

Integration of Albany and Mesh Adaptation for Parallel Applications

**Glen Hansen* Brian Granzow Dan Ibanez
Seegyong Seol Mark Shephard**

***Sandia National Laboratories**

Rensselaer Polytechnic Institute

SIAM PP 2014

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.





FASTMath: Adaptivity across the software stack

Albany : agile component-based parallel unstructured mesh application

- **Agile component design provides**
 - Efficient in-memory integration of external mesh database
 - Abstract interfaces to components (mesh, solvers, assembly, analysis tools)
 - Requirements imposed on external tools through generic interfaces
- **Fully representative of a typical advanced implicit, unstructured finite element application**
 - Trilinos supplies components: modern linear & nonlinear solvers, preconditioning strategies, continuation tools, ...
 - Genericism of physics evaluation, residual based, with Sacado Jacobian and matrix free options
 - Parallel MPI(+X)
 - Large problems, representative boundary conditions, test suite 230+ problems
 - Embedded SA & UQ
 - Large problems, representative boundary conditions
 - Open source

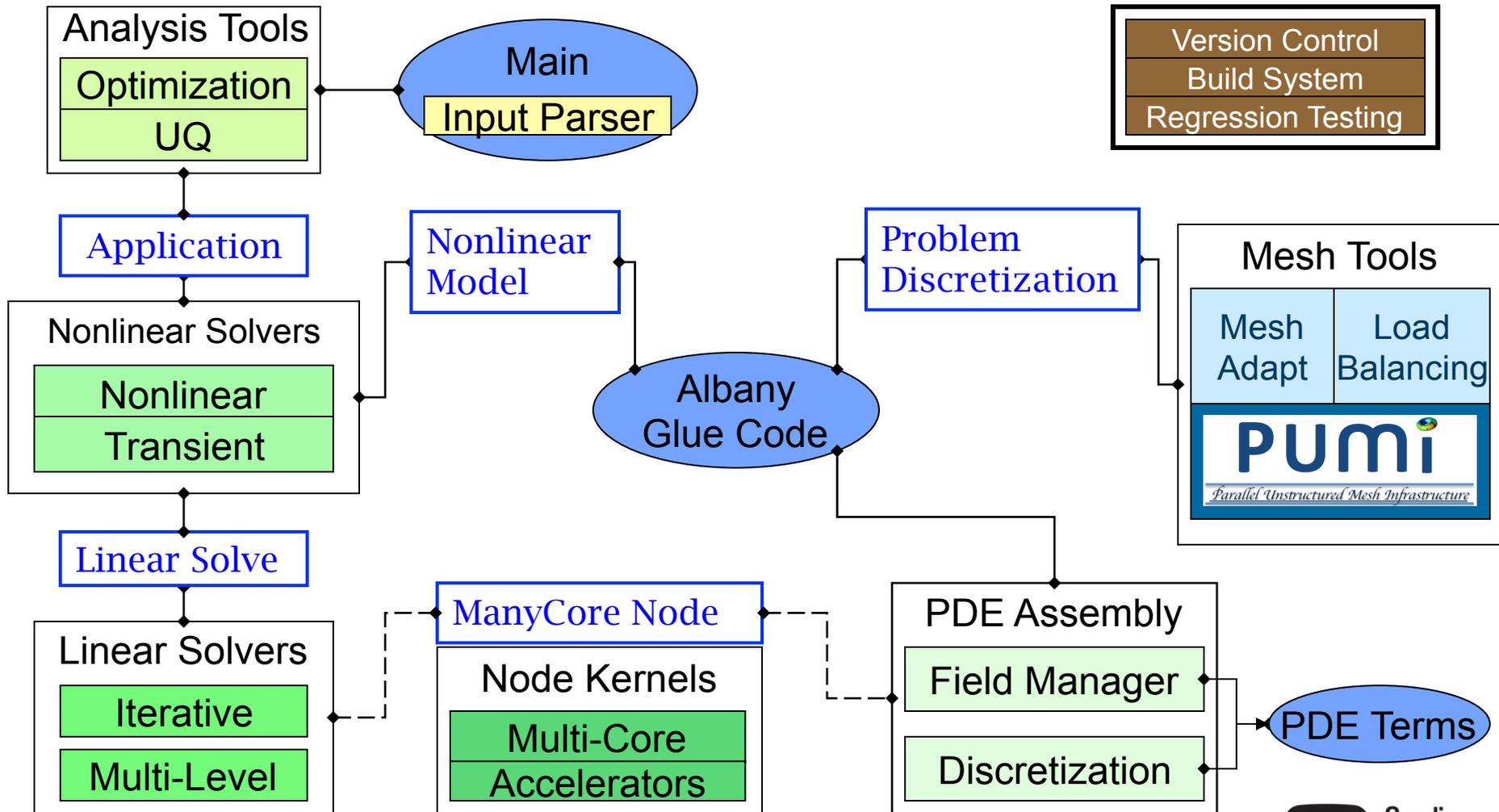
Albany Architecture

Software Quality Tools

Libraries

Interfaces

Existing Apps



Integration process

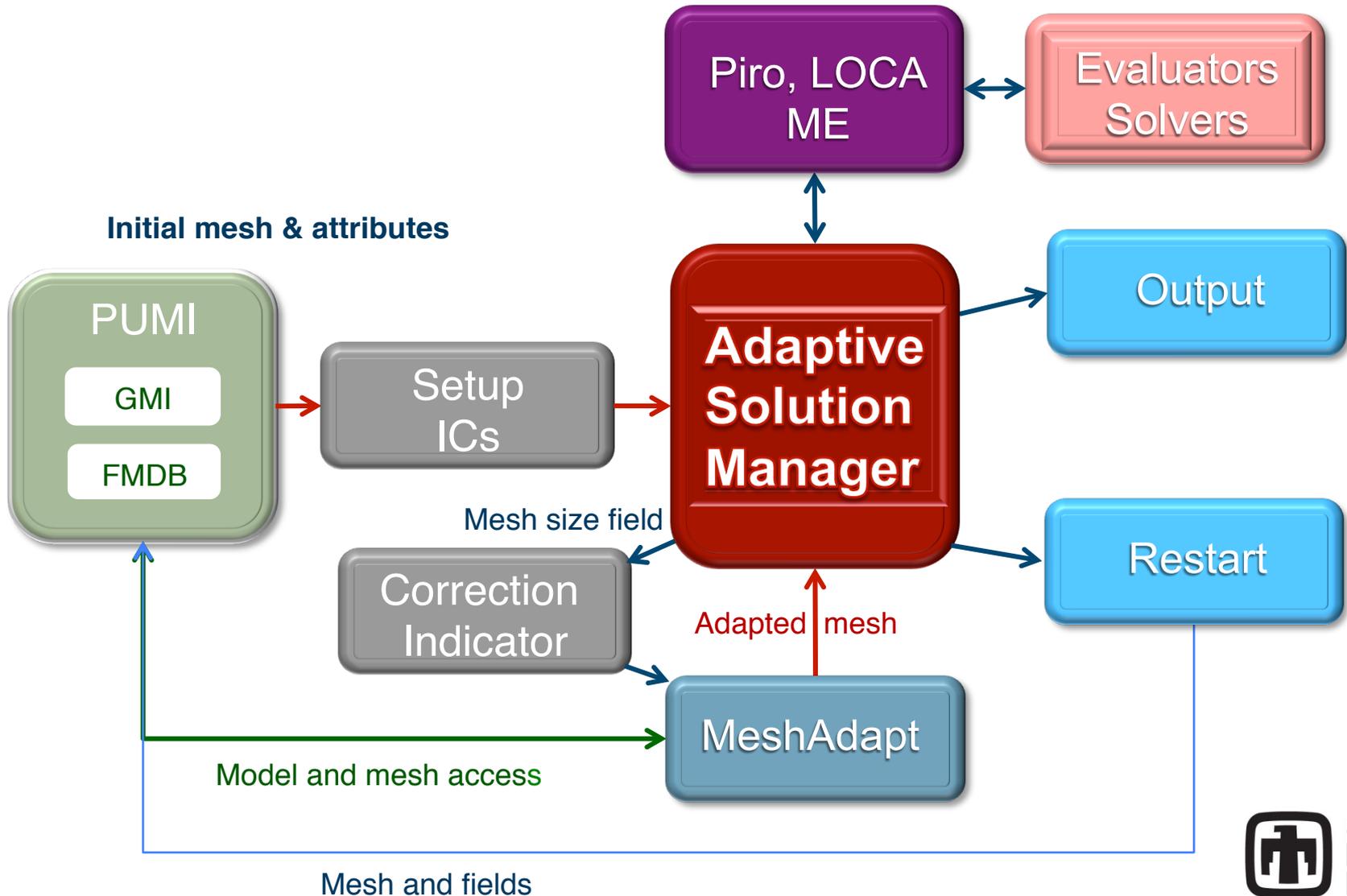
- **Restructure Albany to resize all mesh-dependent data structures**
- **Develop generic interface to SCOREC PUMI mesh database and adaptation tools**
 - Efficient, direct link to PUMI mesh infrastructure
 - Create specialized LOCA stepper and Piro solution manager classes to interface with adaptation libraries and solution transfer classes
- **Develop and demonstrate increasingly more sophisticated adaptation capabilities**



