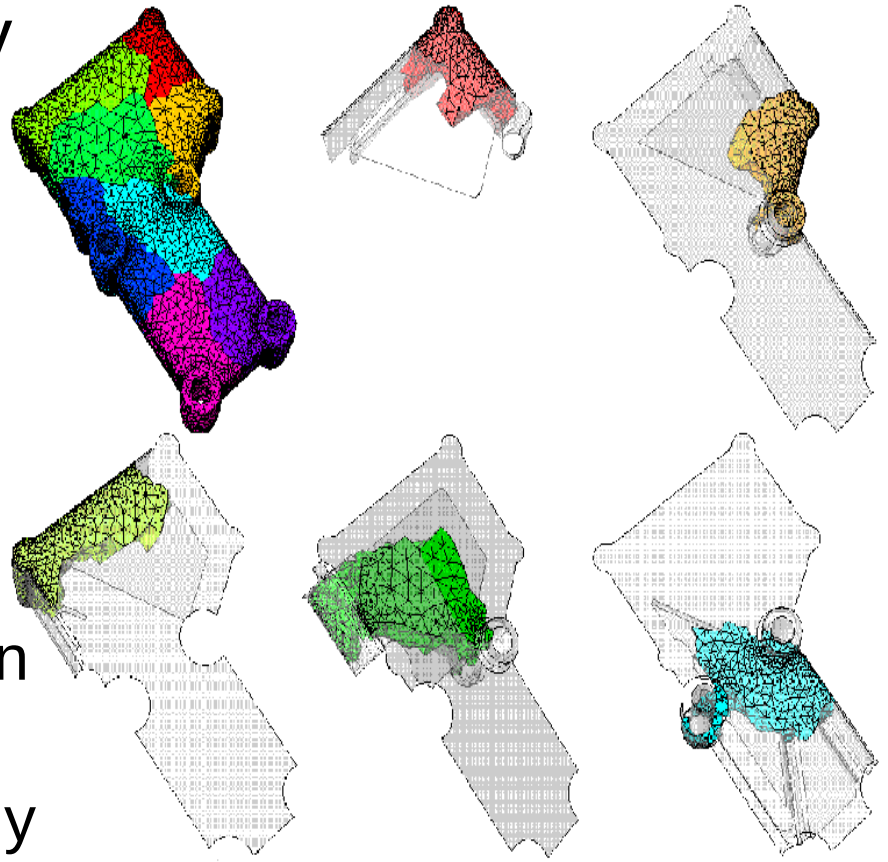
Distributed Parallel Geometry

Adaptive simulations on very large number of processors

- Keeping geometry on each processor is a memory issue

Partitioned model based on mesh partitioning

- Similar to partitioned mesh
- Migrate model entities between processors
- Properly hook up local topology on migration
- Ties into mesh migration so model migration is automatic
- Substantial memory savings

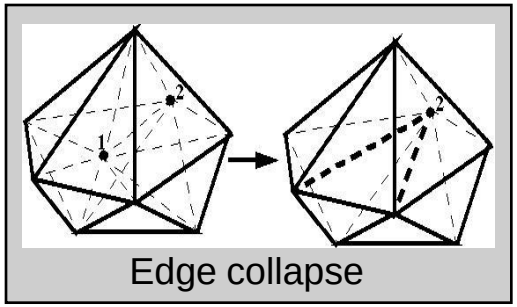


Model entities on part based on classification of mesh entities on part

Mesh Adaptation by Mesh Modification

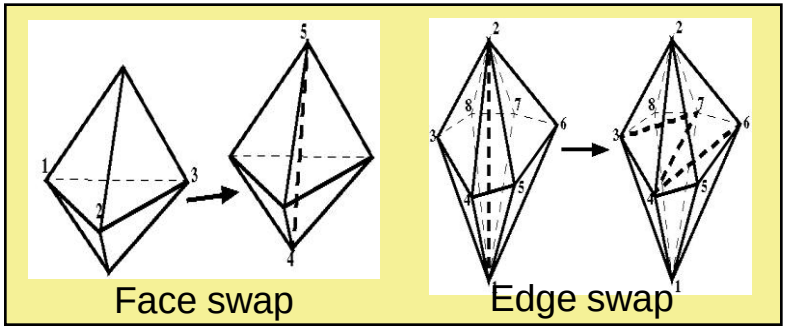
Given the “mesh size field”:

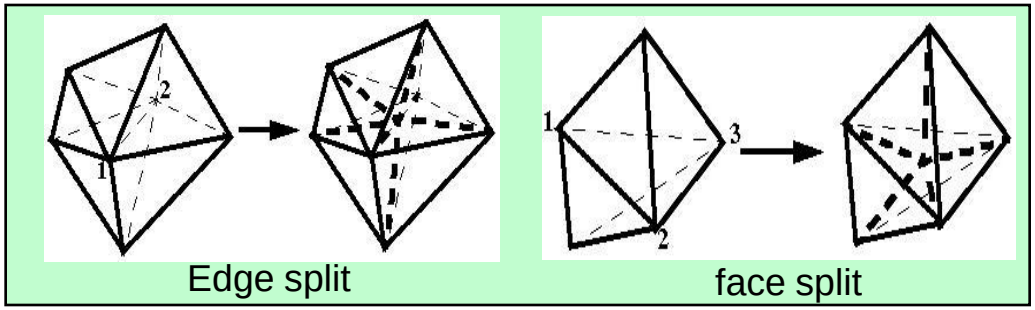
- Look at edge lengths and shape
  - If both satisfactory continue to next element
  - If not satisfied select “best” modification
- Continue until size and shape is satisfied or no more improvement possible

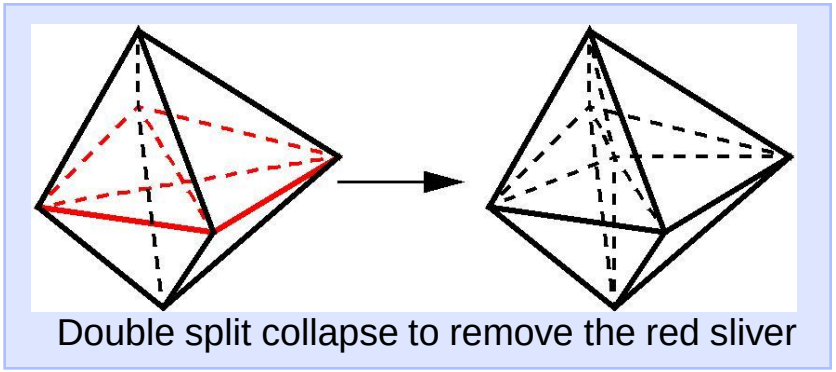


Determination of “best” mesh modification

- Selection of mesh modifications based on satisfaction of the element requirements
- Appropriate consideration of neighboring elements
- Choosing the “best” mesh modification

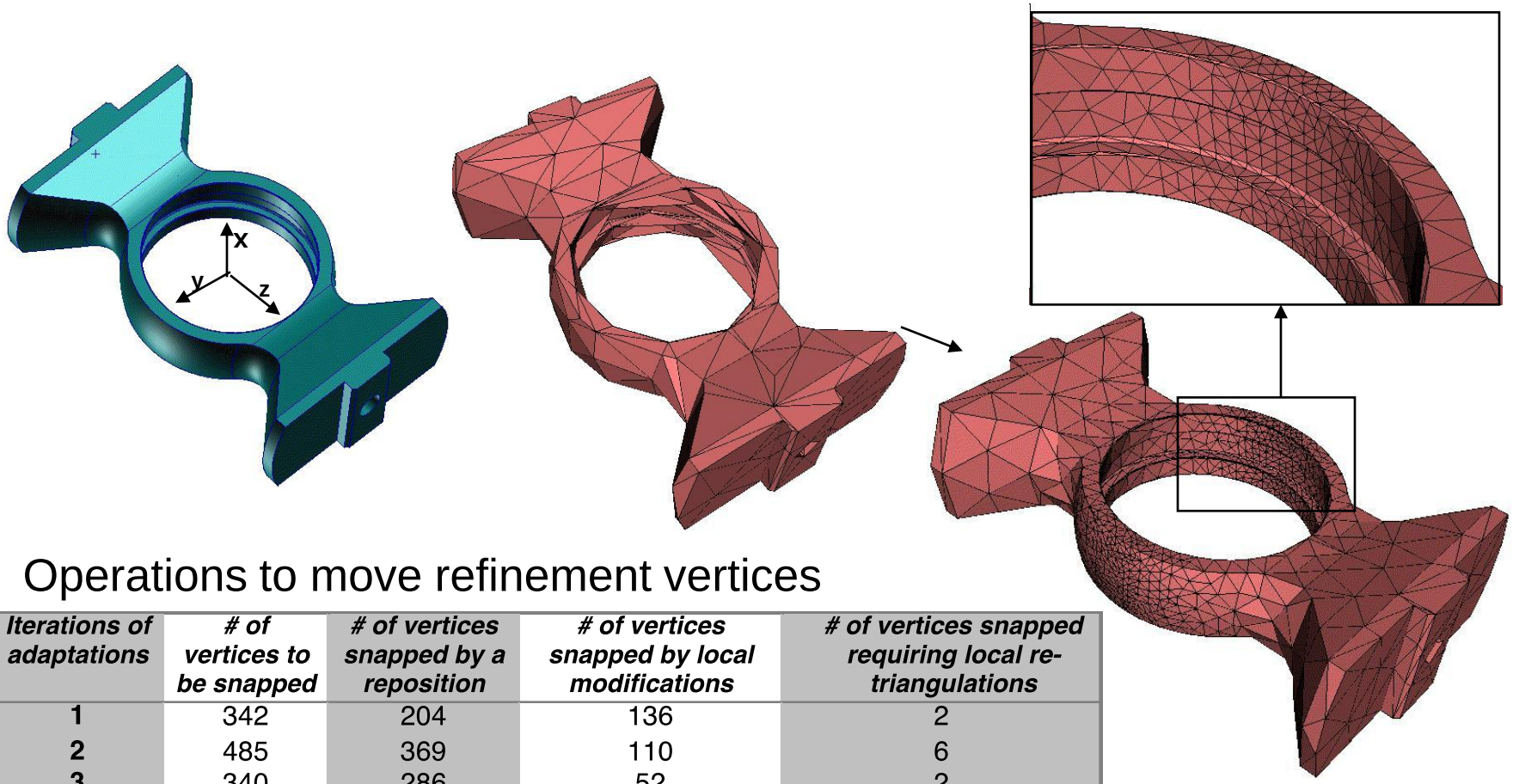






Accounting for Curved Domains During Refinement

- Moving refinement vertices to boundary required mesh modification (see IJNME paper, vol58 pp247-276, 2003)
- Coarse initial mesh and the mesh after multiple refinement/coarsening



- Operations to move refinement vertices

Iterations of adaptations	*# of vertices to be snapped*	*# of vertices snapped by a reposition*	*# of vertices snapped by local modifications*	*# of vertices snapped requiring local re-triangulations*
1	342	204	136	2
2	485	369	110	6
3	340	286	52	2
4	74	34	40	-
5	26	3	23	-

Mesh adaptation to an Anisotropic Mesh Size Field

Based on applying mesh modifications following mesh metric

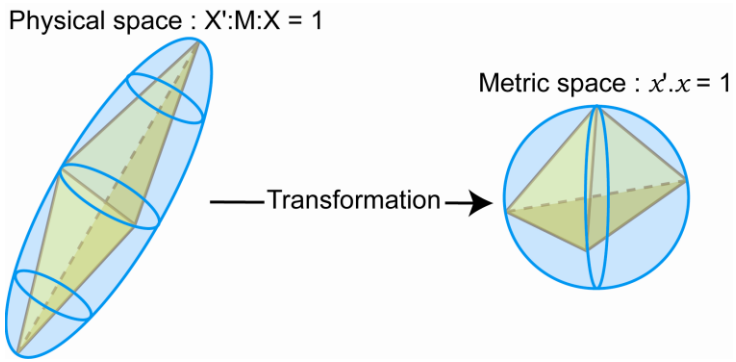
- Transformation matrix field T(x,y,z)

$$T(x,y,z) = \underbrace{\begin{bmatrix} 1/h_1 & 0 & 0 \\ 0 & 1/h_2 & 0 \\ 0 & 0 & 1/h_3 \end{bmatrix}}_{\text{Distortion}} \cdot \underbrace{\begin{bmatrix} \vec{e}_1 \\ \vec{e}_2 \\ \vec{e}_3 \end{bmatrix}}_{\text{Rotation}}$$

$\vec{e}_1, \vec{e}_2, \vec{e}_3$: **Unit vectors associated with three principle directions**

h_1, h_2, h_3 : **Desired mesh edge lengths in these directions**

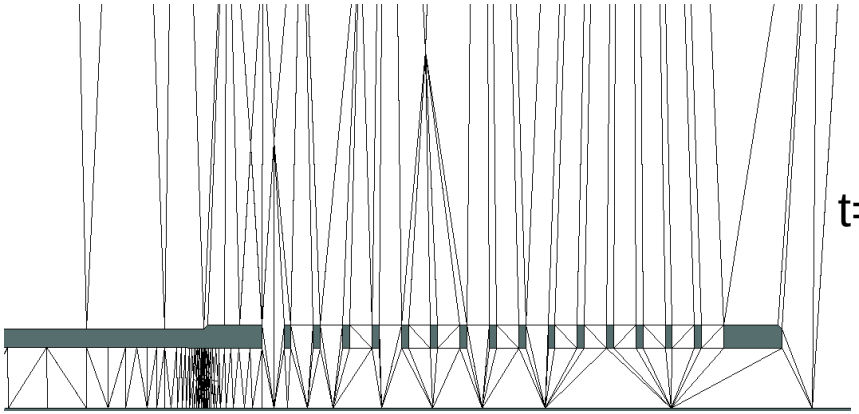
- Ellipsoidal in physical space transformed to normalized sphere



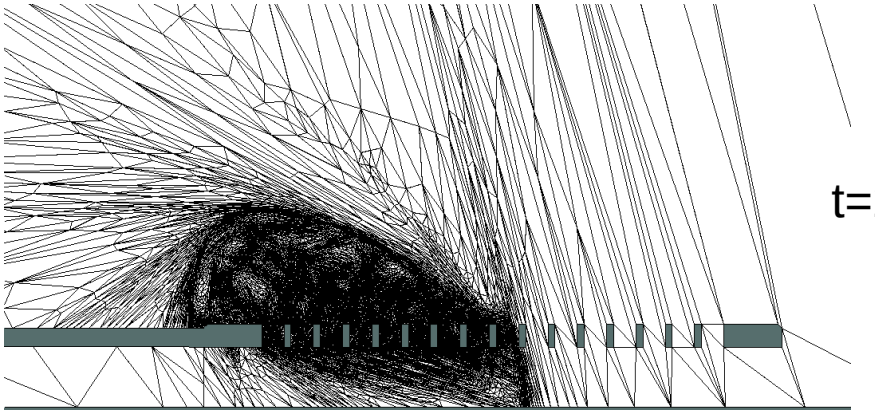
- Volume relation between physical space and the transformed space:

$$V_{transformed} = |T(x,y,z)| \times V_{physical}$$

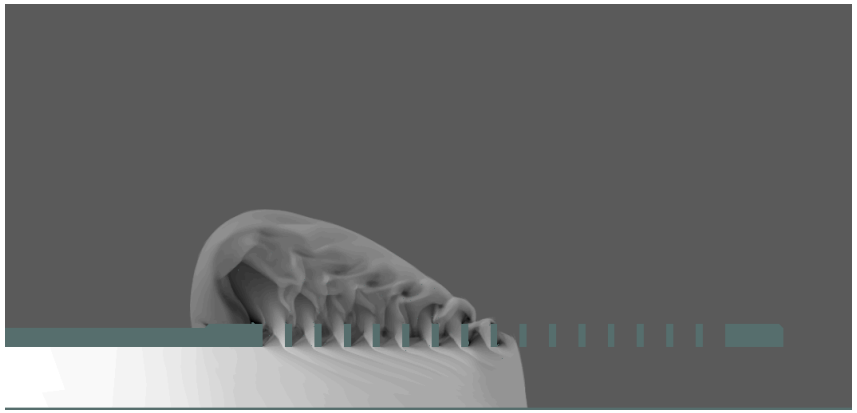
Muzzle Blast

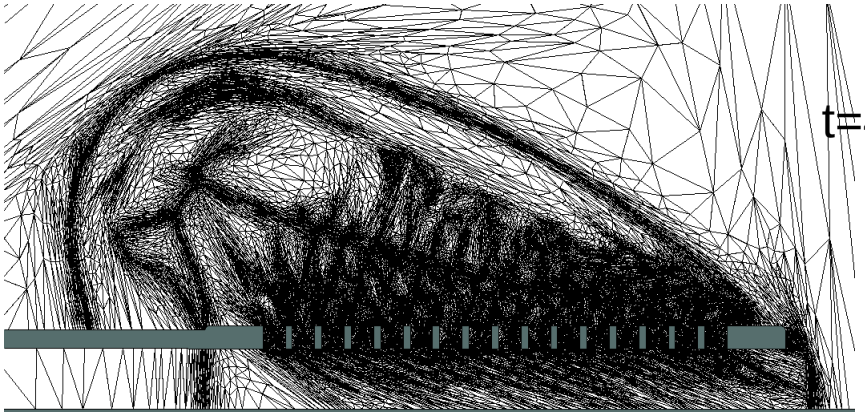


t=0.0

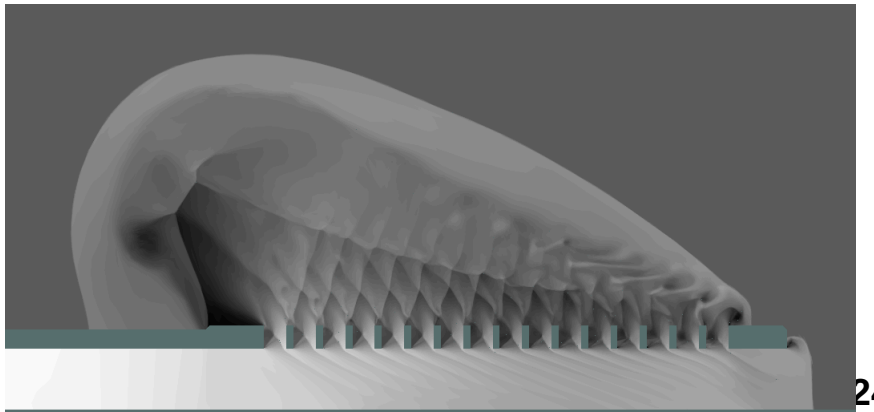


t=2e-4





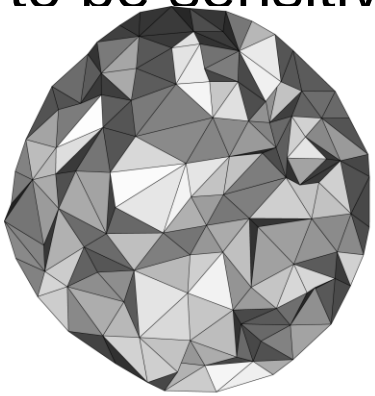
t=4e-4

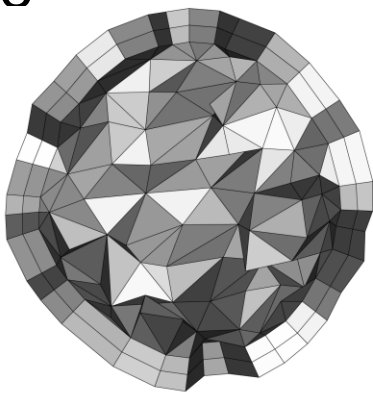


Maintaining “Structure” for Derivative Recovery

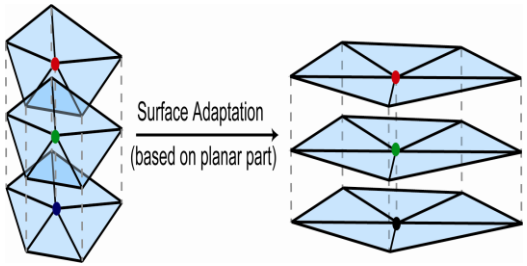
Post-processing procedure for recovering conservative wall shear stress has been observed to be sensitive to near wall mesh “structure”.

