Building Hierarchical Toolchains for Nonlinear Analysis

Roger Pawlowski, Eric Phipps, Andrew Salinger
Sandia National Laboratories

Roscoe Bartlett
Oak Ridge National Laboratory

Trilinos User Group Meeting
Wednesday, November 2nd, 2011

SAND2011-8253C

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.
Embedded Nonlinear Analysis Capability Area

- **Basic Capabilities:**
  - TBGP Automatic Differentiation (Sacado)
  - (Globalized) Nonlinear solution methods (NOX)
  - Time Integration (Rythmos)

- **Advance Analysis Capabilities:**
  - (Multi-)Parameter Continuation (LOCA)
  - Stability analysis (LOCA)
  - Bifurcation analysis (LOCA)
  - Optimization (Aristos/ROL, MOOCHO, TriKota/DAKOTA)
  - Uncertainty Quantification (Stokhos TriKota/DAKOTA)

- **Analysis beyond direct simulation:**
  - Often a simple direct solve is not enough
  - Automate computational tasks that are often performed by application code users by trial-and-error or repeated simulation
Agile Components Code Design: Albany, Panzer/Drekar, Peridigm

Analysis Tools
- Optimization
- UQ

Application

Solvers w/ Sensitivities
- Nonlinear
- Transient
- Stochastic Galerkin

Nonlinear Model

Application

Physics Model

Linear Solve

Linear Solvers / Preconditioners
- Iterative
- Block Iterative
- Direct
- Eigensolve
- Domain Decomp
- MultiLevel
- SchurComp

ManyCore Node Kernels

Multit-Core Accelerators

Field Manager

Field Evaluators

Meshing I/O
- DataContainers
- Hand-Coded:
- Inline Mesher
- Quality Improvement
- Load Balancing

PDE Assembly is Templated for AD, PCE

Discretization

SchurComp
This Talk

Trilinos Interfaces and Components for the Analysis Layers

Piro

Dakota
OptiPack

R. O. Model Evaluator

Phalanx

PDE Assembly is Templated for Sacado, Stokhos

Intrepid, Shards

Application

Model Evaluator

NOX
Rythmos
LOCA
Stokhos

Thyra
LOWS
Stratimikos

Phipp’s Talk

PDE Assembly is Templated for Sacado, Stokhos

Phalanx

PDEs in Phalanx

ntrepid, Shards
General Physics Model


\[ f(\dot{x}, x, \{p_l\}, t) = 0 \]

\( x \in \mathbb{R}^{n_x} \) is the vector of state variables (unknowns being solved for),
\( \dot{x} = \frac{\partial x}{\partial t} \in \mathbb{R}^{n_x} \) is the vector of derivatives of the state variables with respect to time,
\( \{p_l\} = \{p_0, p_1, \ldots, p_{N_p-1}\} \) is the set of \( N_p \) independent parameter sub-vectors,
\( t \in [t_0, t_f] \in \mathbb{R}^1 \) is the time ranging from initial time \( t_0 \) to final time \( t_f \),

\[ g_j(\dot{x}, x, \{p_l\}, t) = 0, \text{ for } j = 0, \ldots, N_g - 1 \]

\[ g_j(\dot{x}, x, \{p_l\}, t) : \mathbb{R}^{(2n_x + (\sum_{l=0}^{N_p-1} n_{p_l}) + 1)} \rightarrow \mathbb{R}^{n_{g_j}} \] is the \( j \)th response function.

- Input Arguments: state time derivative, state, parameters, time
- Output Arguments: Residual, Jacobian, response functions, etc…
Model Evaluator: Thyra and EpetraExt Versions

Nonlinear ANA

**Thyra::ModelEvaluator**
- `createInArgs() : InArgs`
- `createOutArgs() : OutArgs`
- `create_W() : LinearOpWithSolveBase`
- `create_W_op() : LinearOpBase`
- ...
- `evalModel(in InArgs, out OutArgs)`

- Common interface for ANAs
  - Residuals, Jacobians, parameters, parameter sensitivities, response functions, stochastic Residuals/Jacobians
- Stateless model (All state passed in as parameters)
- Allows for efficient multiple shared calculations (e.g. automatic differentiation)
- Inputs and Outputs are extensible without requiring changes to user code
ModelEvaluator and Response Only ModelEvaluator

**Analysis Tools**
- (black-box)
  - Optimization
  - UQ (sampling)
- (embedded)
  - Time Integration
  - Nonlinear Solver

**InArgs**
- I can make use of: $p$
- I can compute: $g$, $\frac{dg}{dp}$

**OutArgs**
- I can make use of: $p$
- I can compute: $g$, $\frac{dg}{dp}$

**Mathematical Formulas**

- $f(x, p) = 0$
- $\frac{dg}{dp} = \frac{\partial g}{\partial x}^T J^{-1} \frac{df}{dp} + \frac{\partial g}{\partial p}$
- $W = \alpha \frac{df}{dx} + \beta \frac{df}{dx}$
- $f(\dot{x}, x, p, t) = 0$

**Symbols**
- $f$ = residual
- $x$ = solution vec
- $p$ = parameters (properties)
- $g$ = responses

**Diagram Notes**
- Piro::NOXSolver
- Local Fill
Concept

• Use **inheritance** and **composition** to wrap analysis tools as model evaluators to build a hierarchical chain.

