SAND2012-3448P

Enabling Embedded Algorithms R&D Through Template-based Generic Programming

Eric Phipps

Optimization and Uncertainty Quantification Department 1441

CIS External Panel Review

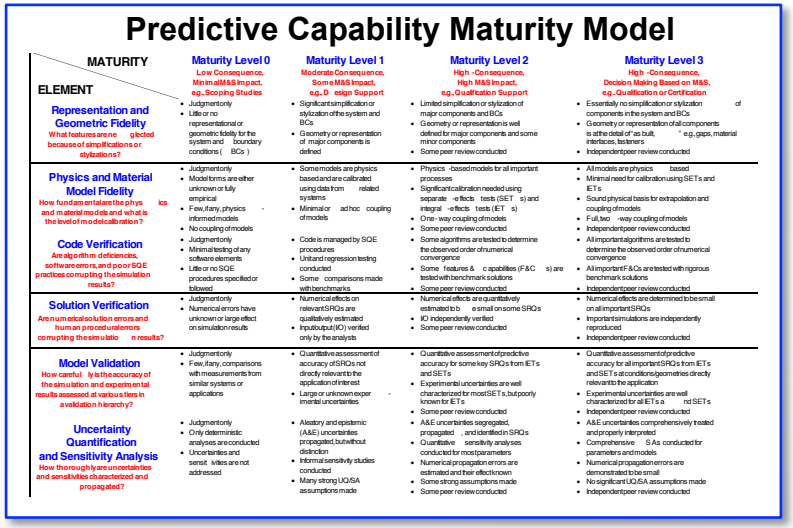
May 8-10, 2012

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.

Sandia National Laboratories

Embedded Algorithms are Critical for Mission Success

- V&V&UQ are crucial to today's mission
  - Foundation of predictive simulation
- Single-point forward simulations are insufficient
  - Sensitivities
  - Error estimates
  - Probability distributions
  - Epistemic uncertainties
- Embedded algorithms are critical to achieving success
  - Predictive science demands of complex, multiscale physics
  - Exascale