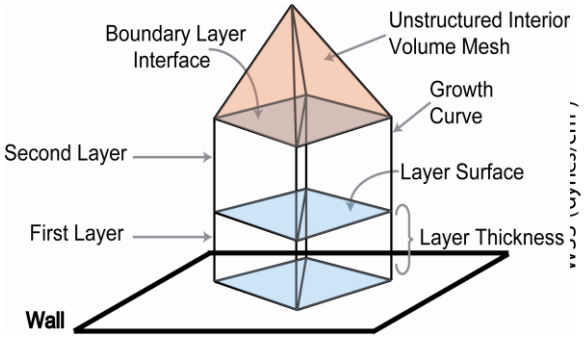
- Coarse example of arterial cross section

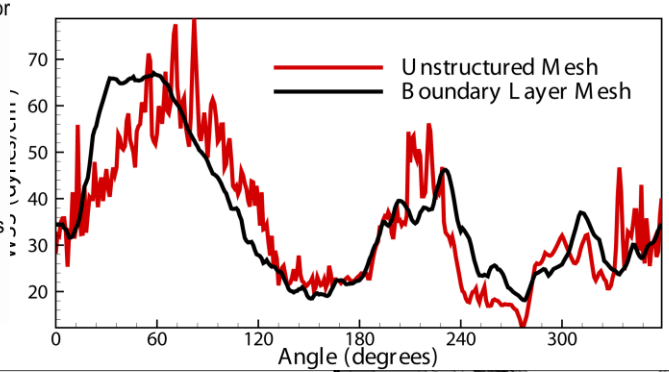


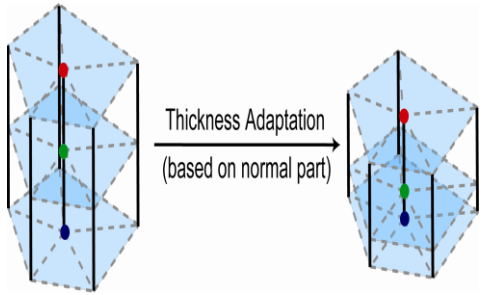


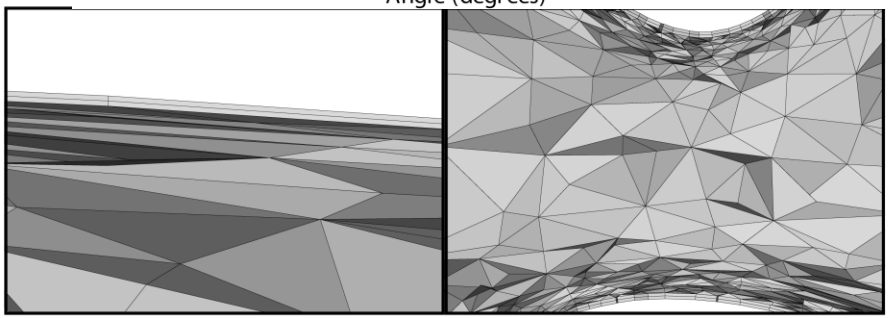
Semi-structured meshes mesh adaptation added





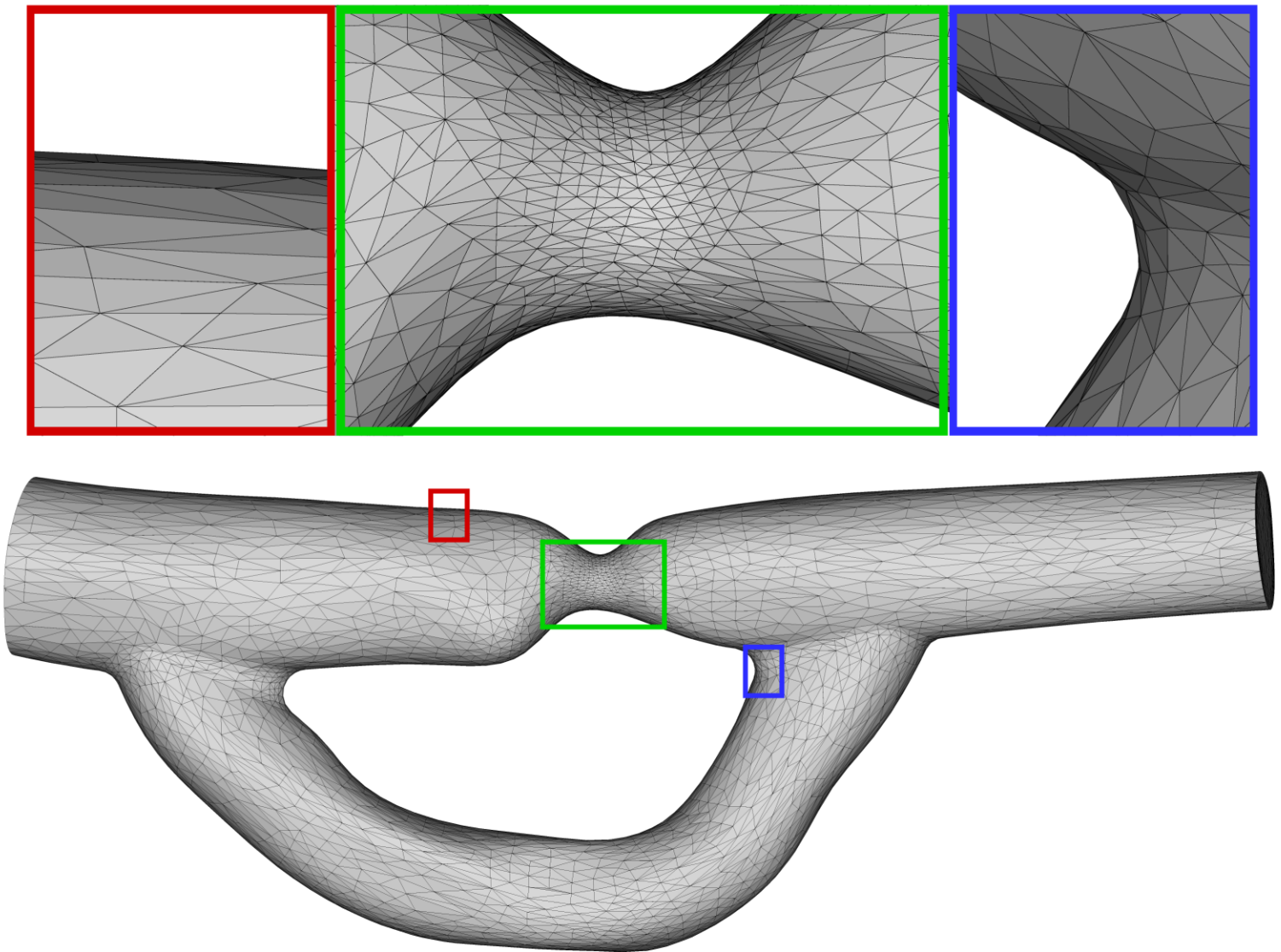






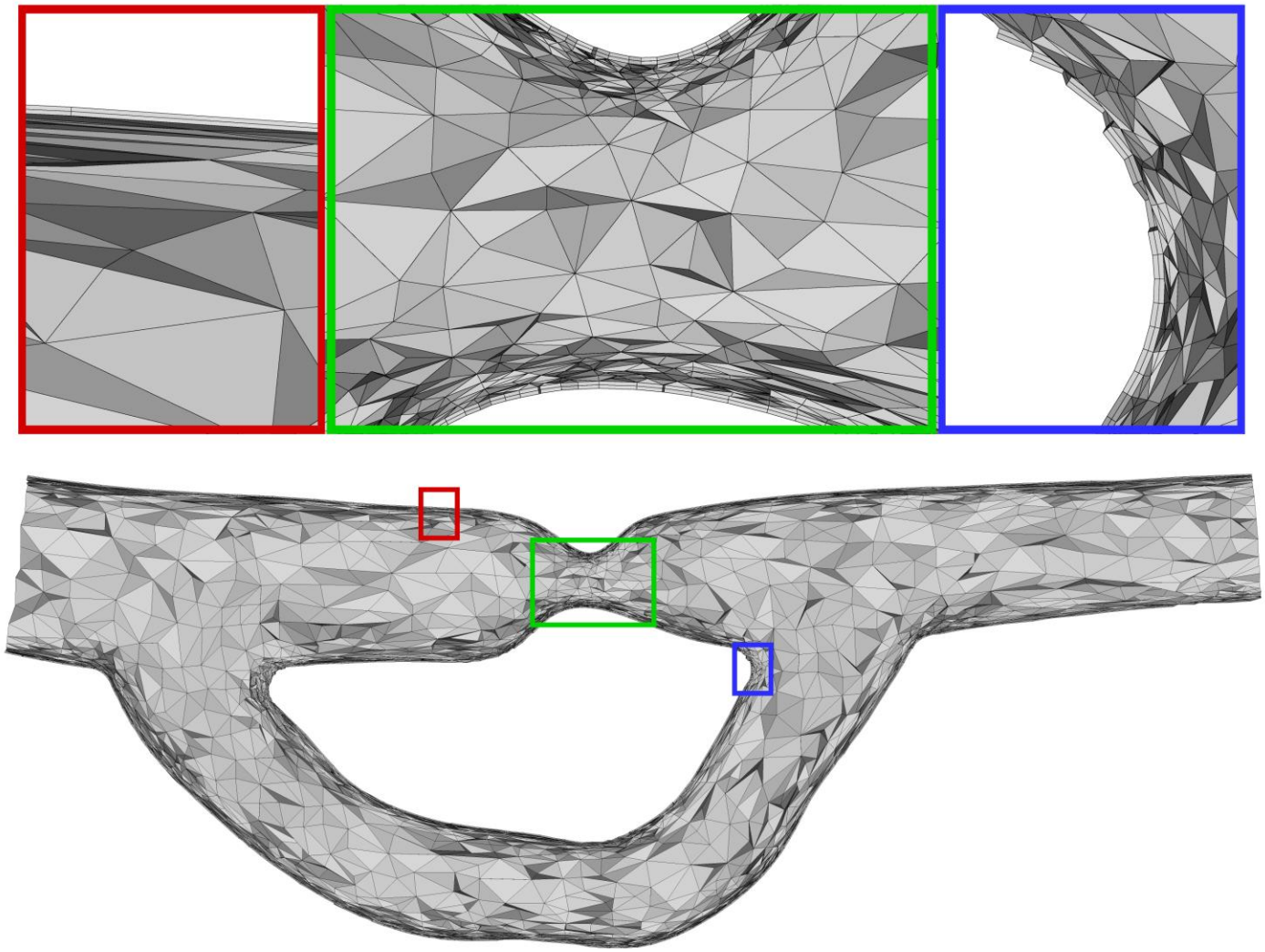
Surface Anisotropy

Surface of adapted mesh for porcine aorta



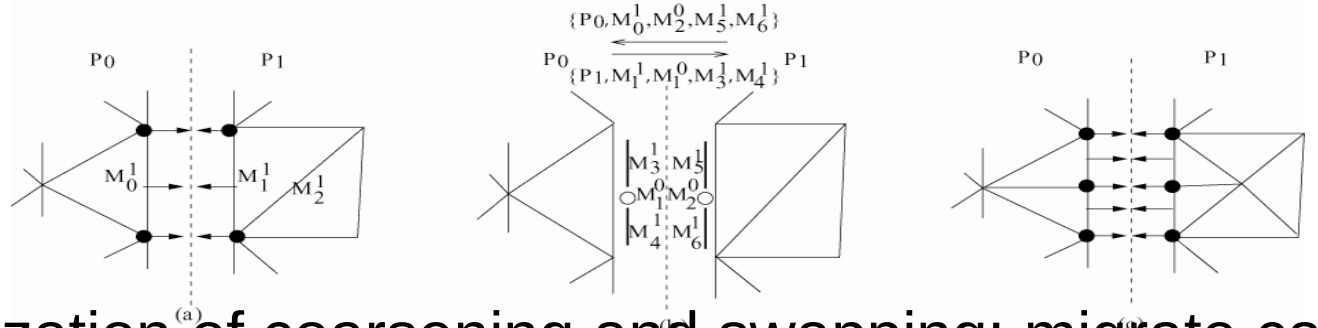
Interior Anisotropy

Clip plane of adapted mesh for porcine aorta

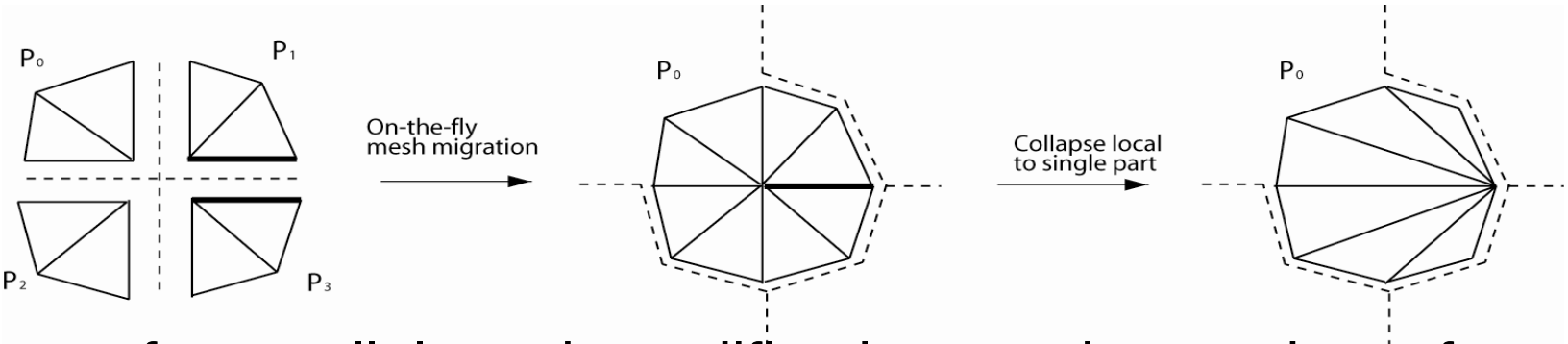


Parallel Mesh Adaptation

Parallelization of refinement: perform on each part and synchronize at inter-part boundaries.



Parallelization of coarsening and swapping: migrate cavity (on-the-fly) and perform operation locally on one part.



Support for parallel mesh modification requires update of evolving communication-links between parts and dynamic mesh partitioning.

Mesh Migration

What needs to be migrated

- Dictated by operation - in swap and collapse it's the mesh entities on other parts needed to complete the mesh modification cavity
- Information determined based on mesh adjacencies

Complexity of mesh migration a function of mesh representation

- Complete representation can provide any adjacency without mesh traversal - a requirement for satisfactory efficiency
- Both full and reduced representations can be complete
  - Full representation - all mesh entities explicitly represented
  - Reduced representation - requires all mesh vertices and mesh entities of the same order as the model (or partition) entity they are classified on

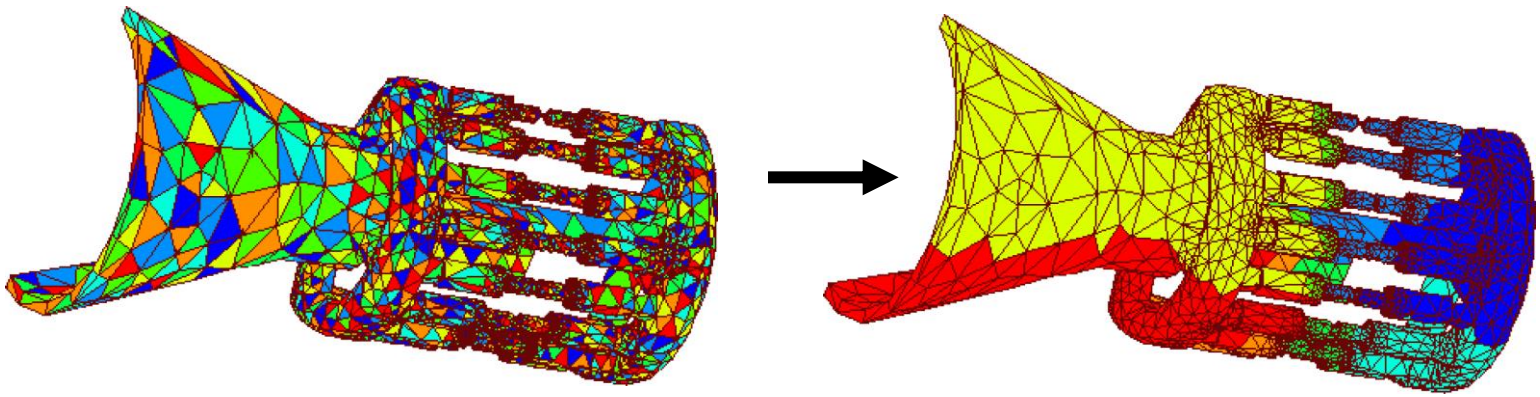
Dynamic Load Balancing

Zoltan Dynamic Services (http://www.cs.sandia.gov/Zoltan/)

- Supports multiple dynamic partitioners
- General control of the definition of part objects and weights supported
- Focused to graph-based (or hypergraph-based) partitioners

Dynamic load balancing directly using mesh adjacencies

- Direct use of the “super graph” of mesh adjacencies – account for multiple criteria at once
- To date developed for partition improvement only