• Model Evaluator Use Cases:
  1. Application Interface
  2. PIRO “Response Only Model Evaluators” with response sensitivities:
     • Nonlinear (NOX),
     • Time Integrator (Rythmos),
     • Optimization (MOOCHO), Param.
     • Continuation/Stability/Bifurcation (LOCA)
  3. Decorators:
     • Default Implementation (DelegatorBase)
     • Scaled
     • Jacobian-Free Newton-Krylov (JFNK)
     • Block Composite (LIME Multiphysics)
Uses **Decorator** to better condition a poorly scaled system of equations.
PIRO ROMEs Add direct support to Nonlinear Analysis Tools and Response Sensitivities

Analysis

\[ p \rightarrow g(p) \]

Piro::NOXSolver

\[ f(x, p) = 0 \]
\[ \frac{dg}{dp} = \frac{\partial g^T}{\partial x} J^{-1} \frac{df}{dp} + \frac{\partial g}{\partial p} \]

Piro::RythmosSolver
Piro::MOOCOSolver
Piro::LOCASolver
Piro::Analysis (Dakota)

Difficult Solve

Main()

Piro::Analysis (Dakota)

Main()

Piro::NOXSolver

NOX::Solver

Thyra::Scaled Model Evaluator

Application's Model Evaluator

Physics set: e.g. Quantum in Albany

Physics set: e.g. Quantum in Albany
**Embedded** UQ can be inserted as a ME Decorator

Basic Solve

- **Main()**
- **Piro::NOXSolver**
- **Application’s Model Evaluator**
- **Physics set:** e.g. MHD in Drekar

Embedded UQ

- **Main()**
- **Piro::NOXSolver**
- **Stokhos::ModelEvaluator**
- **Application’s Model Evaluator**
- **Physics set:** e.g. MHD in Drekar

Stokhos forms a block composite system

Each point is a block corresponding to a basic solve Jacobian
Decorators and multi-physics solvers grow the capabilities with generic implementations

Nonlinear-Elimination Solver (LIME, Piro)

Piro::NOXSolver

Piro::LOCASolver

Application’s Model Evaluator

Physics set #1:

Piro::MatrixFree Decorator

Application’s Model Evaluator

Physics set #2: Residual Only

JFNK implemented as a decorator ME, implements: create_W_op()

\[ Jv \approx \frac{F(x + \delta v) - F(x)}{\delta} \]

Multiphysics coupling examples:
• CASL: CFD/Neutronics/Plant Balance
• QCAD: Coupled Schrodinger-Poisson (nonlinear solve coupled to eigensolve)

6 Model Evaluators in 1 run
Let Go Crazy!

Main() → Piro::Analysis(Dakota) (Optimization) → LIME::NonlinearElimination ModelEvaluator → PIRO::NOXSolver → Stokhos::ModelEvaluator → Thyra::Scaled Model Evaluator → LIME::BlockCompositeME → Application’s Model Evaluator

PIRO::LOCA → Application’s Model Evaluator → RELAP5 Plant Balance

Drekar::CFD

Denovo::Neutronics

\[
f = \begin{bmatrix} f_{RELAP} \\ Jn \end{bmatrix} = \begin{bmatrix} 1 - \phi^T \cdot n \end{bmatrix}
\]
What’s Missing?

• LIME Multiphysics Coupling Library
  – LIME 1.0 internal to Sandia/CASL, hard coded to Epetra data structures

• Stochastic support in Thyra::ModelEvaluator
  – Currently only implemented in EpetraExt::ModelEvaluator
  – Help Eric!
Current and Future Efforts

• Update Thyra::ModelEvaluator
  • *Many capabilities are EpetraExt-only*
  • “Ripen” Tpetra Adapters to Thyra implementations
• Refactor/Expansion of Model Evaluator interface
  • Usability
  • expand in/out args
  • handling of statefulness
  • usability (e.g. selection of parameters by string)
  • adaptivity-enabled (reset maps / vector spaces)
• Grow library of ME capabilities
  • PIRO
  • LIME 2.0
  • Decorators
• System UQ (Phipps, Wildey)
Extra Slides
# Software Integration Models

Inputs and outputs are *optionally* supported by physics model → restricts allowed solution procedures