Discretization components

- **PUMI – Parallel Unstructured Mesh Infrastructure**
- **GMI – Geometric Modeling Infrastructure**
 - Parallel model representation
- **FMDB – distributed mesh database**
 - Entity containers, access functions
- **meshAdapt**
- **PCU**
 - Parallel communication utilities
- **APF**
 - Physics fields, solution transfer

Workflow

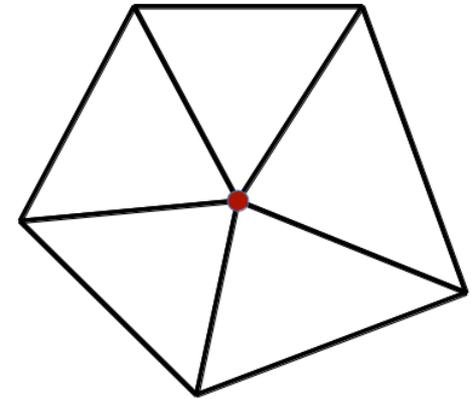


- **AdaptiveStepper** calls concrete instance of **AbstractAdapter::queryAdaptationCriteria()** – returns true if adaptation is needed
- **AdaptiveStepper** calls concrete instance of **AbstractAdapter::adaptMesh()** to execute the PUMI mesh adaptation process
 - concrete instance is templated on a "size field" object. The size field evaluation calculates the desired size of each element based on the solution state
- **adaptMesh()** then calls PUMI **meshadapt** to modify the mesh to satisfy the size field

Adaptation based on error estimation

Given a $(p-1)$ order state variable σ (e.g. Cauchy stress) at integration points

1. Construct an appropriate patch of elements
2. Recover a p -order field σ^* via a least squares fit to integration point data σ over the element patch
3. Repeat steps 1 and 2 for all element patches in the mesh
4. Integrate norms of p -order error field of $e = \sigma - \sigma^*$ over the whole mesh
5. Compute an appropriate mesh size field based on the estimated error