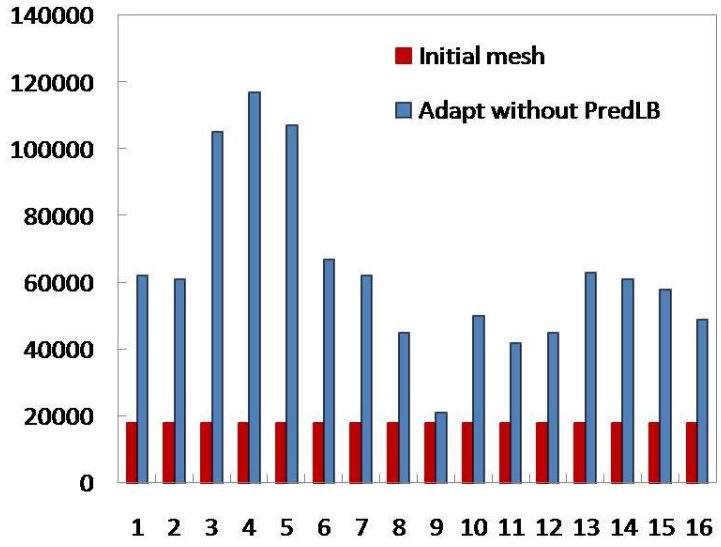
Predictive Load Balancing

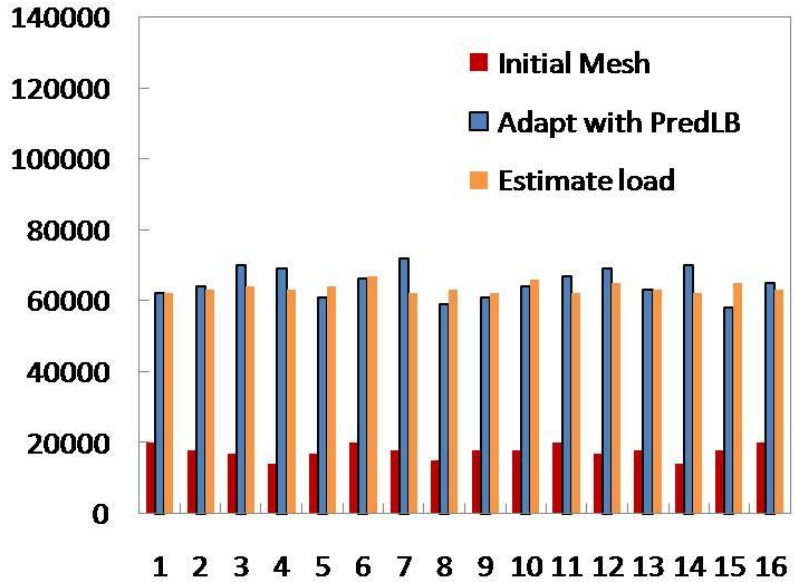
Mesh modification before load balancing can lead to memory problems - Employ predictive load balancing to avoid the problem

Algorithm employ weighted dynamic load balance

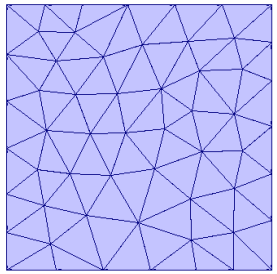
- Mesh metric field at any point P is decomposed to three unit direction (e_1,e_2,e_3) and desired length (h_1,h_2,h_3) in each corresponding direction.
- The volume of desired element (tetrahedron) : $h_1h_2h_3/6$
- Estimate number of elements to be generated:

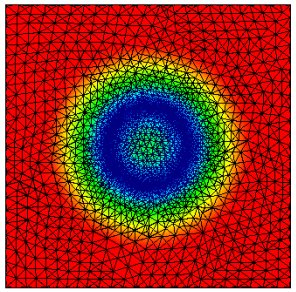
$$num = \frac{R\_volume}{\sum_{p=1}^{n_{en}} h_1(p)h_2(p)h_3(p)/6n_{en}}$$



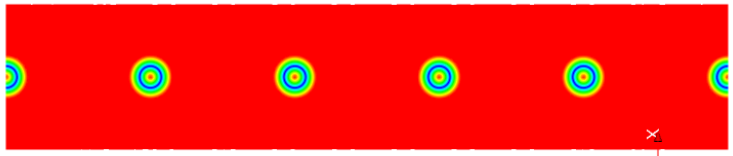


Predictive Load Balancing

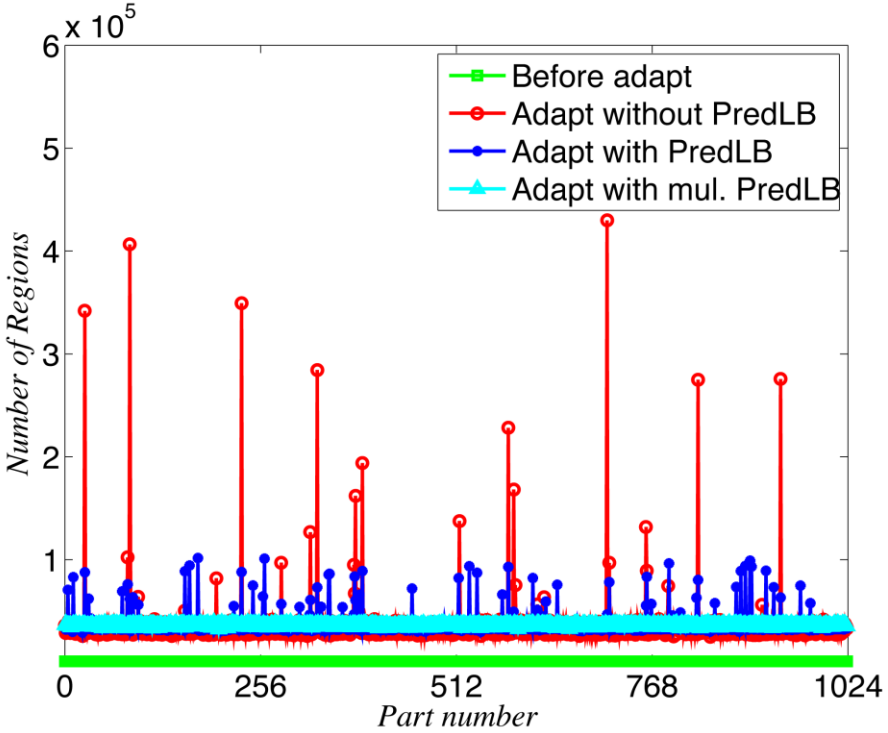




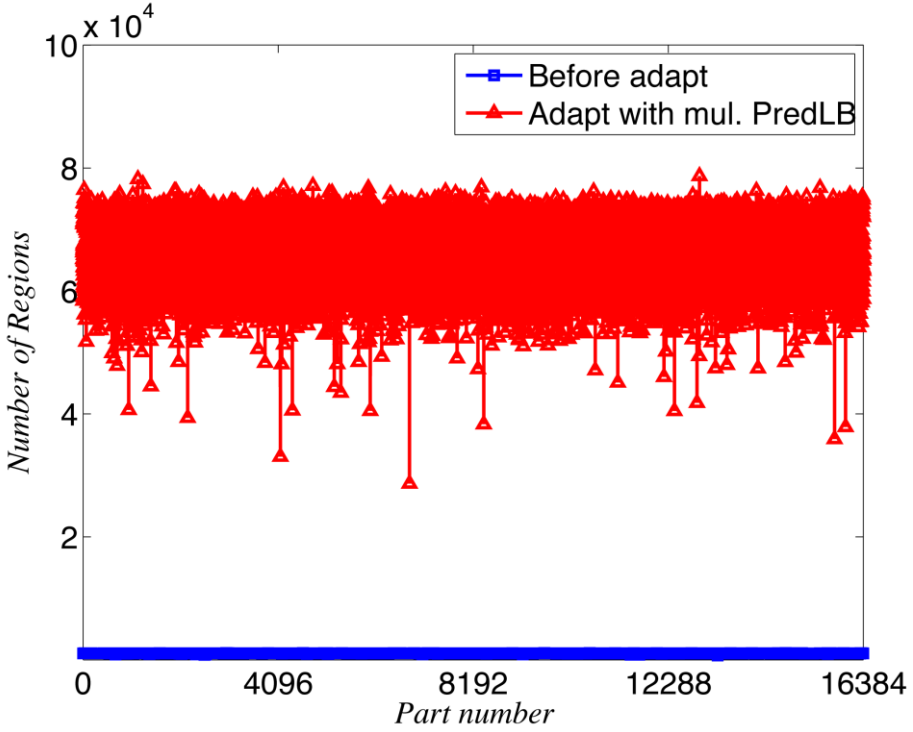
Initial and adapted mesh zoom for ½ bubble colored by mesh size field



The distribution of “air bubbles” in the tube (part of model)



Adapted: 36.6M, 5 “bubbles”



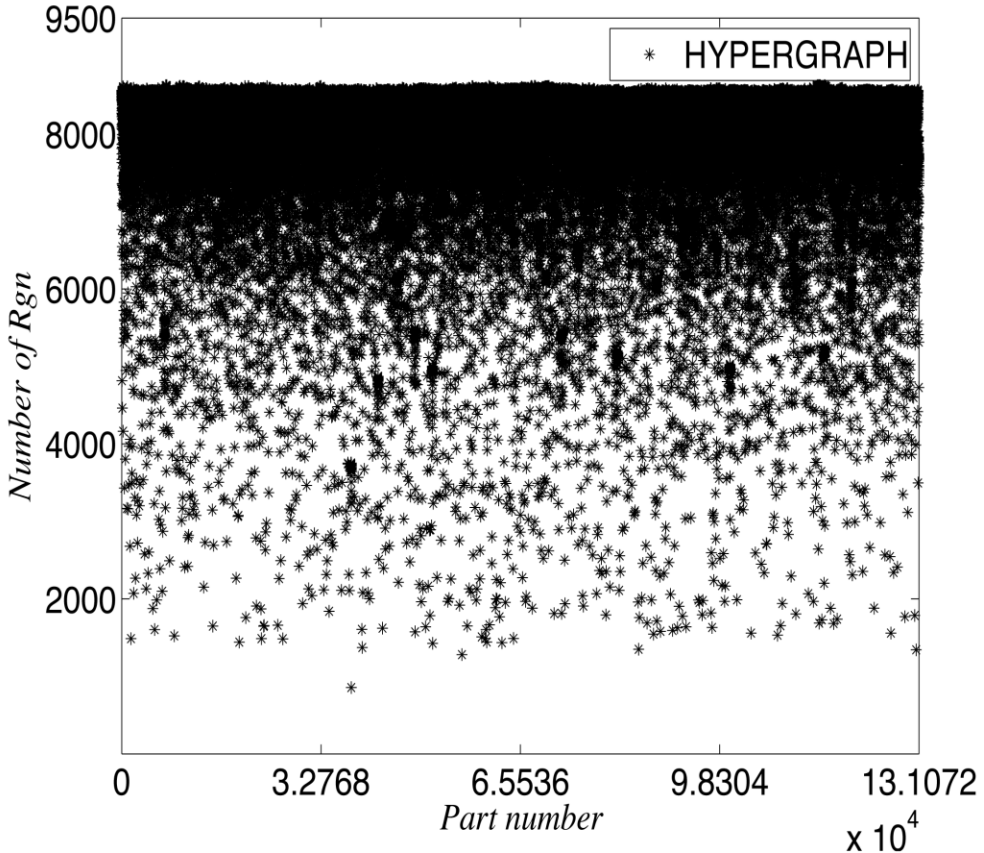
Multiple PredLB is used to make the adaptation possible on 16K BG/L cores

Multiple Parts per Process

Support more that one part per process

- Key to changing number of parts
- Also used to deal with problem with current graph-based partitioners that tend to fail on really large numbers of processors

- 1 Billion region mesh Starts well balanced mesh on 2048 parts
- Each part slits to 64 parts - get 128K parts
- Region imbalance: 1.0549
- Time usage < 3mins on Kraken
- This is the partition used for scaling test on Intrepid



Partition Improvement Procedures

Objective: Incremental redistribution of mesh entities to improve overall balance

Partition improvement based on mesh adjacencies (ParMA)

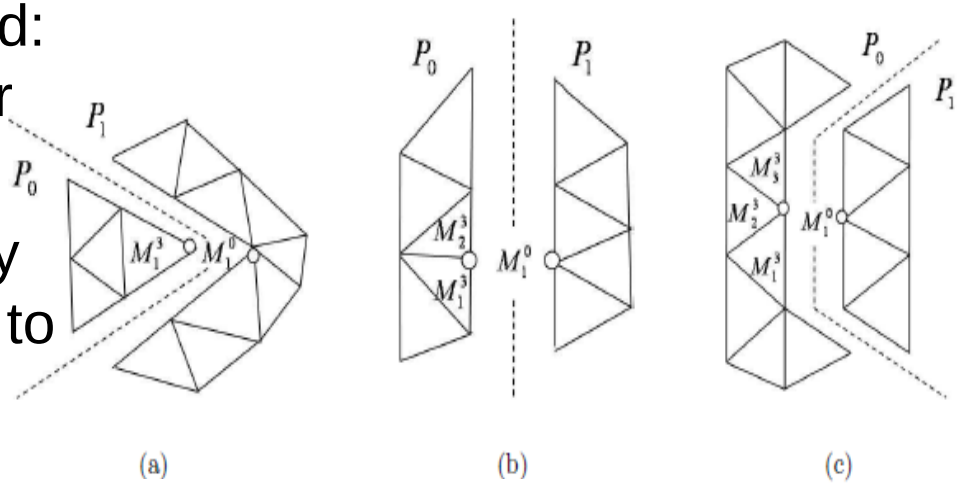
- Designed to improve balance for multiple entities types
- Use mesh adjacencies directly to determine best candidates for movement
- Current implementation based on neighborhood diffusion

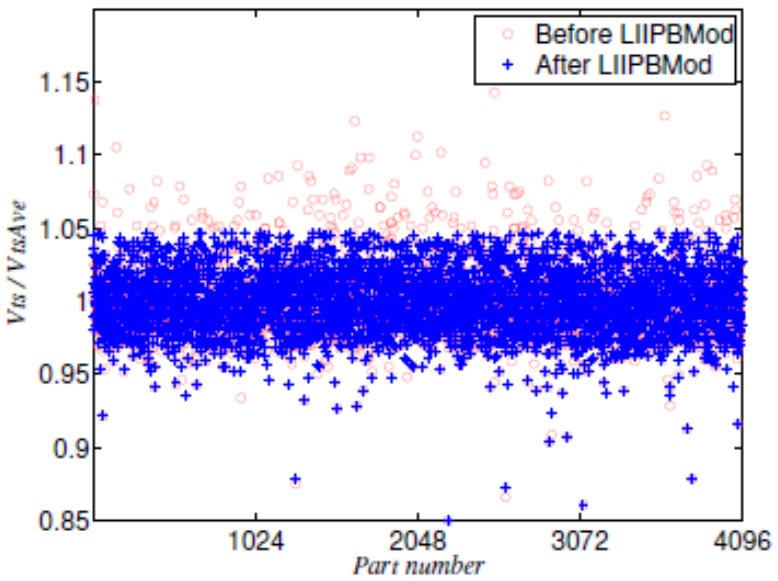
Table: Region and vertex imbalance for a 8.8 million region uniform mesh on a bifurcation pipe model partitioned to different number of parts

num of parts	128	256	512	1024	2048	4096	8192
Ave numRgns	68.856K	34.463K	17.231K	8.616K	4.308K	2.154K	1.077K
Ave numVts	14.02K	7.262K	3.780K	1.984K	1.050K	559.98	305.96
Region imbalance	1.022	1.019	1.021	1.014	1.024	1.021	1.029
Vertex imbalance	1.034	1.036	1.053	1.069	1.092	1.143	1.154

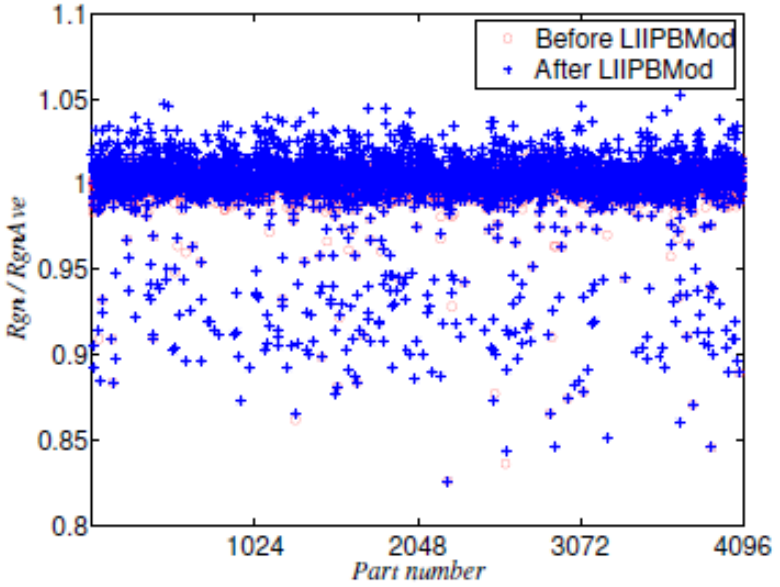
Partition Improvement Procedures

- Selection of vertices to be migrated: the ones bounding a small number of adjacent elements
- Vertices with only one remote copy considered to avoid the possibility to create nasty part boundaries

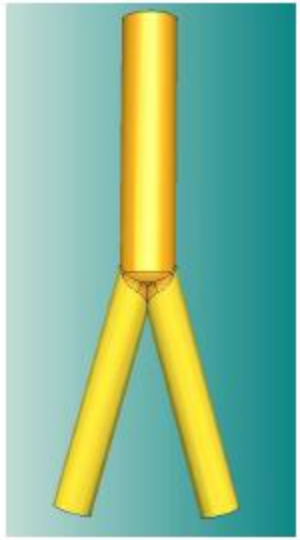




Vertex imbalance: from 14.3% to 5%



Region imbalance: from 2.1% to 5%



File Free Parallel-Adaptive Loop

Current adaptive loops use file transfer between components

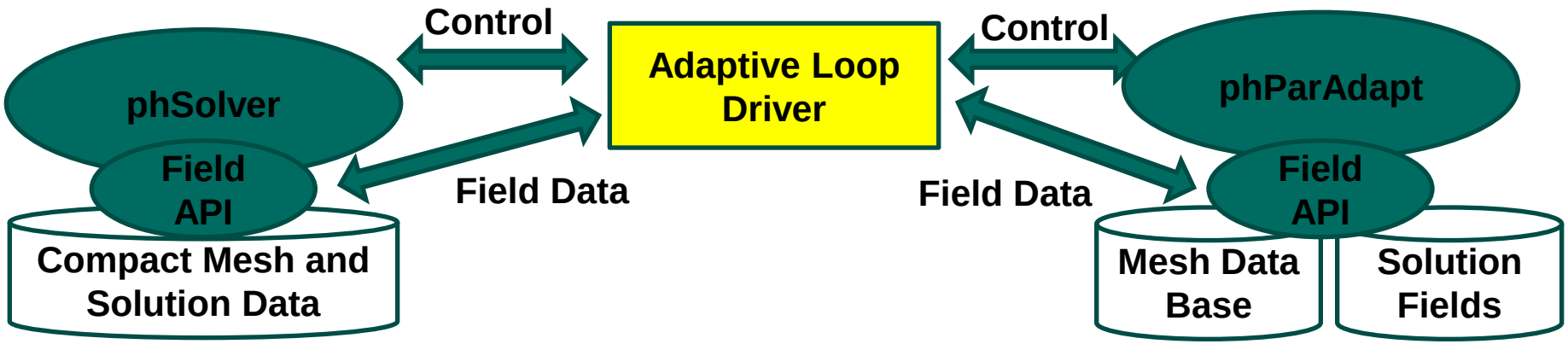
- ~ 1/3 of total time spent for file I/O on even 128 cores

Need to couple components through functional interfaces

Coupling through adaptive loop driver

- Abstract component complexities
- Inter language coupling (Fortan, C, C++, Python, etc.)
- Change/Add components with minimal development costs