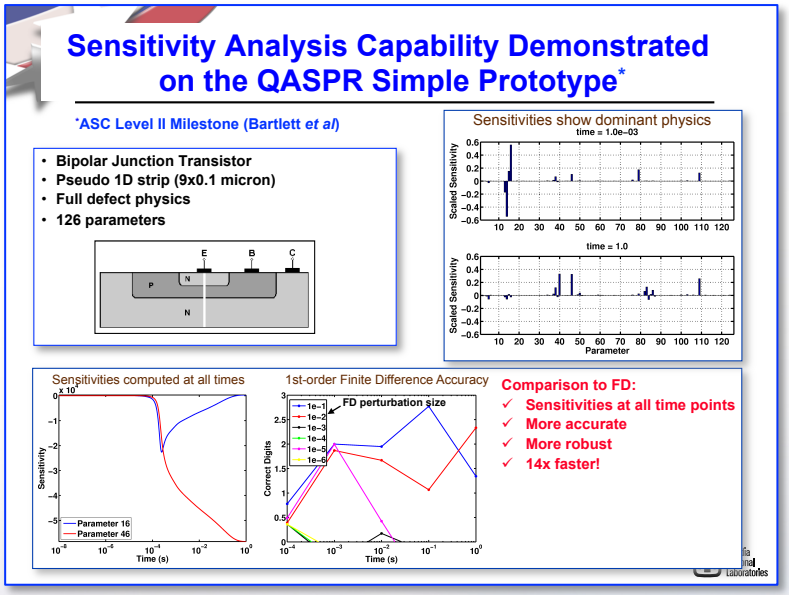


Predictive Capability Maturity Model

MATURITY / ELEMENT	Maturity Level 0 Low Consequence, Minimal M&S Impact, eg., Scoping Studies	Maturity Level 1 Moderate Consequence, Some M&S Impact, eg., Design Support	Maturity Level 2 High -Consequence, High M&S Impact, eg., Qualification Support	Maturity Level 3 High -Consequence, Decision Making Based on M&S, eg., Qualification or Certification
Representation and Geometric Fidelity What features are neglected because of simplifications or stylizations?	• Judgment only • Little or no representational or geometric fidelity for the system and boundary conditions (BCs)	• Significant simplification or stylization of the system and BCs • Geometry or representation of major components is defined	• Limited simplification or stylization of major components and BCs • Geometry or representation is well defined for major components and some minor components • Some peer review conducted	• Essentially no simplification or stylization of components in the system and BCs • Geometry or representation of all components is at the detail of "as built, " e.g., gaps, material interfaces, fasteners • Independent peer review conducted
Physics and Material Model Fidelity How fundamental are the physics and material models and what is the level of model calibration?	• Judgment only • Model forms are either unknown or fully empirical • Few, if any, physics - informed models • No coupling of models • Judgment only • Minimal testing of any software elements • Little or no SQE procedures specified or followed	• Some models are physics based and are calibrated using data from related systems • Minimal or ad hoc coupling of models • Code is managed by SQE procedures • Unit and regression testing conducted • Some comparisons made with benchmarks	• Physics -based models for all important processes • Significant calibration needed using separate -effects tests (SET s) and integral -effects tests (IET s) • One- way coupling of models • Some peer review conducted • Some algorithms are tested to determine the observed order of numerical convergence • Some features & capabilities (F&C s) are tested with benchmark solutions • Some peer review conducted	• All models are physics based • Minimal need for calibration using SETs and IETs • Sound physical basis for extrapolation and coupling of models • Full, two -way coupling of models • Independent peer review conducted • All important algorithms are tested to determine the observed order of numerical convergence • All important F&Cs are tested with rigorous benchmark solutions • Independent peer review conducted
Solution Verification Are numerical solution errors and human or procedural errors corrupting the simulation results?	• Judgment only • Numerical errors have unknown or large effect on simulation results	• Numerical effects on relevant SRQs are qualitatively estimated • Input/output (I/O) verified only by the analysts	• Numerical effects are quantitatively estimated to be small on some SRQs • I/O independently verified • Some peer review conducted	• Numerical effects are determined to be small on all important SRQs • Important simulations are independently reproduced • Independent peer review conducted
Model Validation How careful ly is the accuracy of the simulation and experimental results assessed at various tiers in a validation hierarchy?	• Judgment only • Few, if any, comparisons with measurements from similar systems or applications	• Quantitative assessment of accuracy of SRQs not directly relevant to the application of interest • Large or unknown experimental uncertainties	• Quantitative assessment of predictive accuracy for some key SRQs from IETs and SETs • Experimental uncertainties are well characterized for most SETs, but poorly known for IETs • Some peer review conducted	• Quantitative assessment of predictive accuracy for all important SRQs from IETs and SETs at conditions/geometries directly relevant to the application • Experimental uncertainties are well characterized for all IETs and SETs • Independent peer review conducted
Uncertainty Quantification and Sensitivity Analysis How thoroughly are uncertainties and sensitivities characterized and propagated?	• Judgment only • Only deterministic analyses are conducted • Uncertainties and sensitivities are not addressed	• Aleatory and epistemic (A&E) uncertainties propagated, but without distinction • Informal sensitivity studies conducted • Many strong UQ/SA assumptions made	• A&E uncertainties segregated, propagated, and identified in SRQs • Quantitative sensitivity analyses conducted for most parameters • Numerical propagation errors are estimated and their effect known • Some strong assumptions made • Some peer review conducted	• A&E uncertainties comprehensively treated and properly interpreted • Comprehensive SAs conducted for parameters and models • Numerical propagation errors are demonstrated to be small • No significant UQ/SA assumptions made • Independent peer review conducted