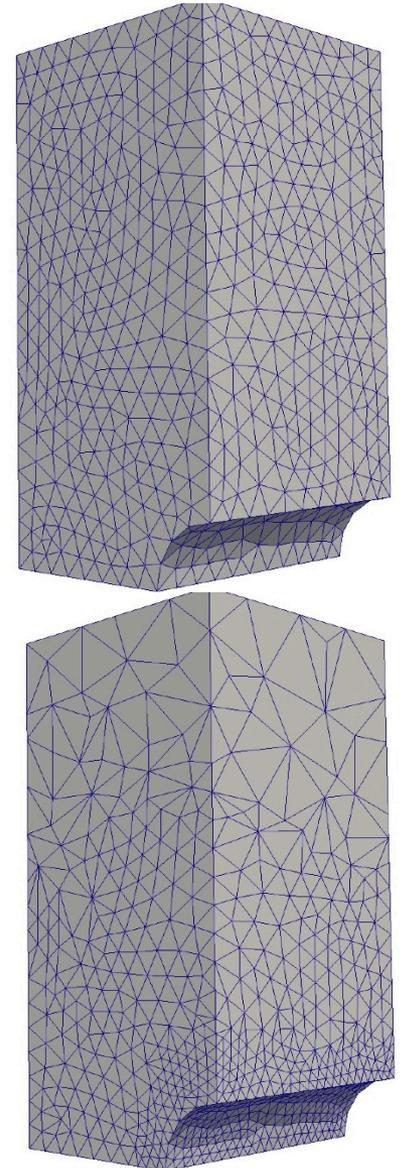


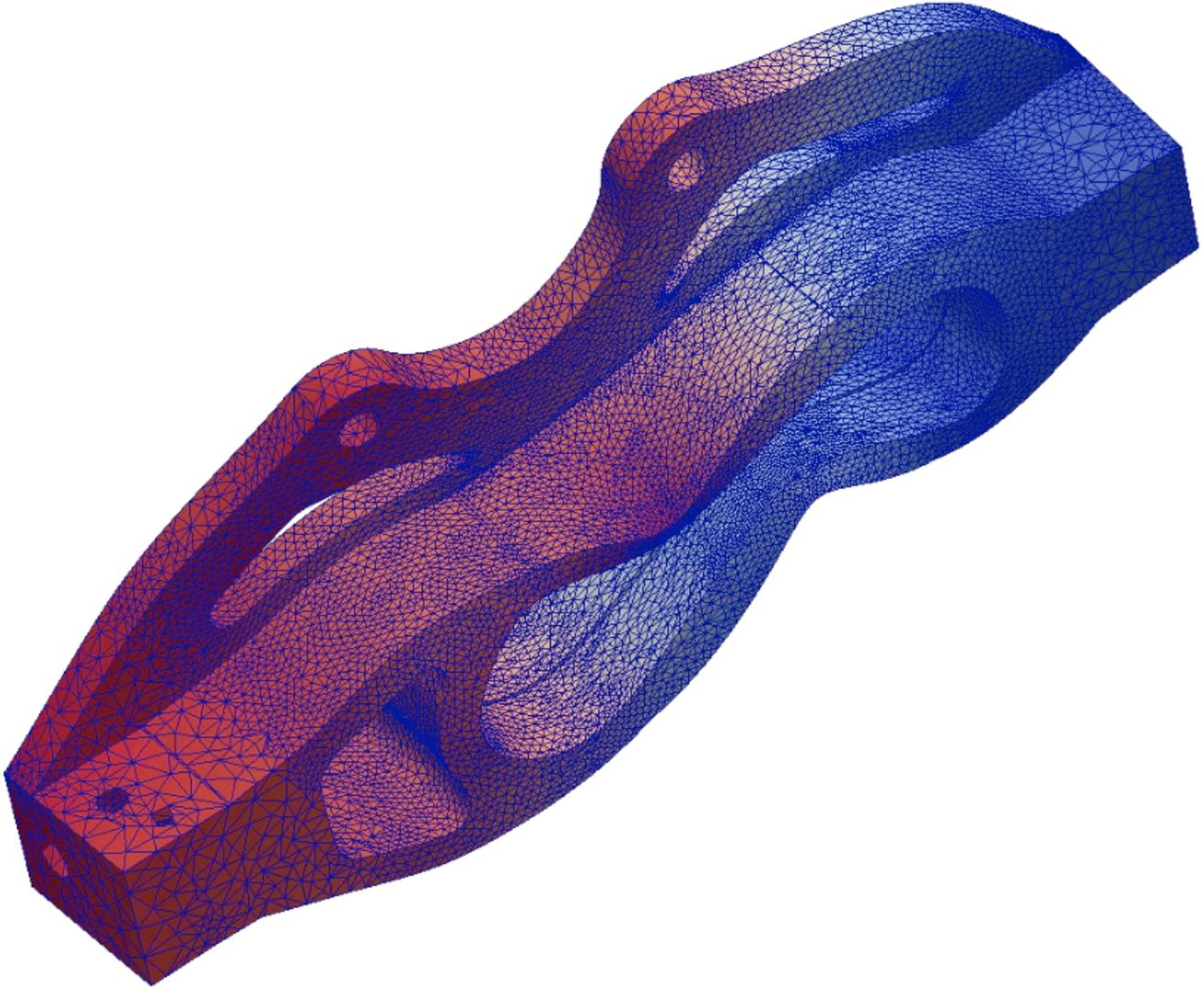
A 2D element patch

- **An element or node field that specifies what the local mesh size should be, as a scalar radius (isotropic) or vector (anisotropic)**
- **Two available approaches**
 - calculate size field in template class to concrete AbstractAdapt object given solution and mesh fields, or
 - ElementSizeField evaluator can calculate size field as an element, QP, or nodal field as a response
- **StateManager manages the resulting fields**
 - NodalDataBlock used for cross-workset and interprocessor consistency of nodal data fields
 - NodalDataBlock accesses all adaptive discretizations through AbstractDiscretization base class