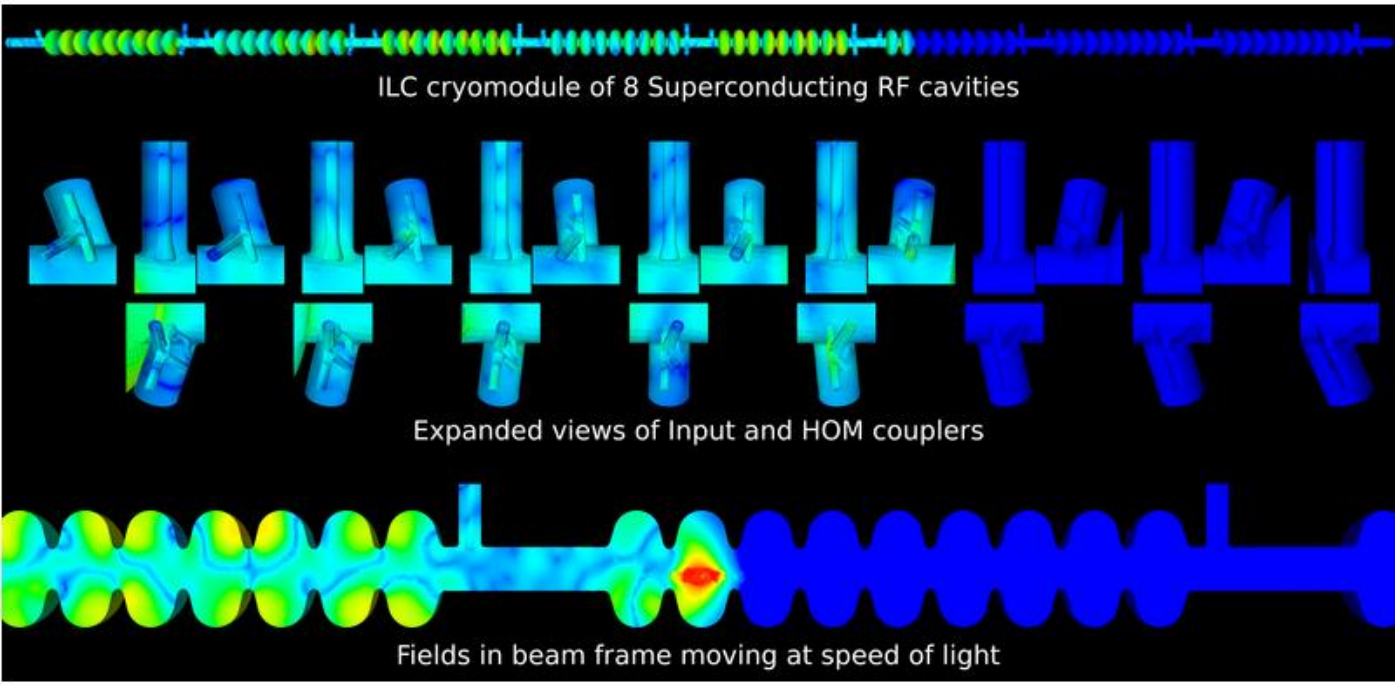
This is in initial development

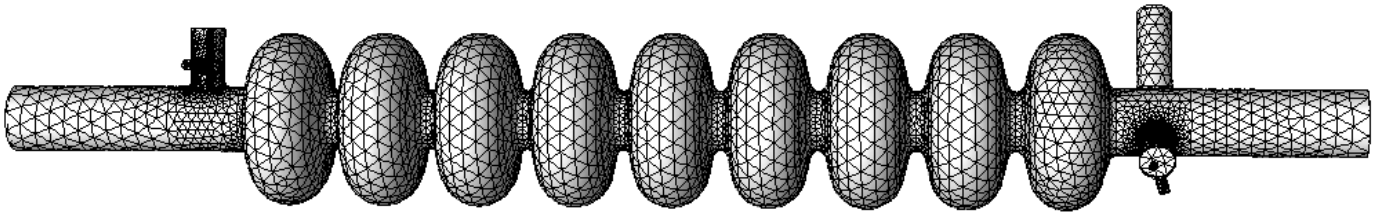


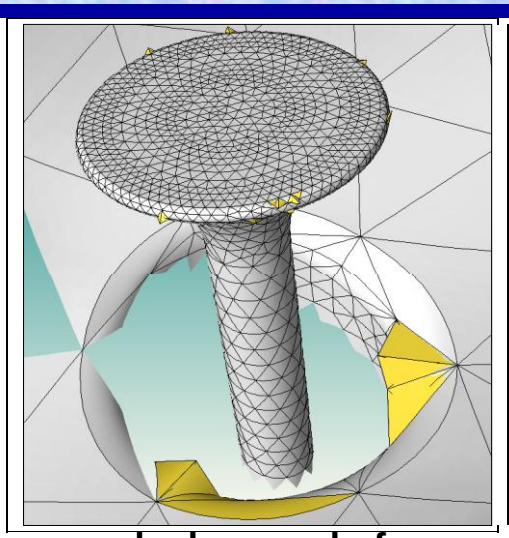
Mesh Generation & Adaptation for Curved Elements

Mesh curving applied to 8-cavity cryomodule simulations

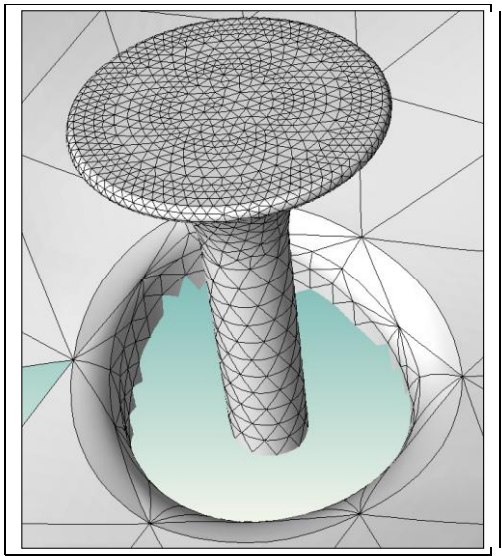
- 2.97 Million curved regions
- 1,583 invalid elements corrected – leads to stable simulation and executes 30% faster







mesh close-up before and after correcting invalid mesh regions marked in yellow

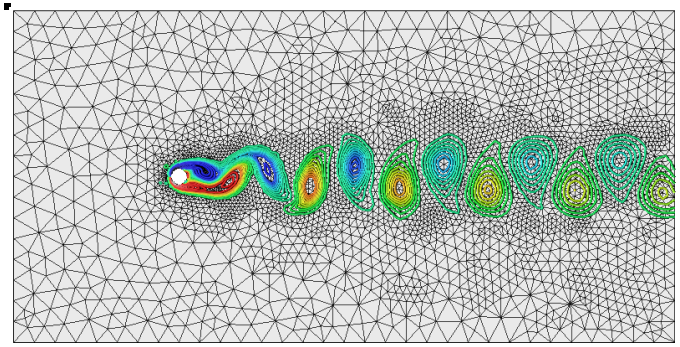


Scalable Analysis Procedure – PHASTA

Stability with Accuracy
- Hierarchic spatial basis (currently p<4) $O(h^{p+1})$
- Stabilized finite element method
- Combined, yield accurate, well controlled, stabilization
- Time integration: explicit (4^{th} order RK) and implicit (2^{nd} order generalized alpha method).

Adaptivity
- Grid matches physical scale
- Anisotropic and transient

Parallel
- Excellent scaling to 288k processors



Parallel Hierarchic Adaptive Stabilized Transient Analysis

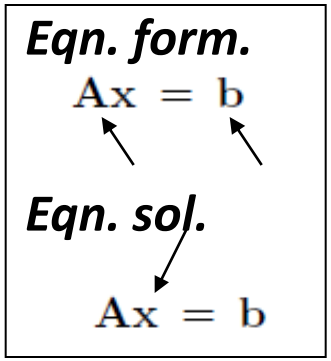
Parallelization of PHASTA

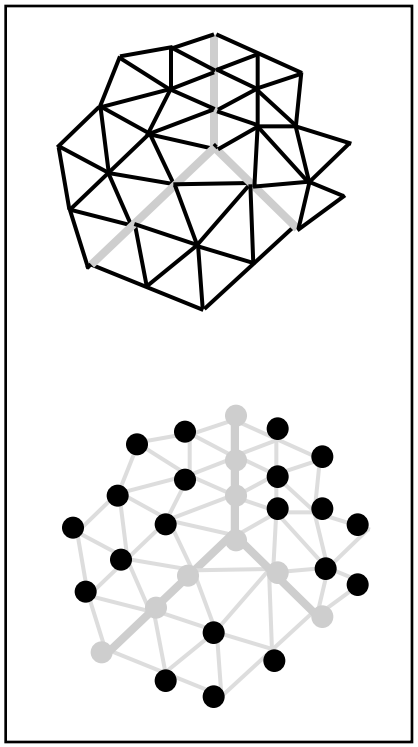
Two main computational steps

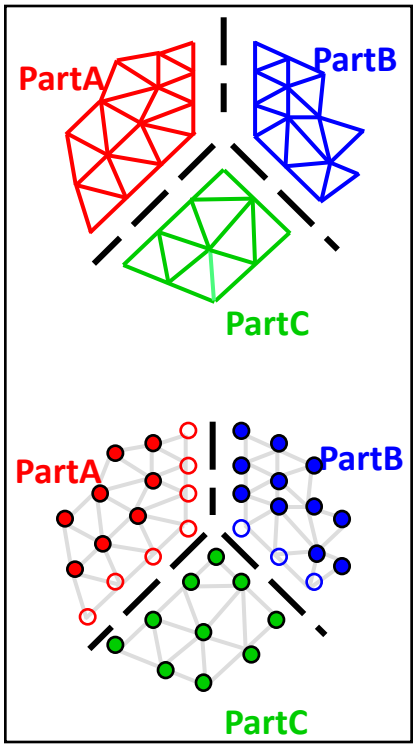
- Formation of element matrices to form global equation
- Solution of resulting algebraic equations

Parallel strategy:

- Both compute stages operate off the same mesh partition
- Partition defines inter-part relations (part-to-part comm.)







Locally, *incomplete* values (in *b*, *A*, *q*, etc.) for shared *dofs.*

Apply communications to *complete* values/entries (in *b*, *q* only)

b during *Eqn. form.*

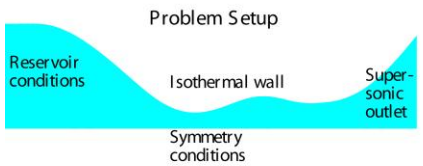
q during *Eqn. sol.*

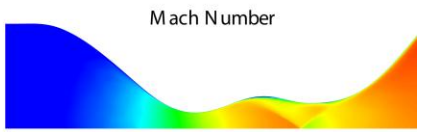
Compressible Implicit Flow Solver – Compress Time

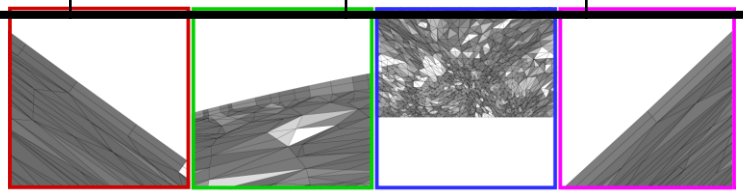
- Strong scalability results for double-throat nozzle (approx. 1.5M elems.):

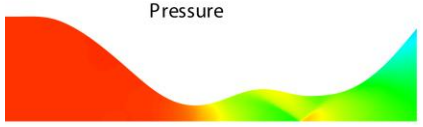
scale factor, $s_i = (t_{base} \times np_{base}) / (t_i \times np_i)$ - 1 implies perfect scaling

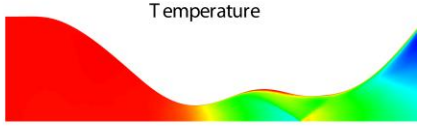
Cores (avg. elems./core)	**Cray XT3 PSC**		**Sun AMD TACC**		**IBM's BG/L RPI-CCNI**	
	t (secs.)	scale factor	t (secs.)	scale factor	t (secs.)	scale factor
16 (96000) - base	390.84	**1** (base)	425.96	**1** (base)	2121.10	**1** (base)
32 (48000)	190.63	**1.03**	208.73	**1.02**	1052.42	**1.01**
64 (24000)	89.57	**1.09**	98.10	**1.09**	528.62	**1.00**
128 (12000)	46.08	**1.06**	50.05	**1.06**	265.37	**1.00**
256 (6000)	24.49	**1.00**	27.70	**0.96**	132.83	**1.00**
512 (3000)	13.28	0.92	14.81	0.90	67.35	**0.98**
1024 (1500)	7.97	0.77	9.63	0.69	33.70	**0.98**
2048 (750)	-	-	-	-	17.13	**0.97**
4096 (375)	-	-	-	-	9.09	0.91
8192 (187)	-	-	-	-	5.00	0.83

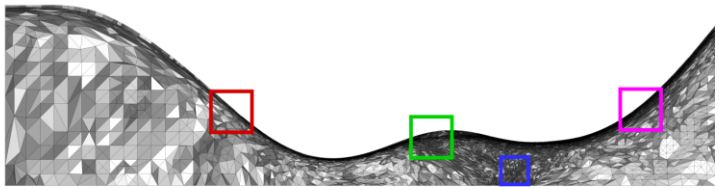








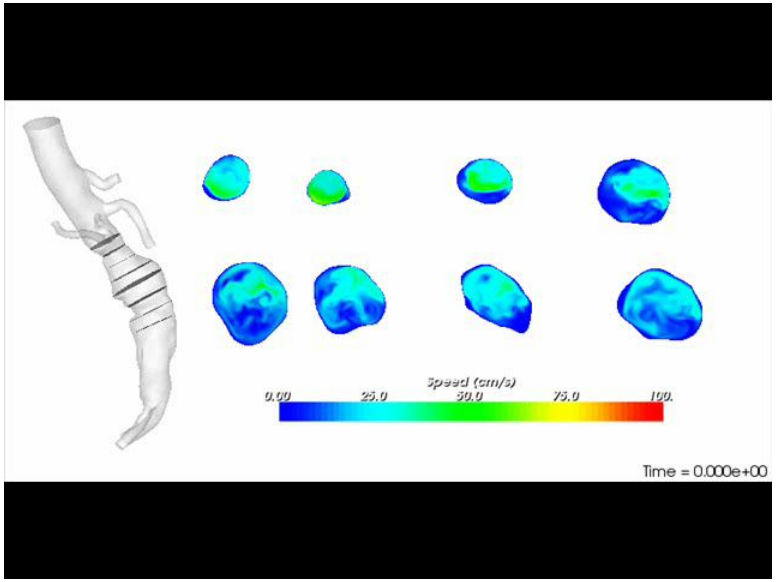




Parallel Implicit Flow Solver – Incompressible

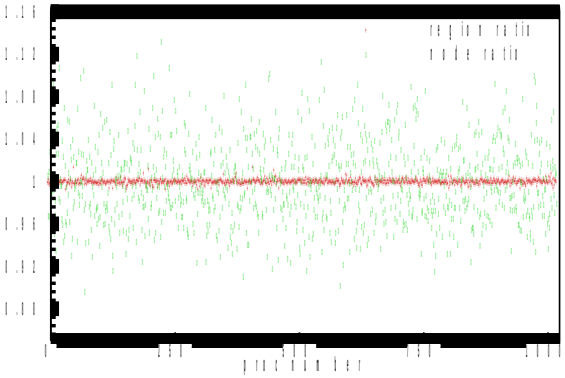
Abdominal Aorta Aneurysm (AAA) 105 Million Elements

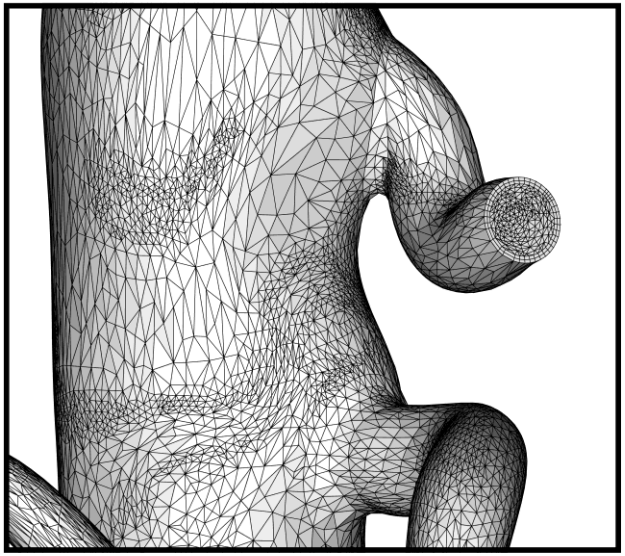
Cores (avg. elems./core)	IBM BG/L RPI-CCNI	
	t (secs.)	scale factor
512 (204800)	2119.7	**1** (base)
1024 (102400)	1052.4	**1.01**
2048 (51200)	529.1	**1.00**
4096 (25600)	267.0	**0.99**
8192 (12800)	130.5	**1.02**
16384 (6400)	64.5	**1.03**
32768 (3200)	35.6	0.93



32K parts show modest degradation due to 15% node imbalance (with only about 600 mesh-nodes/part)

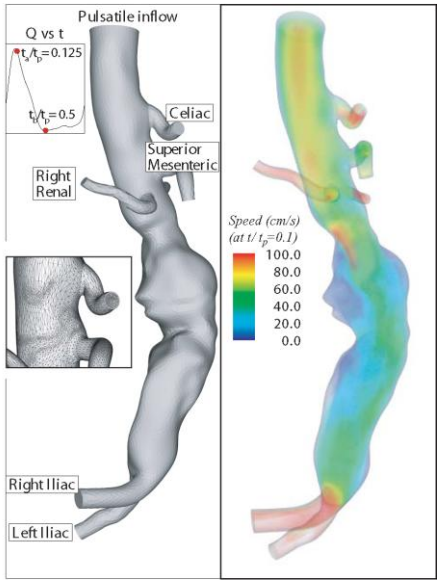
Rgn./elem. ratio$_i$ = rgns$_i$/avg_rgns
Node ratio$_i$ = nodes$_i$/avg_nodes

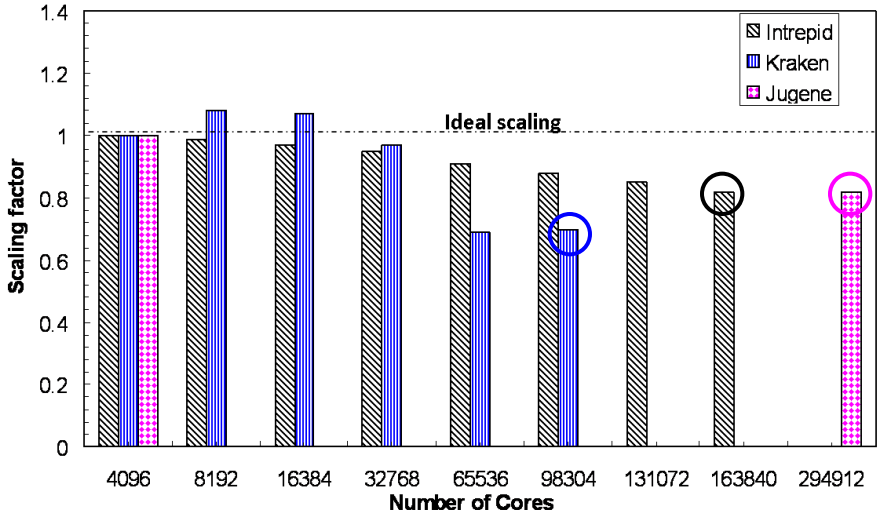




Strong Scaling – 1B Mesh up to 288k Cores

AAA 1B elements: three supercomputer systems up to full-scale





1.07B elements mesh		Intrepid: IBM BG/P		Kraken: Cray XT5		Jugene: IBM BG/P	
num. of cores	avg. elem./core	time	s-factor	time	s-factor	time	s-factor
4,096 (base)	261,600	844.38	1	311.34	1	845.68	1
8,192	130,800	427.33	0.99	144.23	1.08	–	–
16,384	65,400	217.05	0.97	73.06	1.07	–	–
32,768	32,700	109.87	0.96	39.35	0.97	–	–
65,536	16,350	58.65	0.91	28.04	0.69	–	–
98,304	10,900	39.06	0.90	18.67	0.70	–	–
131,072	8,175	29.68	0.89	–	–	–	–
163,840	6,540	24.12	0.88	–	–	–	–
294,912	3,630	–	–	–	–	14.39	0.82