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
<th>Required Inputs</th>
<th>Required Outputs</th>
<th>Optional Outputs</th>
<th>Time Integration Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response Only Model (Coupling Elimination)</td>
<td>$p \rightarrow g(p)$</td>
<td>$p$</td>
<td></td>
<td>$g$</td>
<td>Internal</td>
</tr>
<tr>
<td>State Elimination Model</td>
<td>$p \rightarrow x(p)$</td>
<td>$p$</td>
<td>$x$</td>
<td>$g$</td>
<td>Internal</td>
</tr>
<tr>
<td>Fully Implicit Time Step Model</td>
<td>$f(x, p) = 0$</td>
<td>$x, p$</td>
<td></td>
<td>$f$</td>
<td>Internal</td>
</tr>
<tr>
<td>Transient Explicitly Defined ODE Model</td>
<td>$\dot{x} = f(x, p, t)$</td>
<td>$x, p, t$</td>
<td></td>
<td>$f$</td>
<td>External</td>
</tr>
<tr>
<td>Transient Fully Implicit DAE Model</td>
<td>$f(\dot{x}, x, p, t) = 0$</td>
<td>$\dot{x}, x, p, t$</td>
<td></td>
<td>$f$</td>
<td>External or Internal</td>
</tr>
</tbody>
</table>

$$W = \alpha \frac{\partial f}{\partial \dot{x}} + \beta \frac{\partial f}{\partial x}$$

$$M = \text{preconditioner}$$
### Some Examples of Nonlinear Analysis Supported by ModelEvaluator

<table>
<thead>
<tr>
<th><strong>Nonlinear equations:</strong></th>
<th>Solve $f(x) = 0$ for $x \in \mathbb{R}^n$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Stability analysis:</strong></td>
<td>For $f(x, p) = 0$ find space $p \in \mathcal{P}$ such that $\frac{\partial f}{\partial x}$ is singular</td>
</tr>
<tr>
<td><strong>Explicit ODEs:</strong></td>
<td>Solve $\dot{x} = f(x, t) = 0$, $t \in [0, T]$, $x(0) = x_0$, for $x(t) \in \mathbb{R}^n, t \in [0, T]$</td>
</tr>
<tr>
<td><strong>DAEs/Implicit ODEs:</strong></td>
<td>Solve $f(\dot{x}(t), x(t), t) = 0$, $t \in [0, T]$, $x(0) = x_0$, $\dot{x}(0) = x'_0$, for $x(t) \in \mathbb{R}^n, t \in [0, T]$</td>
</tr>
<tr>
<td><strong>Explicit ODE Forward Sensitivities:</strong></td>
<td>Find $\frac{\partial x}{\partial p}(t)$ such that: $\dot{x} = f(x, p, t) = 0$, $t \in [0, T]$, $x(0) = x_0$, for $x(t) \in \mathbb{R}^n, t \in [0, T]$</td>
</tr>
<tr>
<td><strong>DAE/Implicit ODE Forward Sensitivities:</strong></td>
<td>Find $\frac{\partial x}{\partial p}(t)$ such that: $f(\dot{x}(t), x(t), p, t) = 0$, $t \in [0, T]$, $x(0) = x_0$, $\dot{x}(0) = x'_0$, for $x(t) \in \mathbb{R}^n, t \in [0, T]$</td>
</tr>
<tr>
<td><strong>Unconstrained Optimization:</strong></td>
<td>Find $p \in \mathbb{R}^m$ that minimizes $g(p)$</td>
</tr>
<tr>
<td><strong>Constrained Optimization:</strong></td>
<td>Find $x \in \mathbb{R}^n$ and $p \in \mathbb{R}^m$ that: minimizes $g(x, p)$ such that $f(x, p) = 0$</td>
</tr>
<tr>
<td><strong>ODE Constrained Optimization:</strong></td>
<td>Find $x(t) \in \mathbb{R}^n$ in $t \in [0, T]$ and $p \in \mathbb{R}^m$ that: minimizes $\int_0^T g(x(t), p)$ such that $\dot{x} = f(x(t), p, t) = 0$, on $t \in [0, T]$ where $x(0) = x_0$</td>
</tr>
</tbody>
</table>
Nonlinear Algorithms and Applications: Thyra & Model Evaluator!

Nonlinear ANA Solvers in Trilinos

- NOX / LOCA
- Rythmos
- MOOCHO
- Trilinos and non-Trilinos Preconditioner and Linear Solver Capability

Sandia Applications

- Xyce
- Charon
- Tramonto
- Aria
- Panzer

Key Points

- Provide single interface from nonlinear ANAs to applications
- Provide single interface for applications to implement to access nonlinear ANAs
- Provides shared, uniform access to linear solver capabilities
- Once an application implements support for one ANA, support for other ANAs can quickly follow