Adaptation process

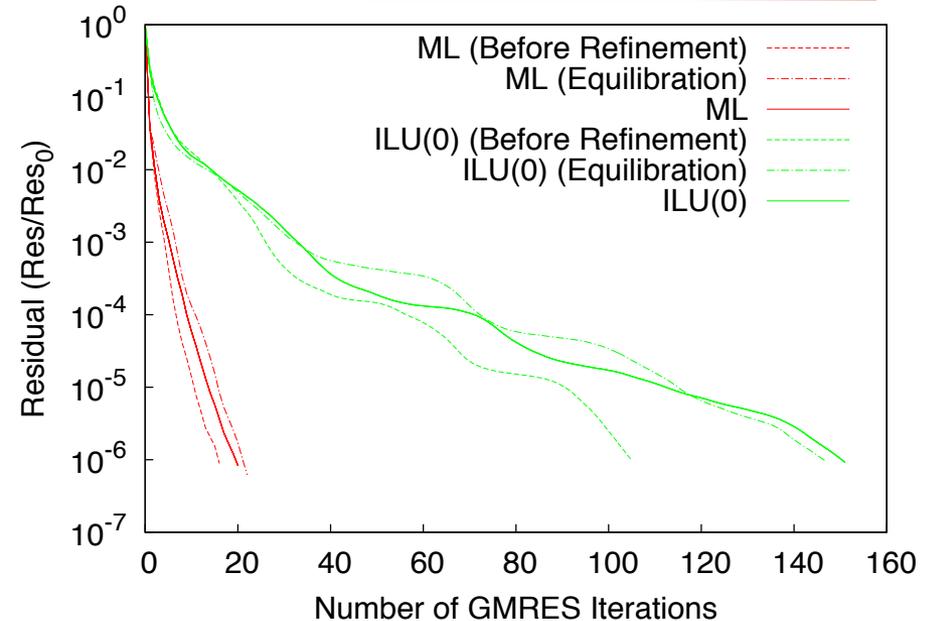
- PUMI meshadapt to modifies the mesh to satisfy the size field
- PUMI then load balances the adapted mesh across the available processors
- Locally, PUMI estimates the nodal solution at the locations where new nodes are added, and passes the modified nodal field data back to Albany
 - AbstractAdapter provides virtual solutionTransfer() method
- Finally, the concrete version of AbstractAdapter::adaptMesh() returns control to the LOCA stepper to begin the equilibration step



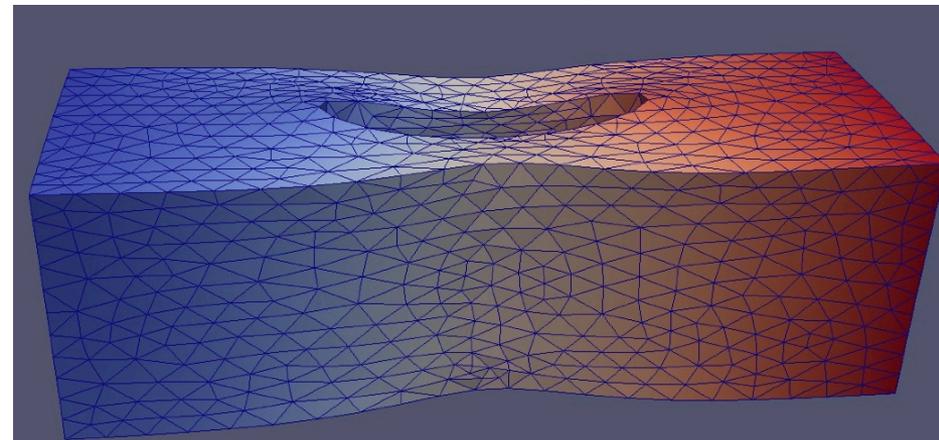


ML rigid body modes

- Null space data structures are resized and repopulated prior to LOCA equilibration step, given the new nodal coordinate data from PUMI
- The equilibration step is designed to "adjust" new interpolated values to satisfy equilibrium conditions, without advancing the problem state, to reduce overall error
- LOCA then resumes the stepping process