Sahni et. al., SC09

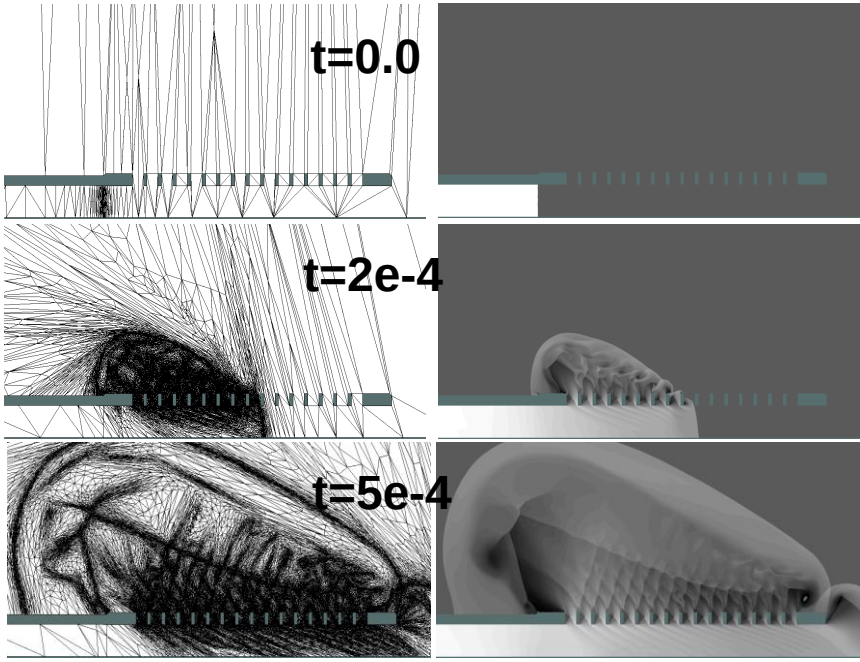
18.67 on XT5 and 14.39 on BGP secs for 20 time steps

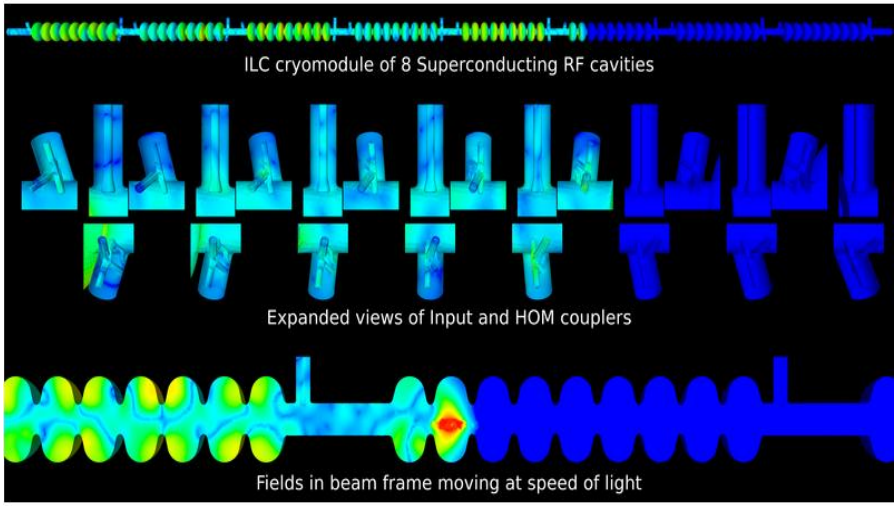
Adaptive Loop Construction

Component based using parallel simulation automation components and existing analysis procedures

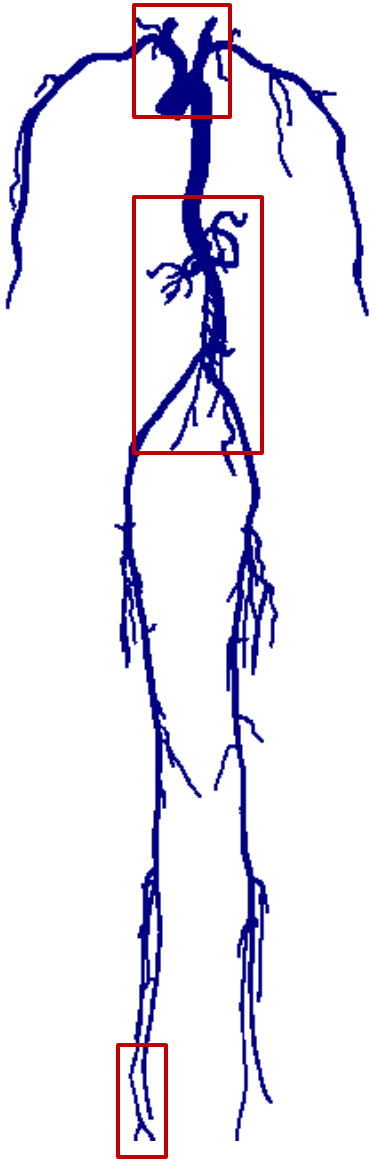
Analysis procedures used to date

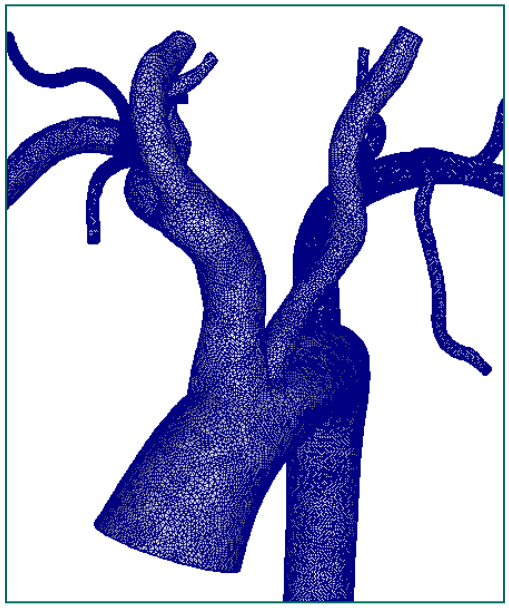
- PHASTA – for various flow simulations
- Commercial CFD codes (e.g., AcuSolve)
- DG code (example to upper right)
- FUN3D flow solver from NASA
- DOE SLAC ACE3P high-order FE code for electromagnetics (example to lower right)
- DOE PPPL M3D-C1 MDH code

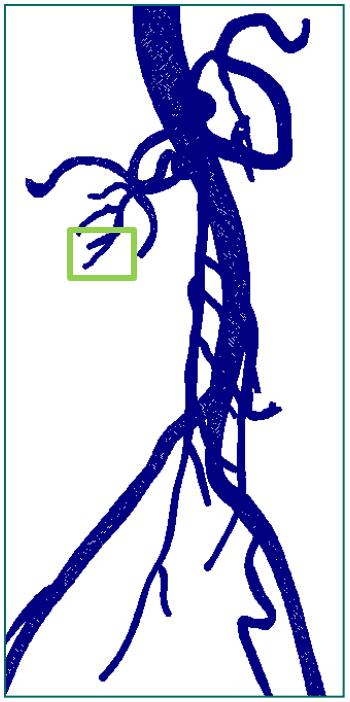


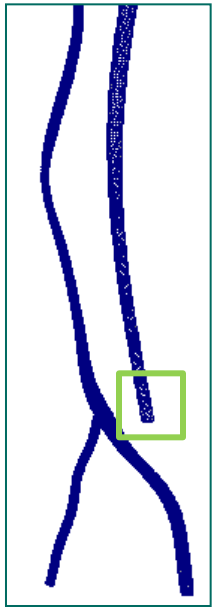


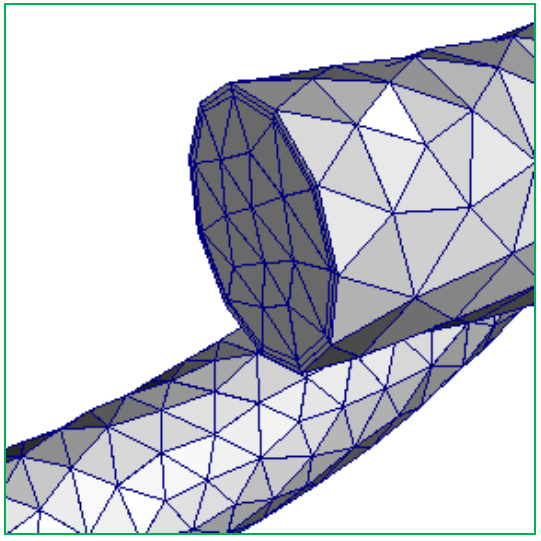
Patient Specific Vascular Surgical Planning

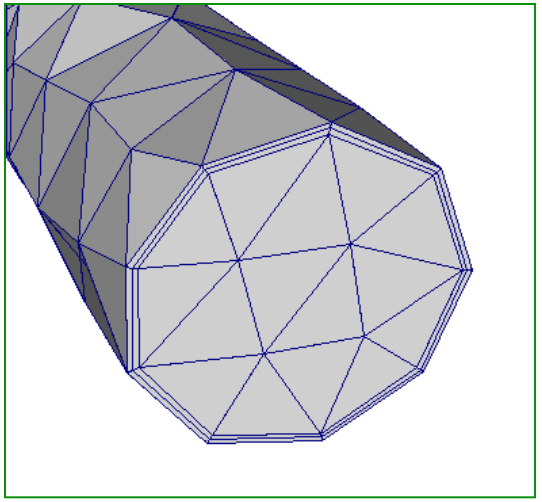












- The initial mesh has 7.1 million regions
- The local size mesh size is between 0.03cm and 0.1cm
- The initial mesh is isotropic
- Initial mesh BL with 0.004cm as the height of each

Patient Specific Vascular Surgical Planning

4 mesh adaptation iterations

The adapted mesh:42.8 million regions

7.1M->10.8M->21.2M->33.0M->42.8M

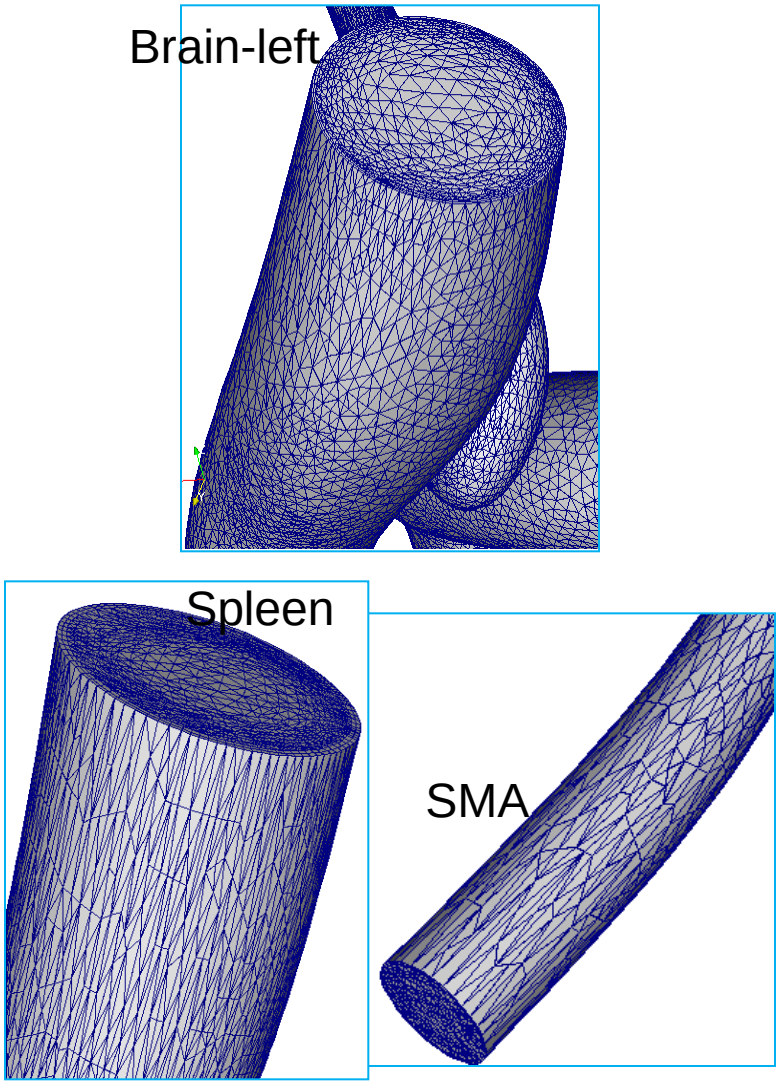
Boundary layer based mesh adaptation

Mesh is **anisotropic**

The minimum local size: 0.004cm, maximum local size: 0.8cm, and the height of the boundary layer: 0.004cm.

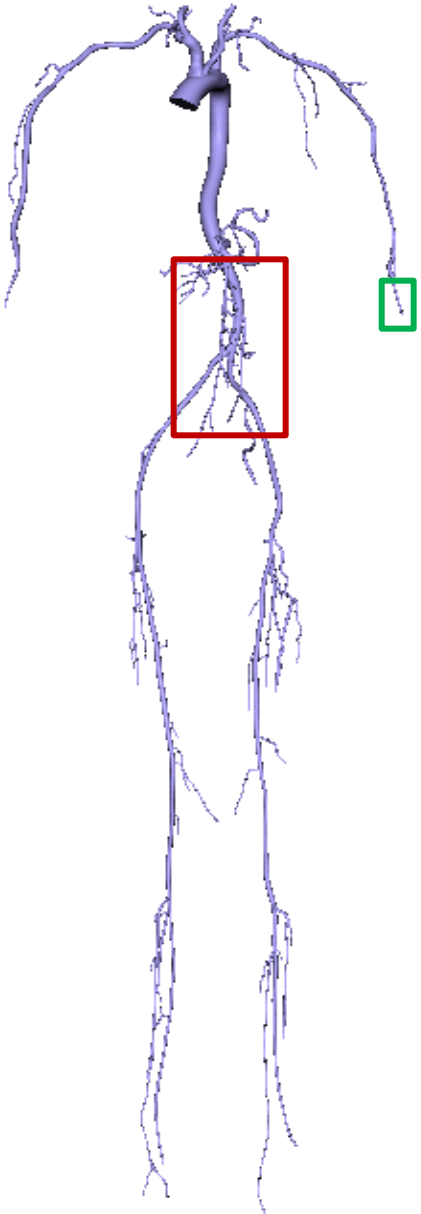
Note: the inflow diameter is 3cm, and the total model length is more than 150cm.

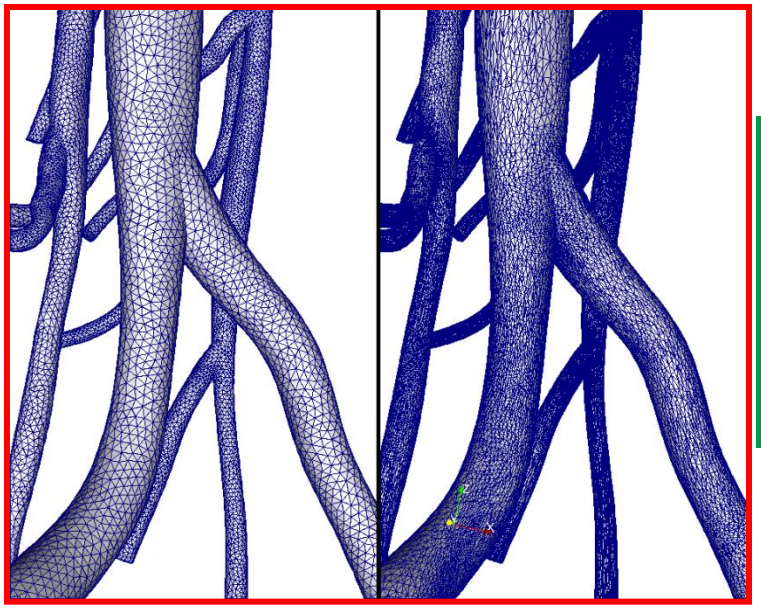
Mesh adaptation driven by 2nd derivatives of appropriate solution field (velocity and pressure in current case)

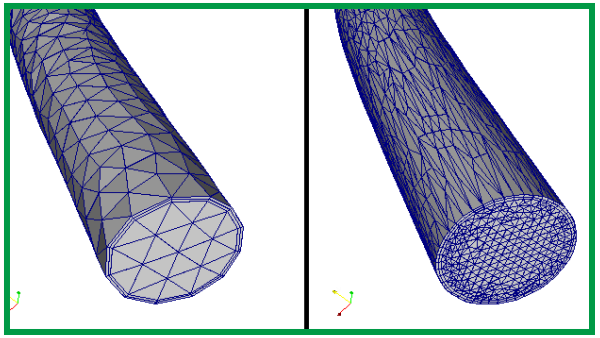


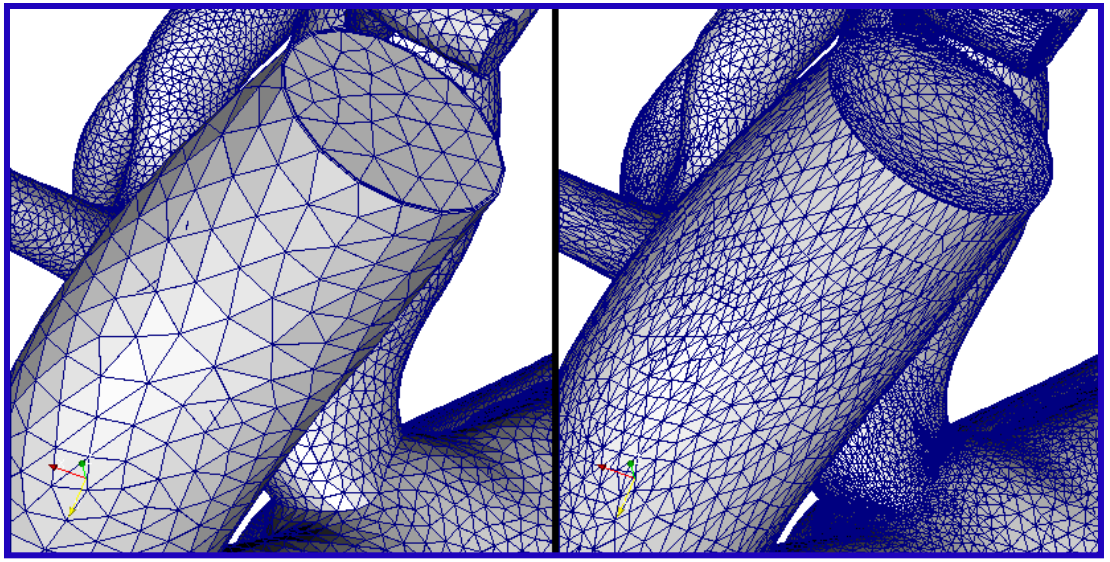
Anisotropic adapted mesh

Patient Specific Vascular Surgical Planning

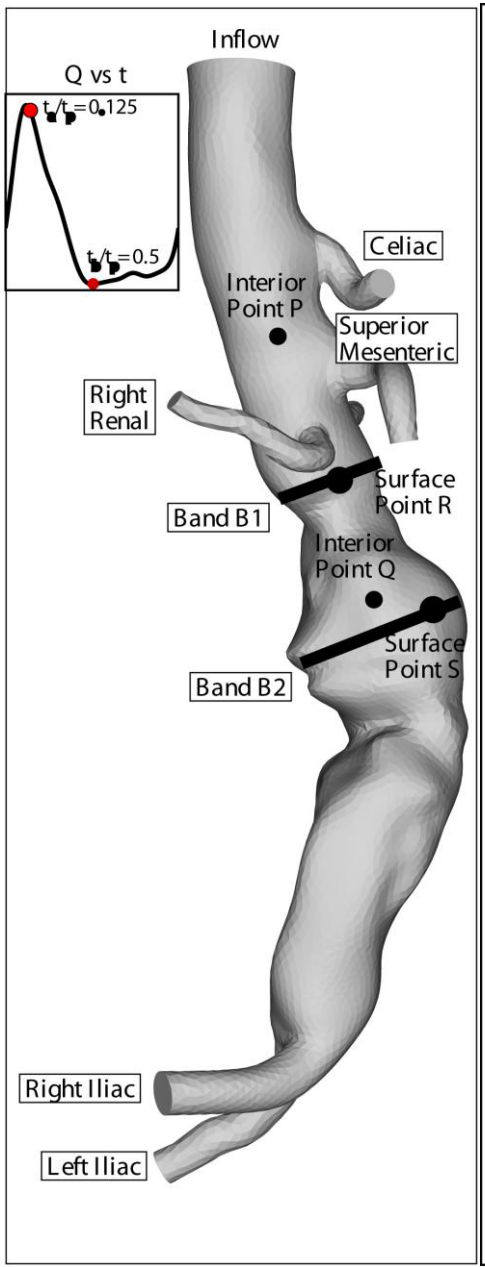


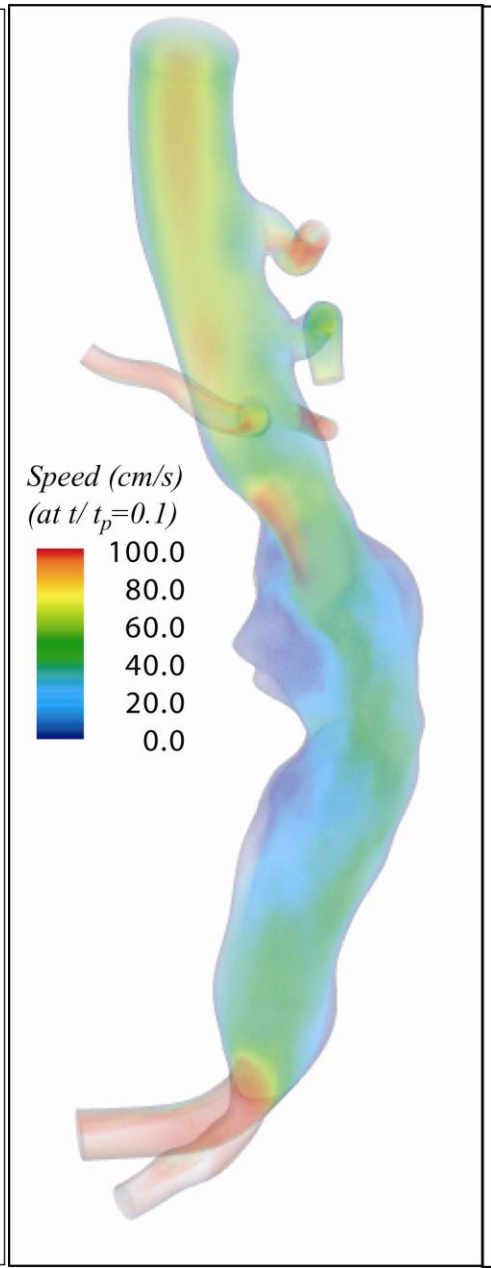


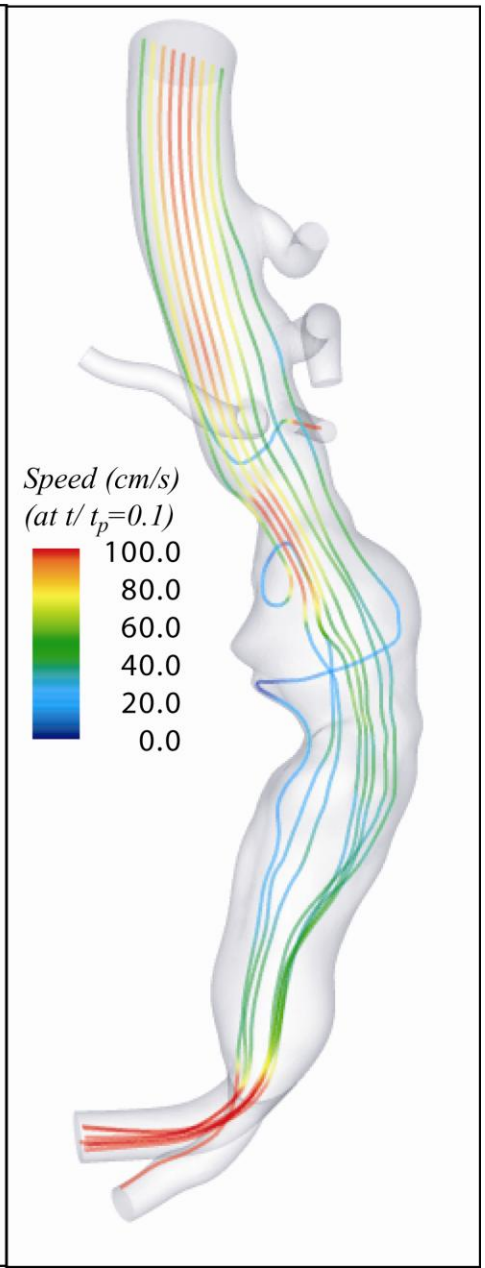




Patient-Specific Abdominal Aortic Aneurysm

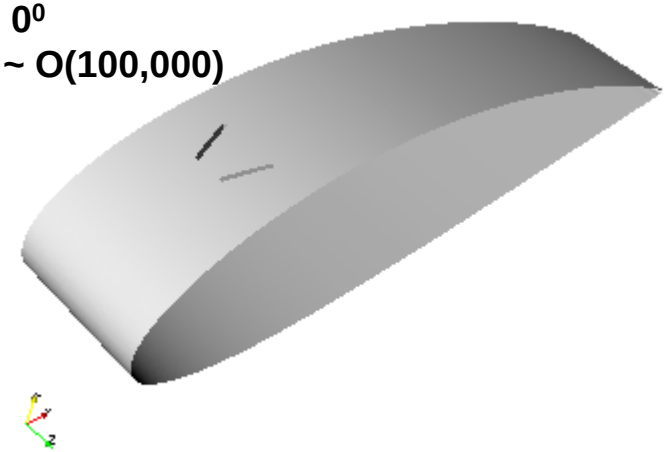


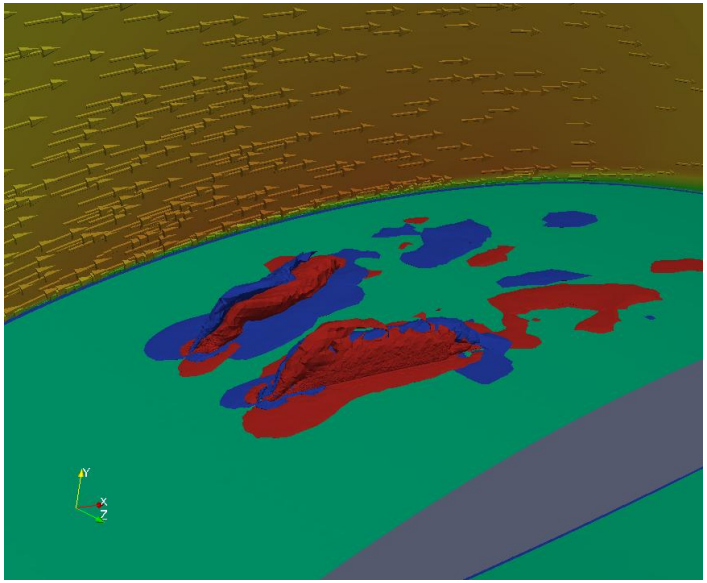


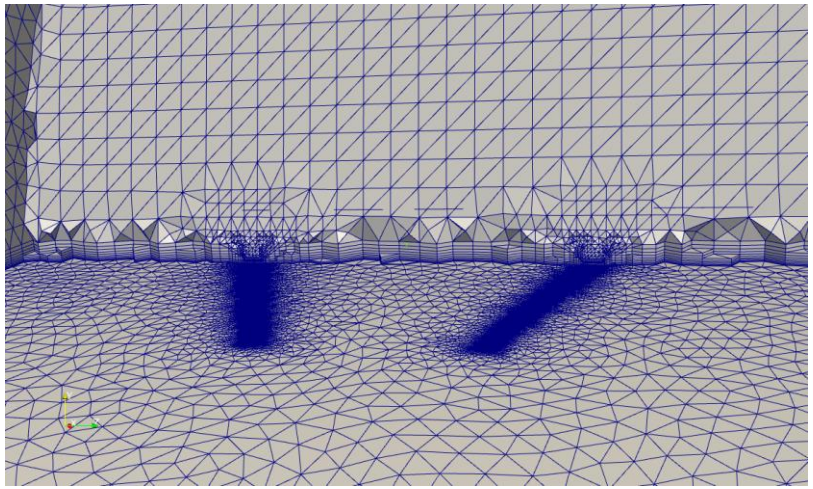


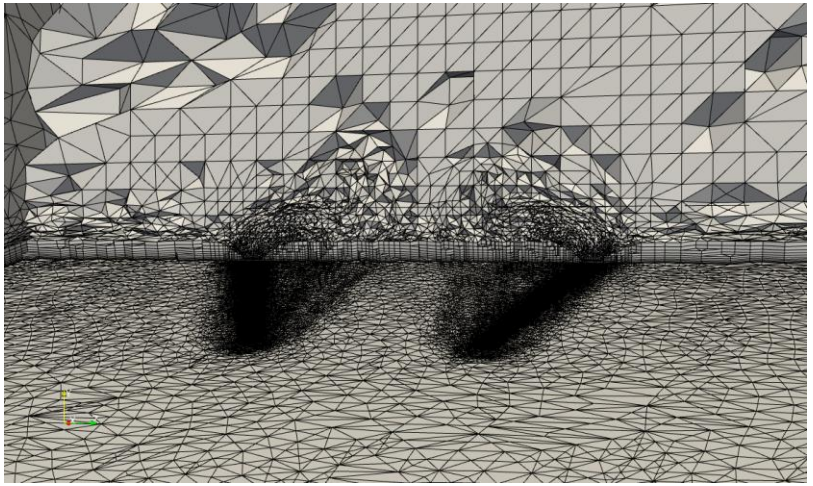
Mesh Adaptivity for Synthetic Jets (O. Sahni)

f_{act} = 2,300Hz
$\alpha = 0^0$
Re ~ O(100,000)



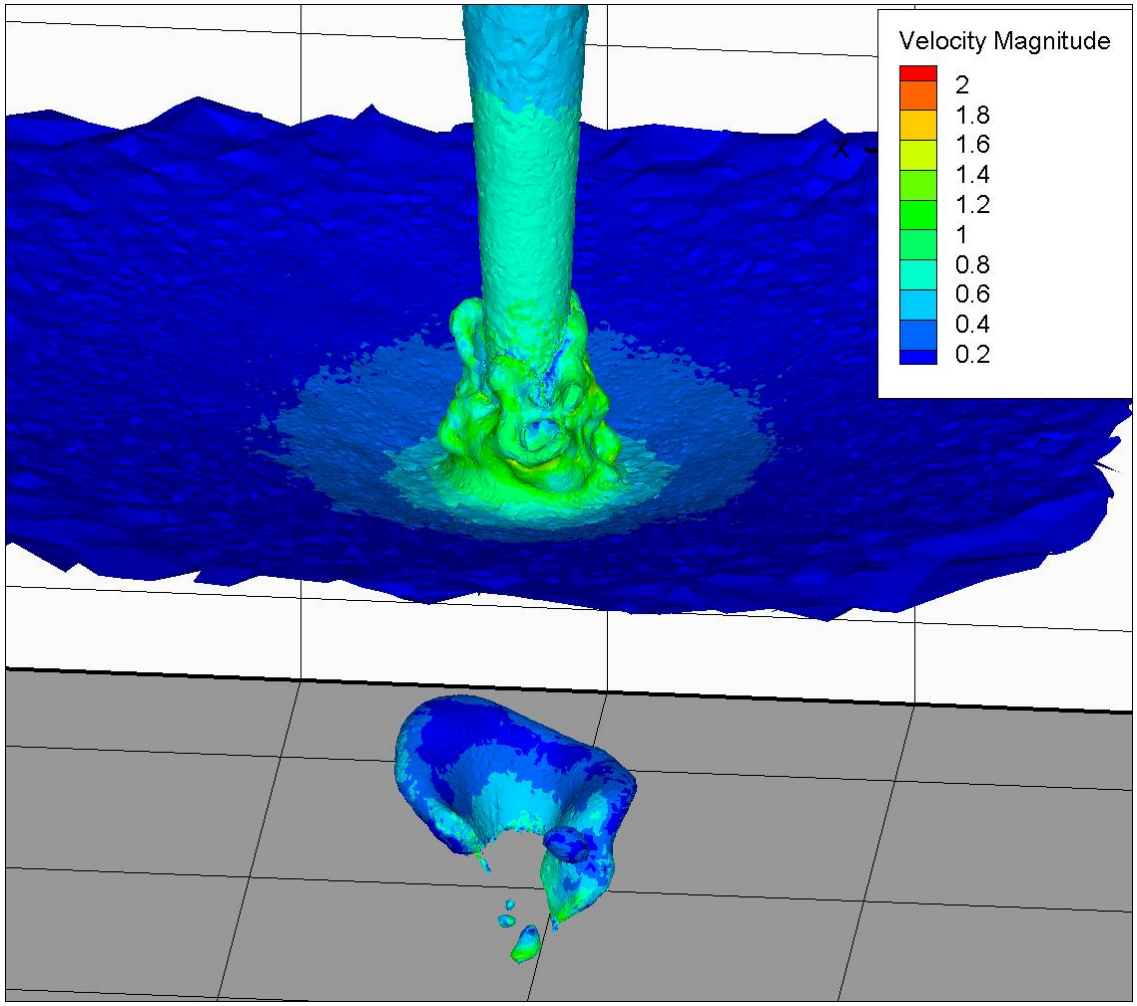


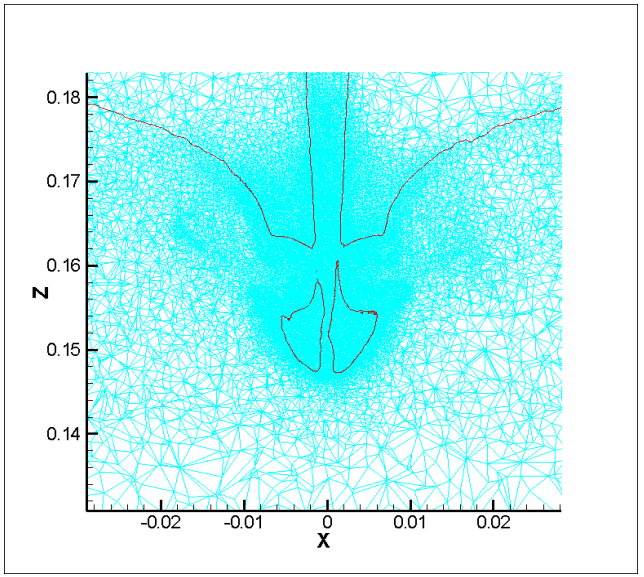




Current Two-Phase Research (Jansen and Lahey)

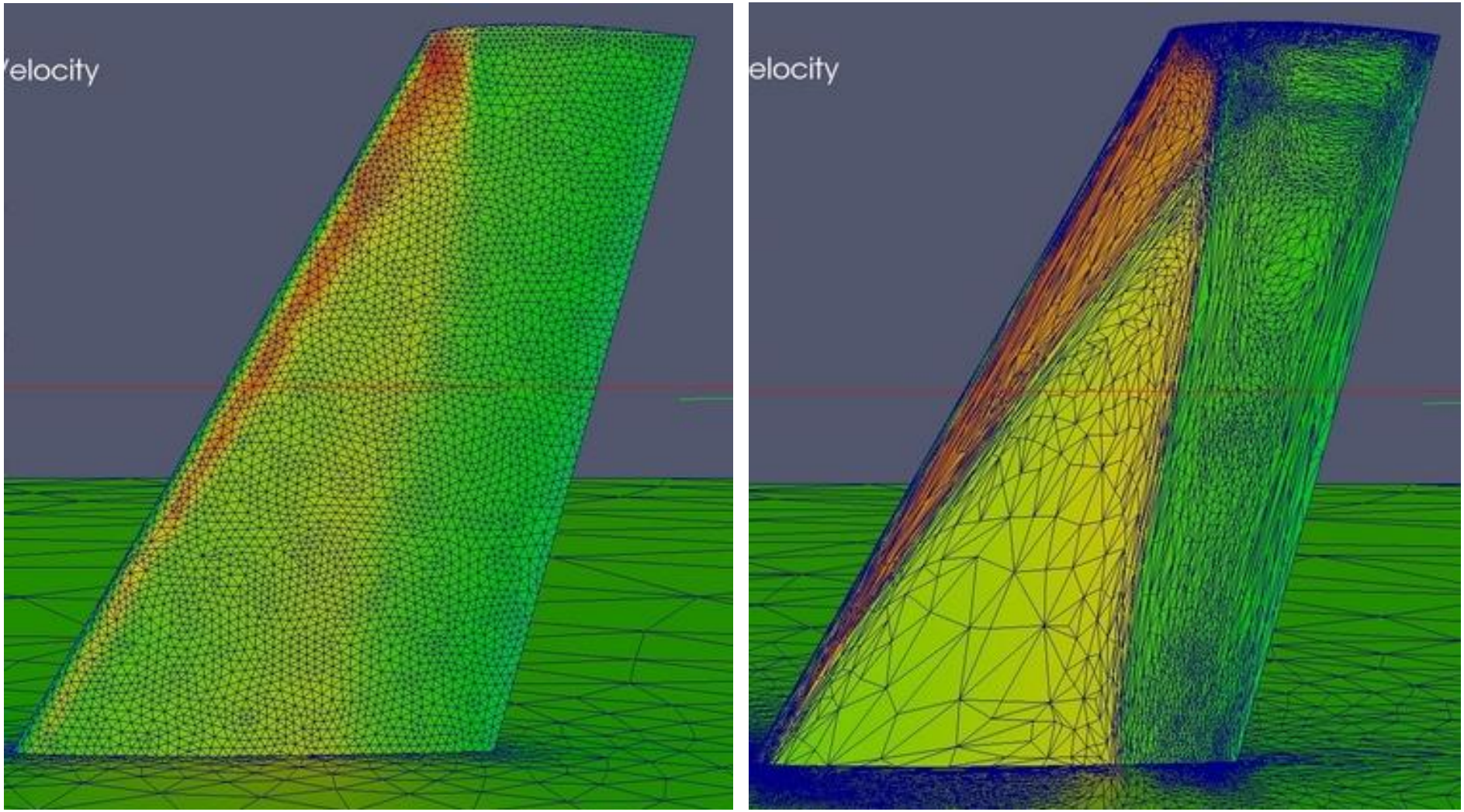
Air Entrainment from Breaking Waves and Plunging Jets (Galimov)