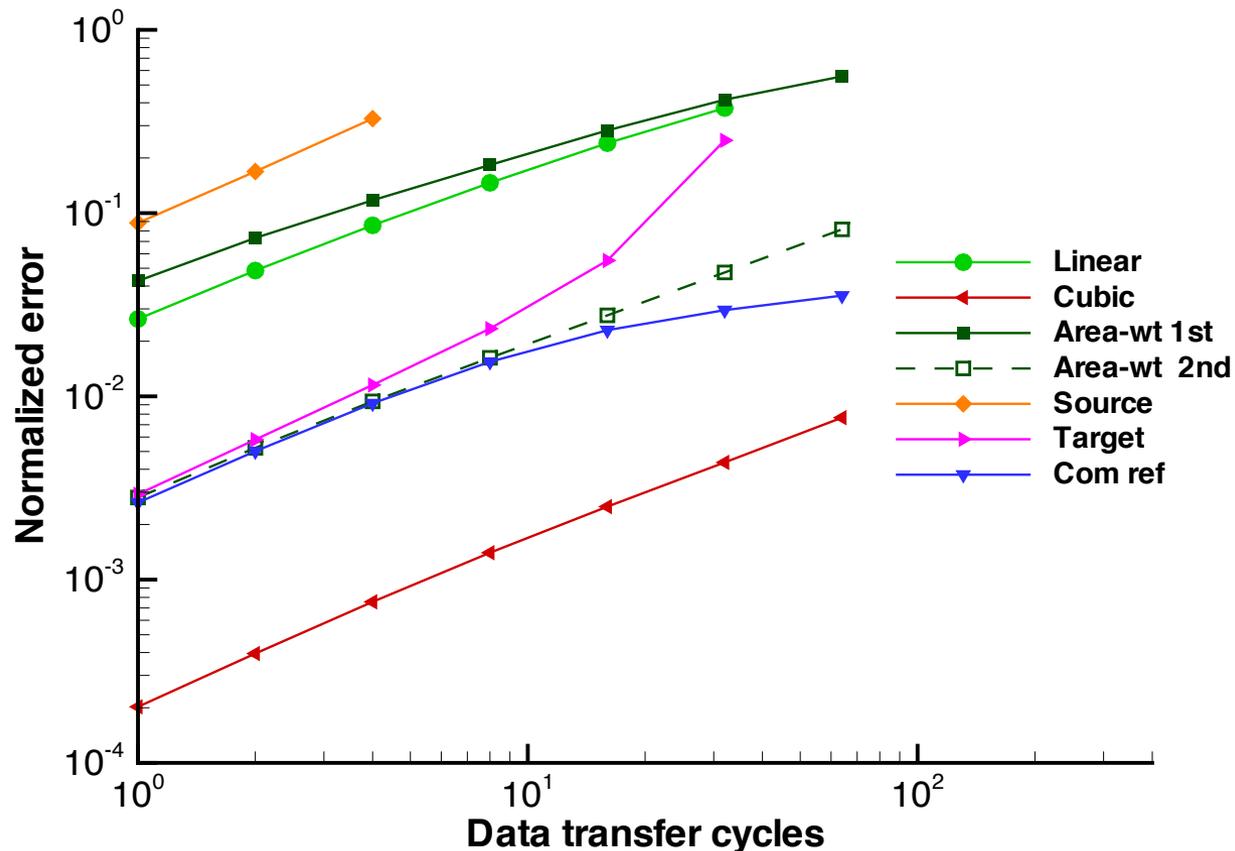
Bar deformation problem



- **As the mesh adapts, one must estimate the values of the nodal fields at the location of new node points, and the values of QP fields at the integration points of new (or modified) elements**
 - Some state quantities cannot be transferred using interpolation and must be treated using Lie Algebra Formalism (c.f. Jake Ostien's talk)
- **It is usually important to ensure that the estimated field values satisfy the conservation laws being employed (i.e., the transfer method is conservative) without introducing new maxima or minima not present in the original fields**
- **One may choose to employ an "equilibration" or "relaxation" process to adjust the estimated values to provide equilibrium with the fields on the existing mesh entities**
 - LOCA provides an equilibration step after each adaptation operation

Error accumulation with different transfer strategies

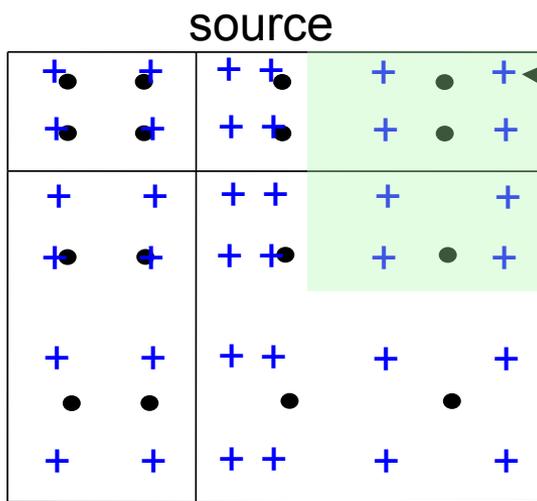
Comparison of errors inherent in various solution transfer schemes



- **Area, Source, Com ref are conservative**
- **Cubic spline is "accurate" but not conservative – fundamentally structured**
- **Accuracy of a given method is dependent on the nature of the fields being transferred**

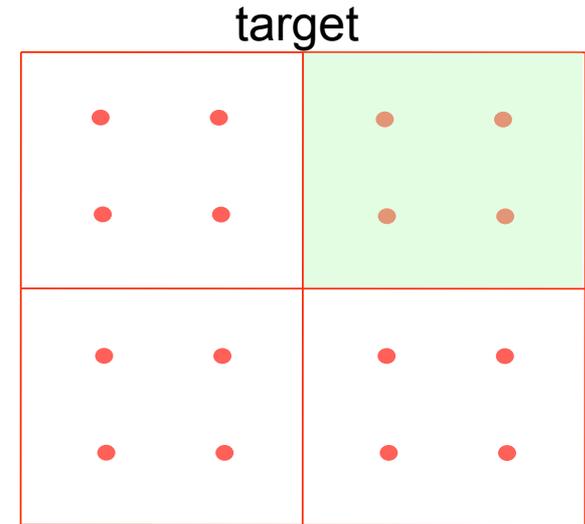
Common refinement solution transfer

(Jiao and Heath, IJNME 2004)