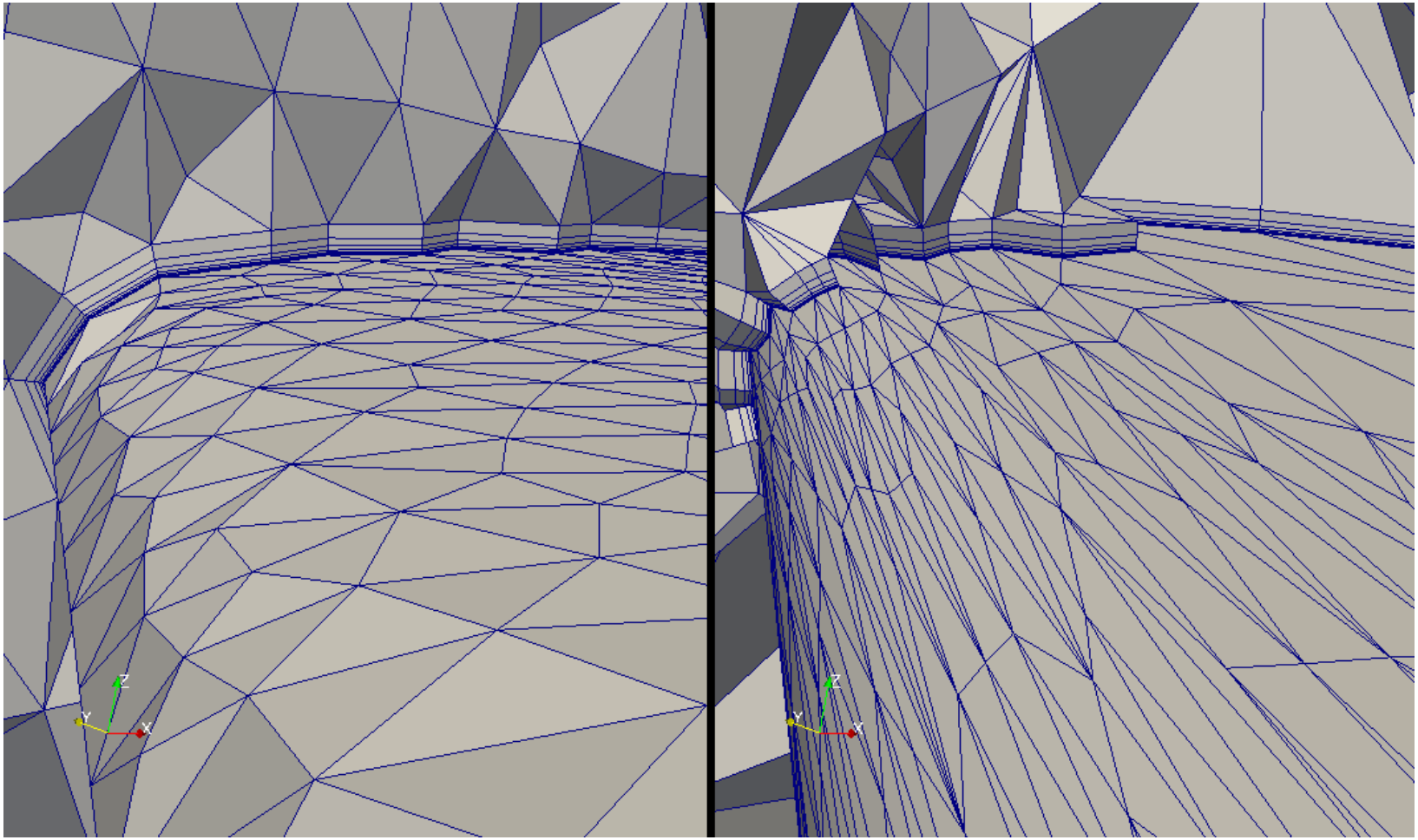
(earlier time adapted mesh)

Anisotropic Adaptivity for Aerodynamics

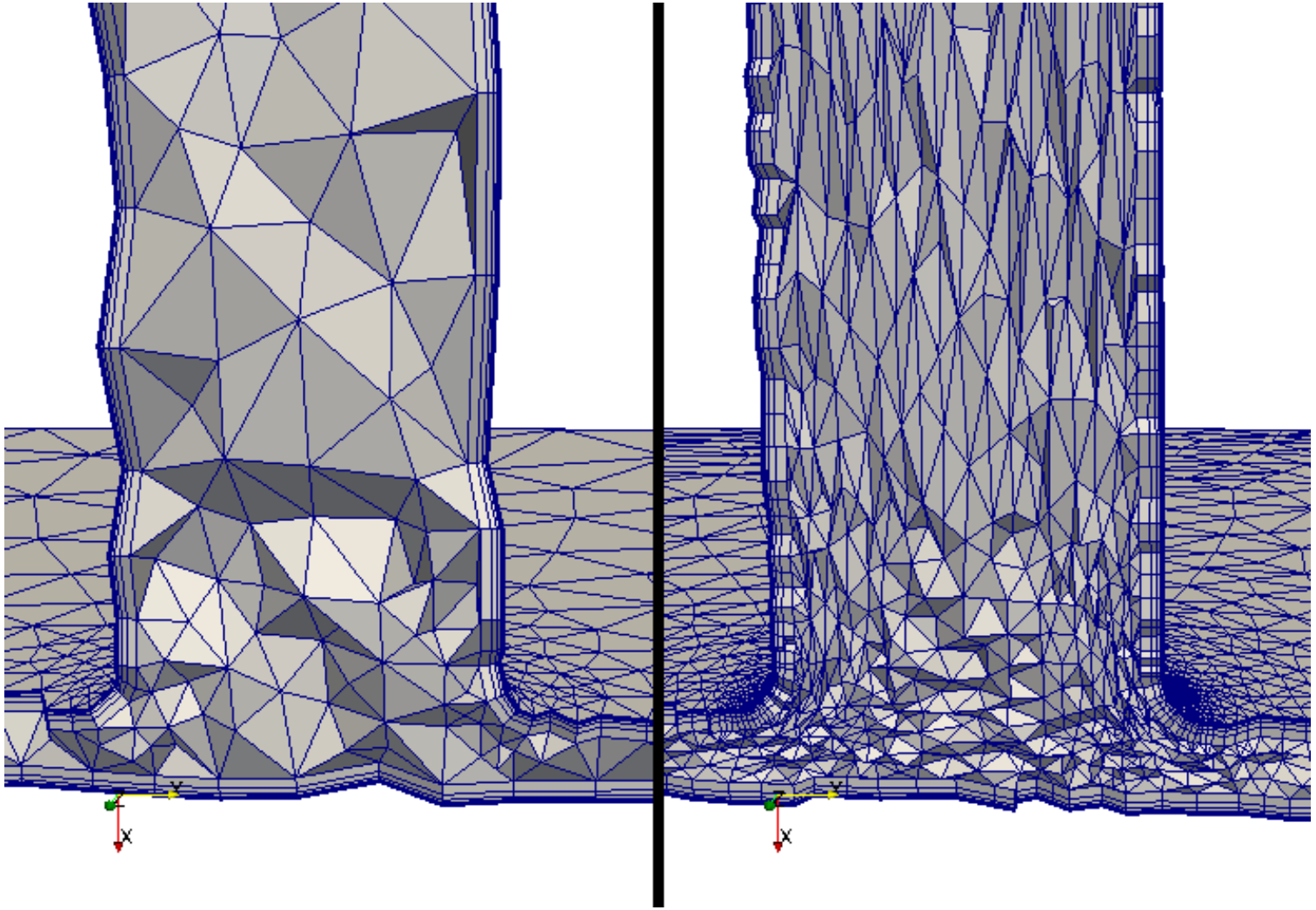


Anisotropic adaptation – ONERA M6 wing

Example of Anisotropic Adaptation



Close-up of Boundary Layer Adaptivity



BL Adaptivity – pipe manifold example

Industrial Applications

NY State supported project where RPI is bringing high performance computing to industry

Development HPC work flows using components

Examples

- Modeling Two-phase Flows - Interfacing commercial, industrial, and research software
- Modeling Viscous Flows - Increasing model complexity while evaluating scalability
- Modeling Pump Flows - Tools for high performance CFD with rotating geometry and cavitation

Modeling Two-phase Flows

Objectives

- Demonstrate end-to-end solution of two-phase flow problems.
- Couple with structural mechanics boundary condition.
- Provide interfaced, efficient and reliable software suite for guiding design.

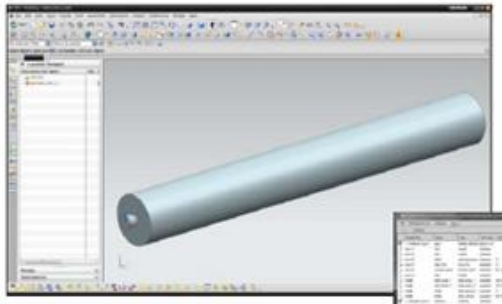
Investigated Tools (to date)

- Simmetrix SimAppS – mesh generation and problem definition
- PHASTA – two-phase level set analysis and adaptation software
- Commercial CFD Software - two-phase flow solver
- Structural mechanics analysis software
- Kitware Paraview – visualization

Status and Plans

- Demonstrated automated mesh adaptivity
- Implemented file based interface to couple with structural mechanics analysis software
- Evaluated scaling of adaptive coupled simulation on CCNI Opterons

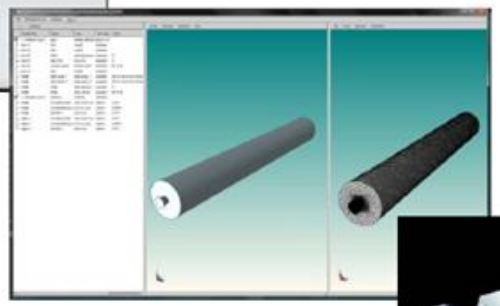
Workflow and Tools



NX

SOLIDWORKS

Geometric Modeling



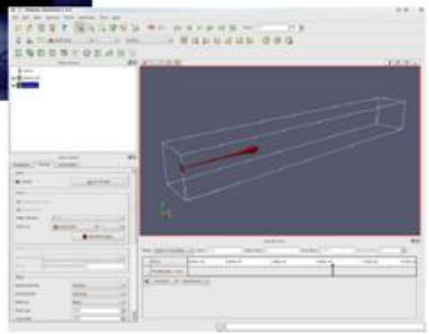
Simmetrix Inc. Enabling Simulation-Based Design

Mesh Generation and Problem Definition

PHASTA (Open Source CFD)

ACUSIM SOFTWARE

Flow Solver, Structural Mechanics Solver, and Mesh Adaptation

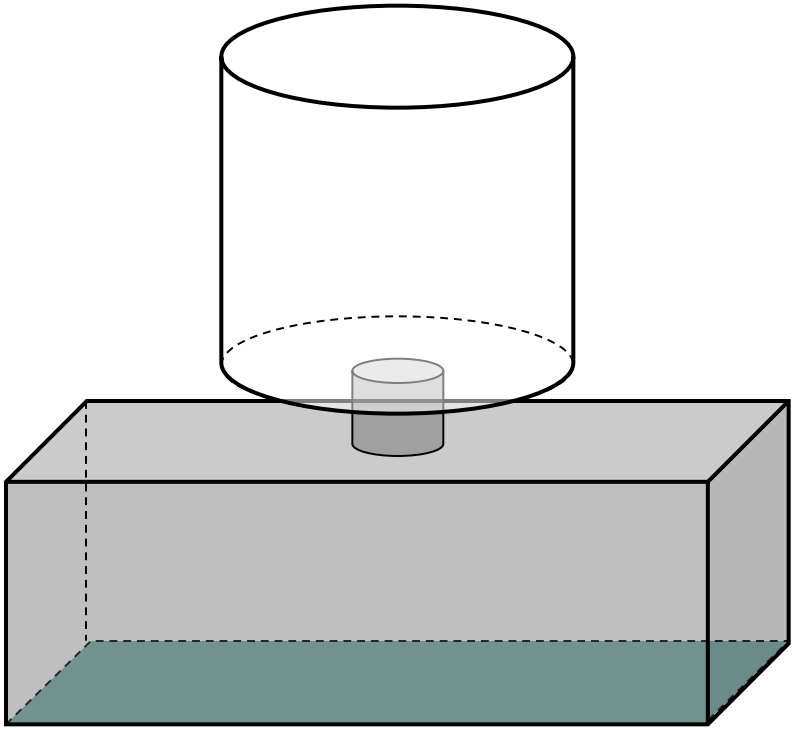


ParaView

Post-Processing

Example Simulation

- Two-phase modeling using level-sets in PHASTA
- Natural pressure BC on outlet surface
- BC on membrane is derived from the given displacements (see below)
- No-slip velocity BC on all other surfaces
- Initial pressure and velocity is zero



- Schematic of the geometry:
  - gray is the fluid chamber
  - dark gray is the membrane
  - white is air

Two-Phase Automated Mesh Adaptation

Simulation

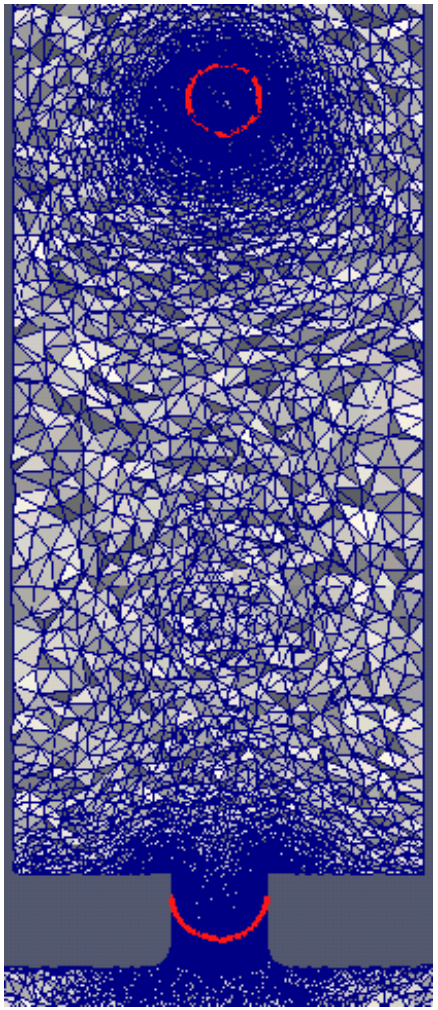
- Two-phase modeling using level-sets
- Natural pressure BC on outlet surface
- BC on membrane is derived from the given displacements

Fixed mesh

- Max size: ~20.0M elements
- Simulation runs until the fluid-air interface reaches the coarse mesh region

Adaptive mesh

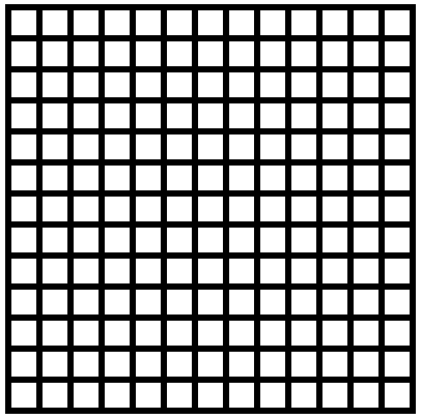
- Max size: 0.98M elements
- Anisotropy factor of 4.0 in the nozzle and around the 0-levelset
- Simulation runs until the detached fluid exits the computational domain



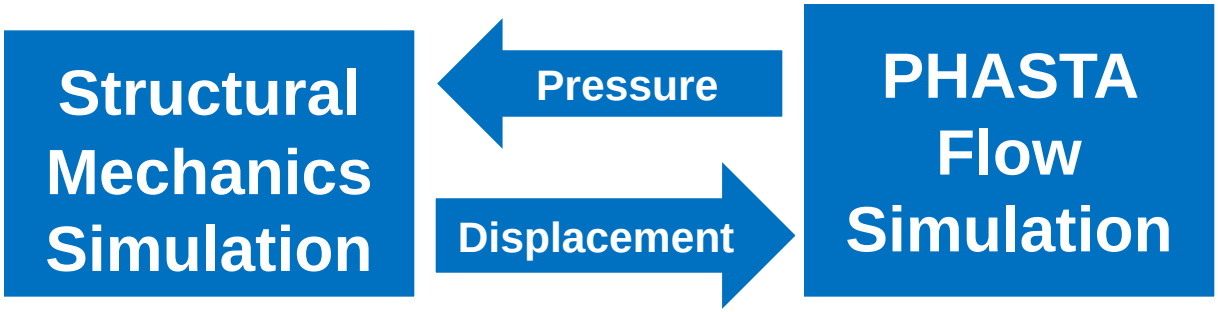
Axial 2D slice of two-phase flow. Red line depicts the fluid-air interface.

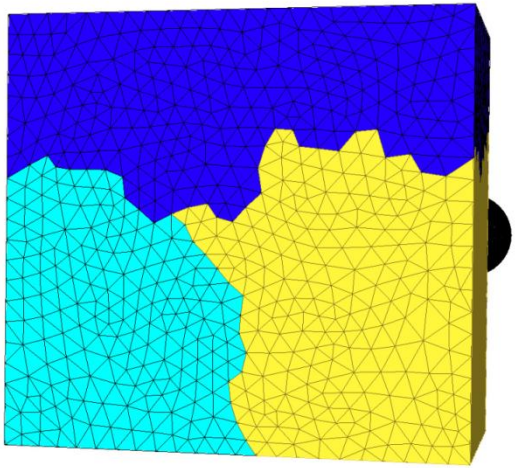
Fluid Structure Interactions

- Implemented functionality to couple with structural boundary condition.
- Spatial (temporal) interpolation supports different discretizations in structural mechanics and PHASTA simulations.