$$M_{ij} = \int_{\Omega} \psi_i \psi_j d\Omega$$

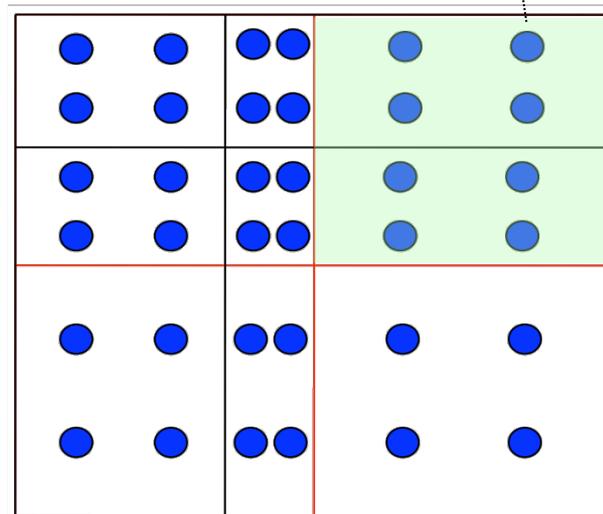
Get source f at CR
qp location $\rightarrow f_k$



$$f = \sum_{i=1}^m f_i \phi_i$$

ϕ_i integrates the
source elements

$$M_{ij} g_i = \sum_{\text{CR}} f_k \Phi_k$$



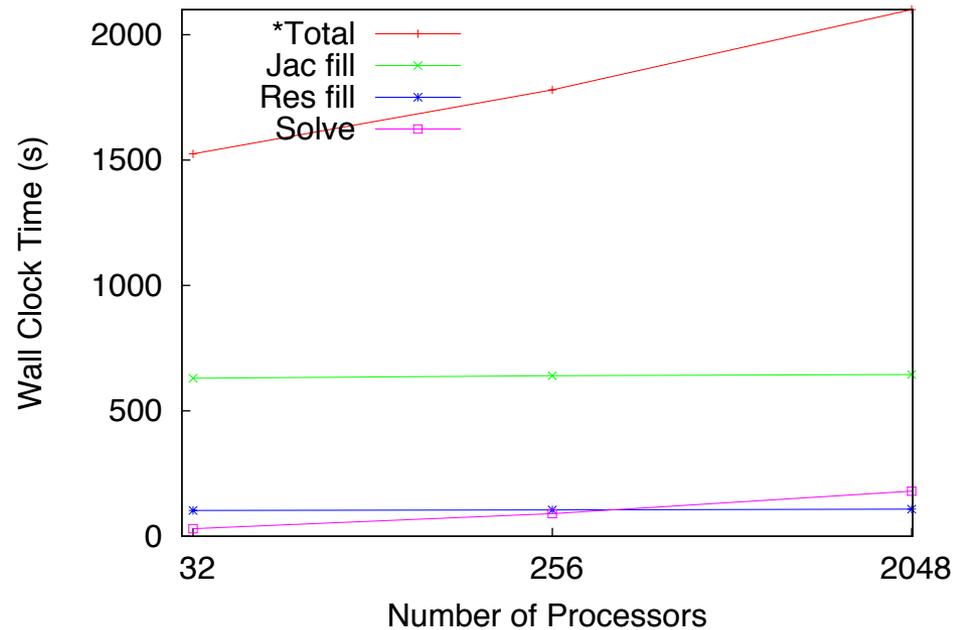
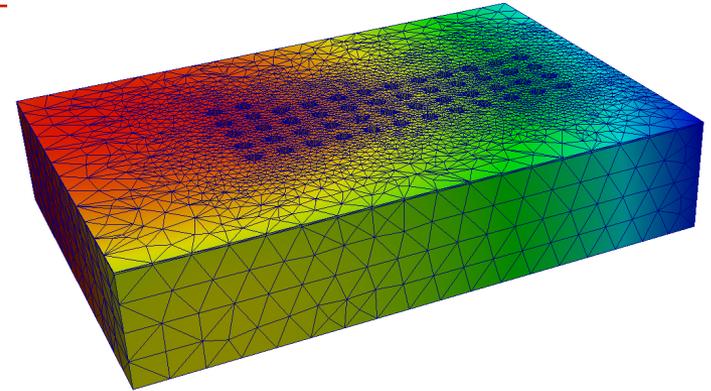
Φ_k integrates the CR elements

$$g = \sum_{i=1}^n g_i \psi_i$$

ψ_i integrates the
target elements

RPI integrated circuit wafer analysis

- 500-channel wafer
- 300K elements per processor
- *Total time ignores time spent in I/O