Structural Mechanics Mesh of Input Face



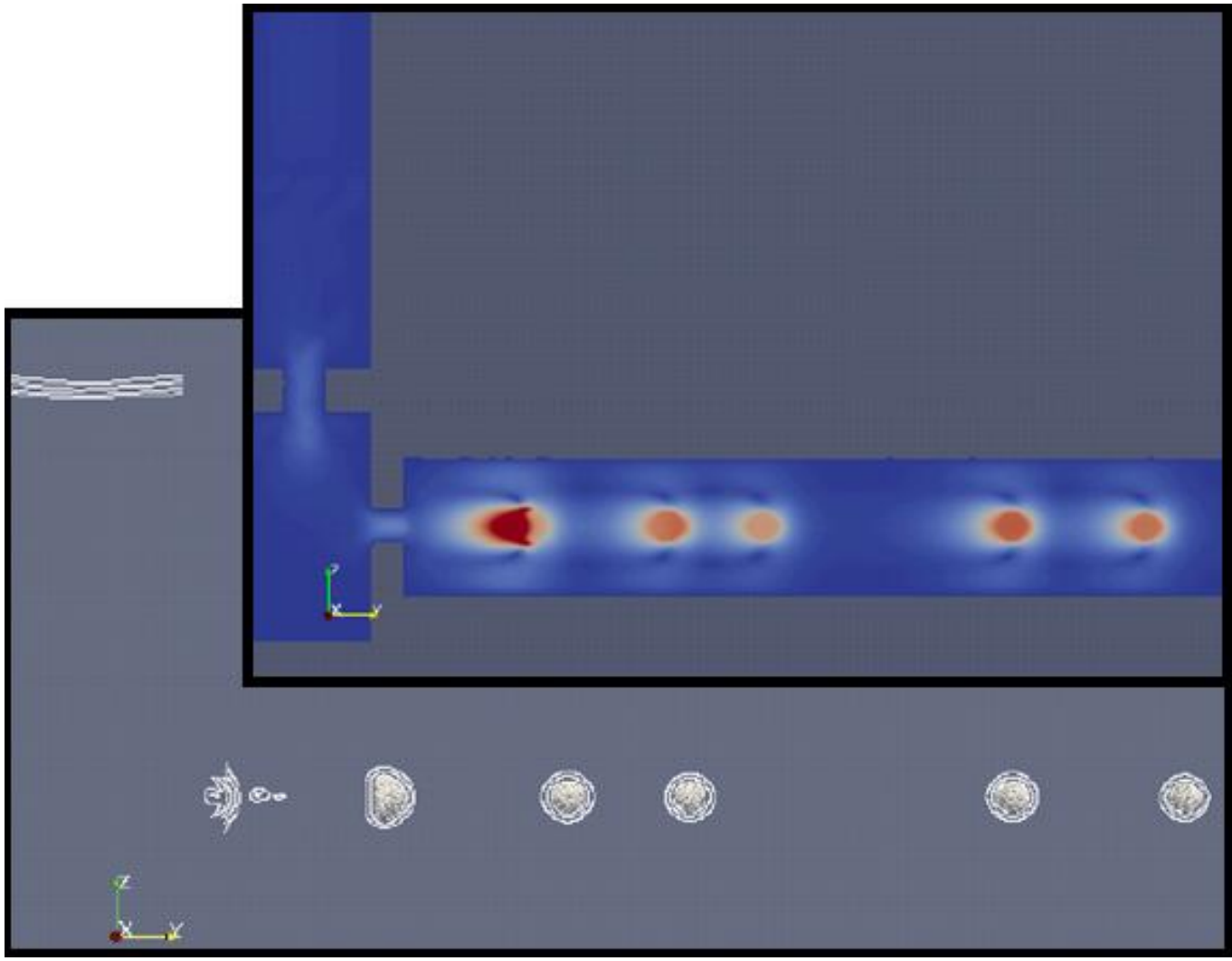


PHASTA Partitioned Mesh of Input Face

Multiple Ejection Cycles

Demonstration case with five ejection cycles

Axial 2D slice of 3D simulation. The magnitude of velocity is depicted in (a). The 0, 1um, and 2um levelsets are depicted with white lines in (b).



Ongoing Efforts

Remove file I/O from PHASTA / Mesh Adaptation Loop

- DOE funded research effort

Improving volume control during flow solve

Improving Mesh Adaptation

- Improved mesh metric that accounts for velocity gradient – bias in the flow direction
- Volume control during mesh adaptation

Extend Structural Mechanics Software to Support Two-way Coupling

Evaluate Scaling of Commercial CFD Software on CCNI Opterons

- Working with vendor's engineers to install and test the software.

Modeling Viscous Flows

Objectives

- Demonstrate capability of available computational tools/resources for parallel simulation of highly viscous sheet flows.
- Solve a model sheet flow problem relevant to the actual process/geometry.
- Develop and define processes for high fidelity twin screw extruder parallel CFD simulation.

Investigated Tools (to date)

- ACUSIM Suite – free surface sheet flow
- Simmetrix SimAppS and MeshSim - mesh generation
- Kitware Paraview

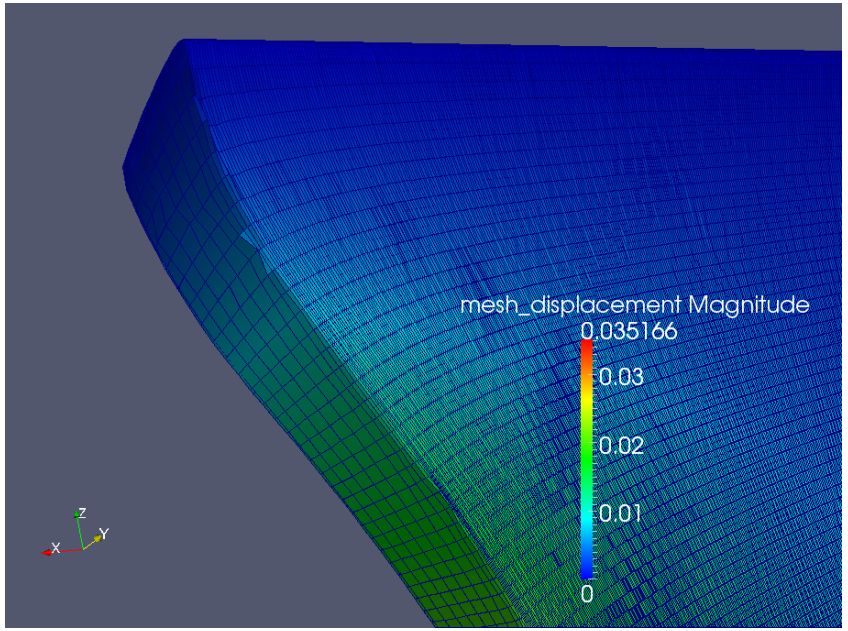
Status and Plans

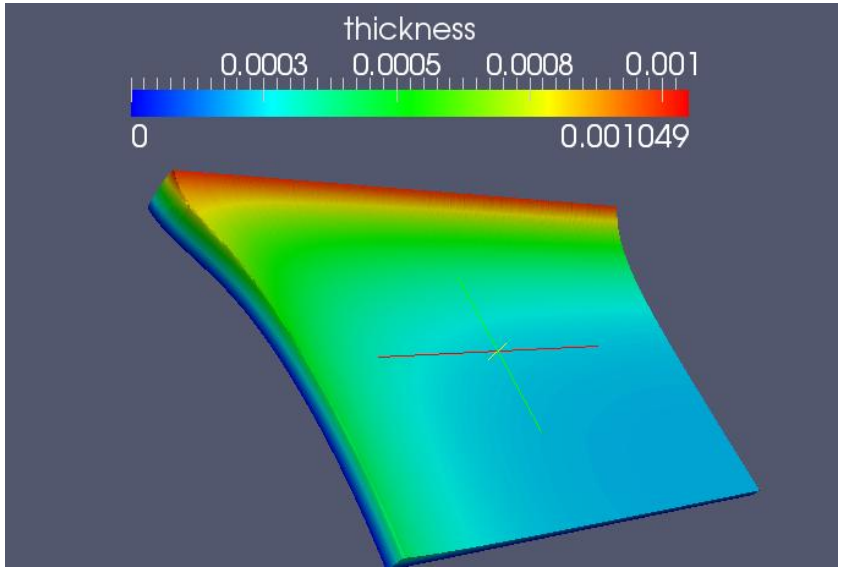
- Iterated with industrial partner and ACUSIM engineers on setup of viscous 3D sheet flow problem.
- Performed strong scaling test to 512 cores of sheet flow simulations
- Generated mesh of twin screw extruder geometry for in-house ANSYS Polyflow simulations.
- Provide access to ANSYS Polyflow

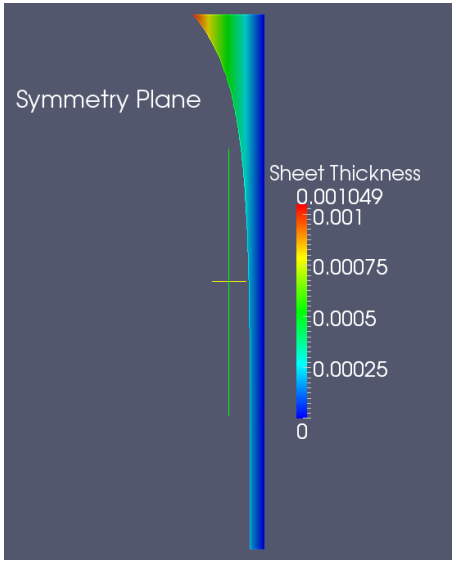
Simplified 3D Sheet Flow Simulation

High Aspect Ratio Sheet

- Aspect ratio : 500:1
- Element count: 1.85 Million
- Velocity boundary conditions
- Draw Ratio (Vout/Vin): 5

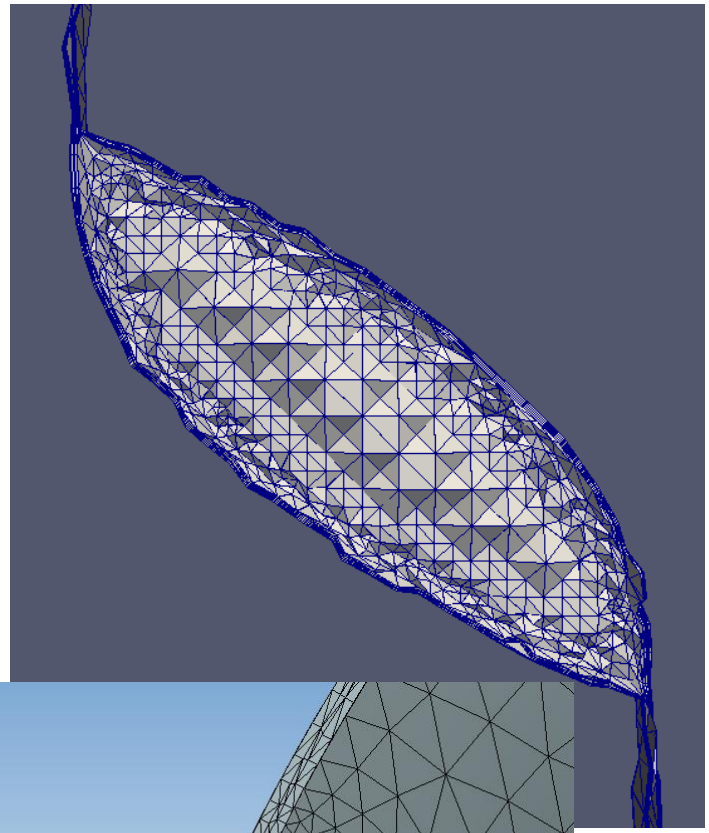


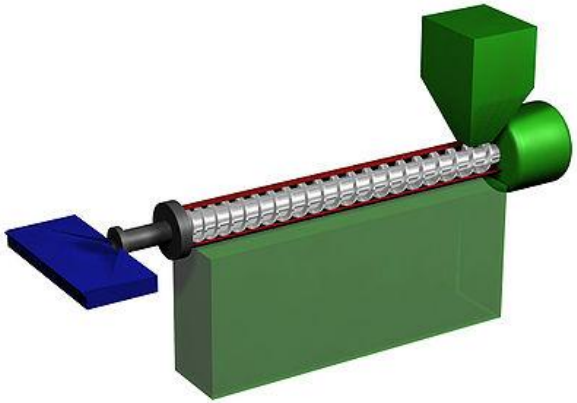




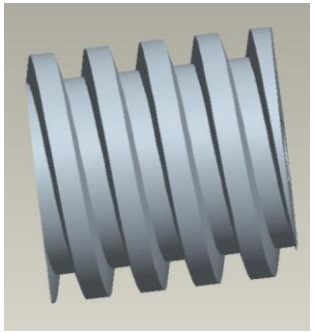
Screw Extruder: Simulation Based Design

- Mesh generation in Simmetrix SimAppS graphical interface.
- Gaps that are ~1/180 of large feature dimension.
- Boundary layers with transition to anisotropic interior mesh.

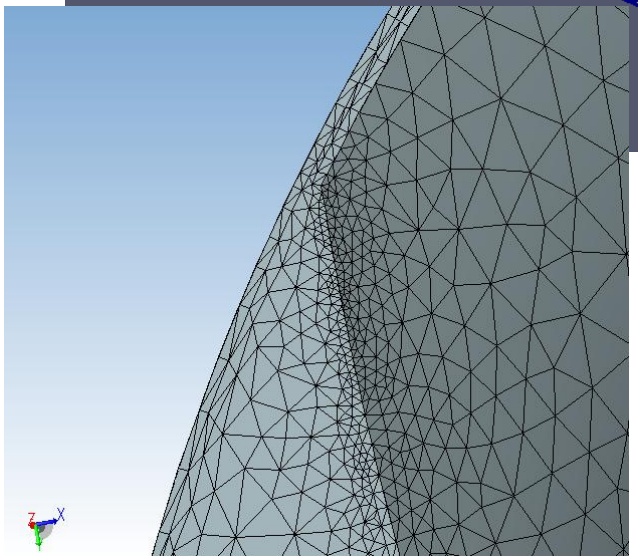




Conceptual Rendering of Single Screw Extruder Assembly*



Single Screw Extruder CAD**



* http://en.wikipedia.org/wiki/Plastics_extrusion

** https://sites.google.com/site/oscarsalazarcespedescaddesign/project03

Modeling Pump Flows

Objectives

- Setup and run 3D pump flow simulations in hours instead of days.
- Provide automated mesh generation for geometries with rotating components.

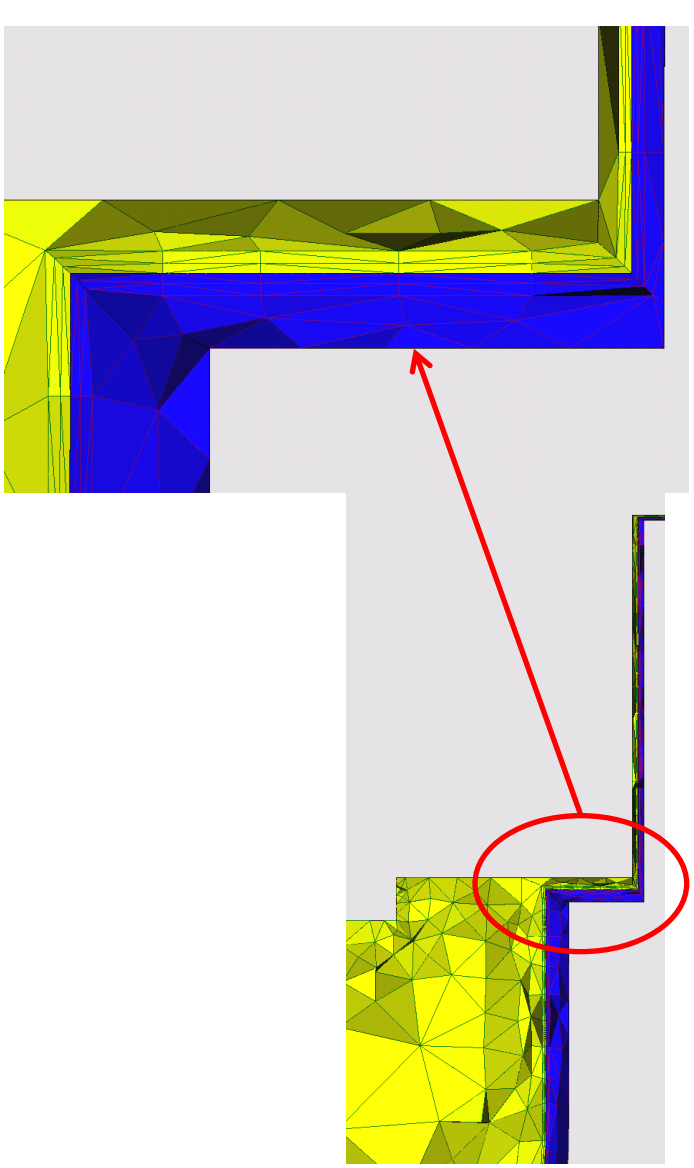
Investigated Tools (to date)

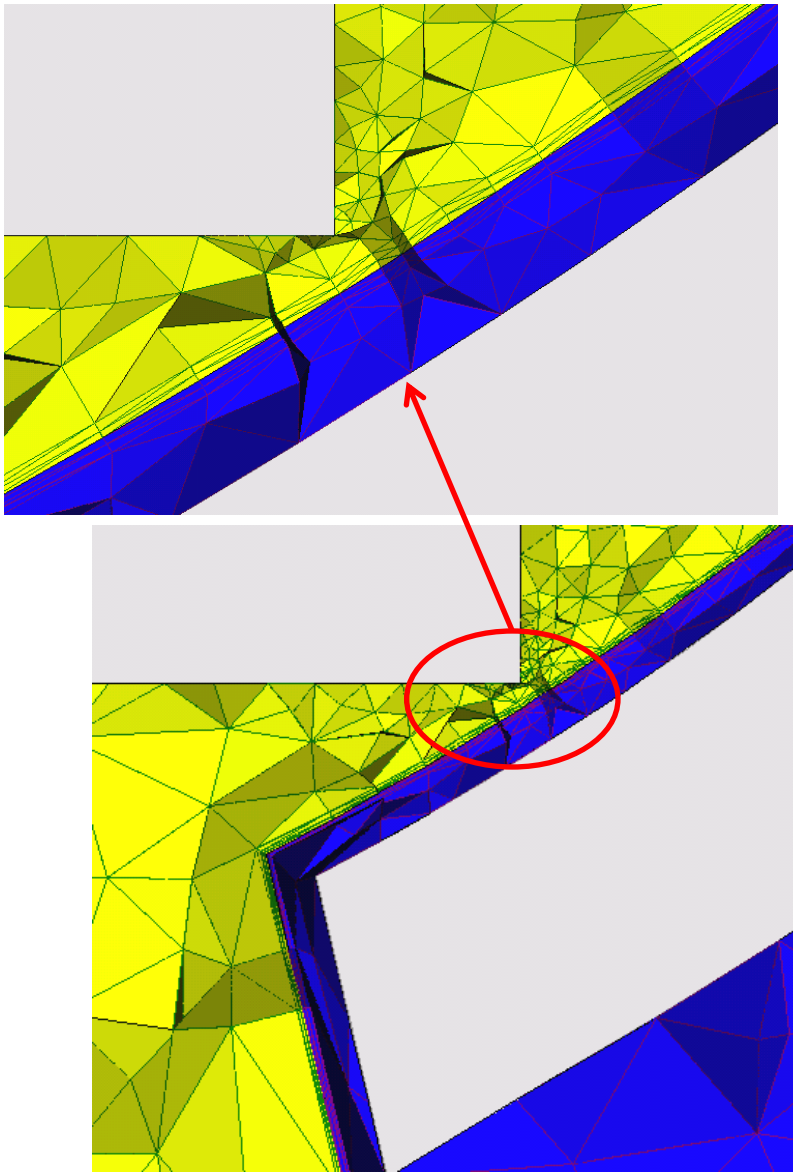
- Simmetrix SimAppS Graphical Interface – mesh generation and problem definition
- FMDB – mesh database
- ACUSIM Suite – CFD software supporting rotating geometry
- ANSYS CFX – CFD software supporting rotating geometry
- PHASTA – two-phase level set flow solver
- Kitware Paraview – visualization

Status and Plans

- Create custom GUI for pump meshing and problem definition.
- Extending FMDB and PHASTA to support rotating geometries and cavitation.
- Provide access to ANSYS CFX

Critical Mesh Regions (can not show full geometry)





Closing Remarks

- Component-based procedures can effectively support automated adaptive simulations
  - Having all components operate in parallel is required for future simulations due to problem size and computer system architectures – increases complexity
- Status
  - A good set of initial tools available
  - Applied to research problems, moving to industrial apps.
  - Massively parallel systems forced developing new procedures to gain scalability
  - Next generation high core count computers will add even more complexity

Contact Mark Shephard (shephard@rpi.edu)