ML parameter settings

```
<ParameterList name="ML Settings">
  <Parameter name="default values" type="string" value="SA"/>
  <Parameter name="aggregation: type" type="string" value="MIS"/>
  <Parameter name="aggregation: damping factor" type="double" value="0.0"/>
  <Parameter name="prec type" type="string" value="full-MGV"/>
  <Parameter name="max levels" type="int" value="4"/>
  <Parameter name="repartition: enable" type="int" value="1"/>
  <Parameter name="repartition: Zoltan dimensions" type="int" value="3"/>
  <Parameter name="repartition: min per proc" type="int" value="1000"/>
  <Parameter name="smoother: type" type="string" value="Chebyshev"/>
  <Parameter name="smoother: sweeps" type="int" value="3"/>
  <Parameter name="smoother: Chebyshev alpha" type="double" value="50"/>
  <Parameter name="smoother: pre or post" type="string" value="both"/>
  <Parameter name="coarse: type" type="string" value="Amesos-Superludist"/>
  <Parameter name="coarse: max size" type="int" value="1500"/>
  <Parameter name="PDE equations" type="int" value="3"/>
</ParameterList>
```

Home Projects Help

Albany at RPI

Overview Activity Issues News Documents **Wiki** Repository

- Installing Required Libraries for Trilinos
- Installing Trilinos
- Installing Albany
- Installing Trilinos and Albany on BG/Q
- Source Control with Git
- Intrepid Cubature
- Example Workflow
- Albany Tutorial at RPI

Build and test

Detailed build instructions at <http://redmine.scorec.rpi.edu/projects/albany-rpi/wiki>

Trilinos: `git clone https://software.sandia.gov/trilinos/repositories/publicTrilinos`

Albany: `git clone https://software.sandia.gov/albany/repositories/Albany.git`

SCOREC: <http://www.scorec.rpi.edu/~cwsmith/FASTMath/pumiSCOREC.tar.gz>

Cdash site: <http://my.cdash.org/index.php?project=Albany>

| Albany | | | | | | | | | | |
|-------------|-----------|---------|------|-------|---------|------|---------|------|------|---------------------|
| Project | | | | | | | | | | |
| Project | Configure | | | Build | | | Test | | | Last submission |
| | Error | Warning | Pass | Error | Warning | Pass | Not Run | Fail | Pass | |
| Albany | 0 | 1 | 2 | 0 | 0 | 3 | 0 | 0 | 181 | 2014-02-21 00:55:54 |
| SubProjects | | | | | | | | | | |
| Project | Configure | | | Build | | | Test | | | Last submission |
| | Error | Warning | Pass | Error | Warning | Pass | Not Run | Fail | Pass | |
| AlbanySrc | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 181 | 2014-02-21 05:55:54 |
| Trilinos | 0 | 1 | 1 | 0 | 0 | 1 | | | | 2014-02-21 05:23:36 |
| SCOREC | | | | 0 | 0 | 1 | | | | 2014-02-21 05:10:16 |

- **Modify solution data structures to employ an allocation with reserve strategy**
- **Higher order elements**
 - 10-node composite tets
 - Field manager allocation and reallocation to support cell topology and data layout changes
- **Demonstrate solution transfer for integration point quantities**
- **Develop concrete RythmosAdaptiveStepper providing similar functionality to AdaptiveStepper**



References

- X. Jiao and M. T. Heath, ***Common-refinement-based data transfer between nonmatching meshes in multiphysics simulations***, International Journal for Numerical Methods in Engineering, 61(14):2402-2427, (2004)
- R.W. Johnson, G. Hansen, and C. Newman, ***The role of data transfer on the selection of a single vs. multiple mesh architecture for tightly coupled multiphysics applications***, Applied Mathematics and Computation, 217(22): 8943-8962, (2011)
- Seol, E.S. and Shephard, M.S., ***Efficient distributed mesh data structure for parallel automated adaptive analysis***, Engineering with Computers, 22(3-4): 197-213, (2006)
- Alauzet, F., Li, X., Seol, E.S. and Shephard, M.S., ***Parallel Anisotropic 3D Mesh Adaptation by Mesh Modification***, Engineering with Computers, 21(3):247-258, (2006)
- Li, X., Shephard, M.S. and Beall, M.W., ***3-D Anisotropic Mesh Adaptation by Mesh Modifications***, Comp. Meth. Appl. Mech. Engng., 194(48-49):4915-4950, (2005)