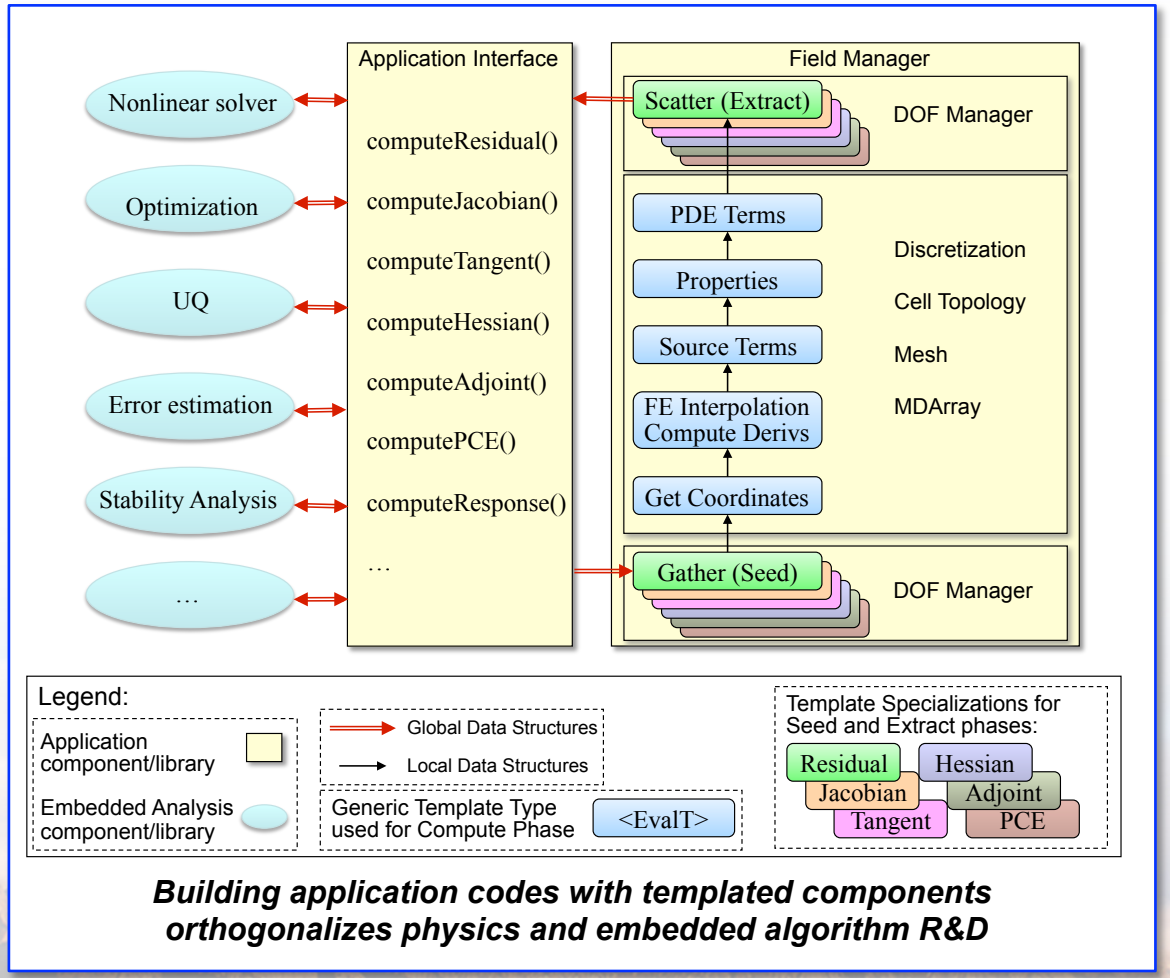
Addressing Embedded Algorithm Challenges through Templating & AD

- **Embedded algorithms leverage simulation structure**
  - Adjoint sensitivities/error estimates
  - Derivative-based optimization/stability analysis
  - Stochastic Galerkin or adjoint-based UQ methods
  - …
- **Incorporating directly requires significant effort**
  - Developers must understand algorithmic requirements
  - Limits embedded algorithm R&D impact
- **'06 and '08 external reviews**
  - Automatic differentiation (AD)
  - C++ templating of physics at "element" level
  - Operator overloading-based AD tools
- **Demonstrated success for QASPR project**
  - Efficient Jacobian evaluation without hand-coding
  - Rapid physics development
  - Forward sensitivity analysis



Unique Component Support of Embedded Algorithms via Templates

- **Templating on scalar type enables analysis abstraction**
  - **float -> forward simulation**
  - **FAD -> sensitivity**
  - **PCE -> probability**
  - **Fuzzy number -> epistemic**
  - ***Template-based generic programming***
- **Templating components provides API to support embedded algorithms**
  - **Developers focus on component implementation**
  - **Embedded quantities are easily incorporated**
- **Our support of embedded algorithms through templates is unique**
  - **Libmesh, deal.II, FEniCS, Sundance, MASA**
  - **Template approach is scalable to many embedded techniques**

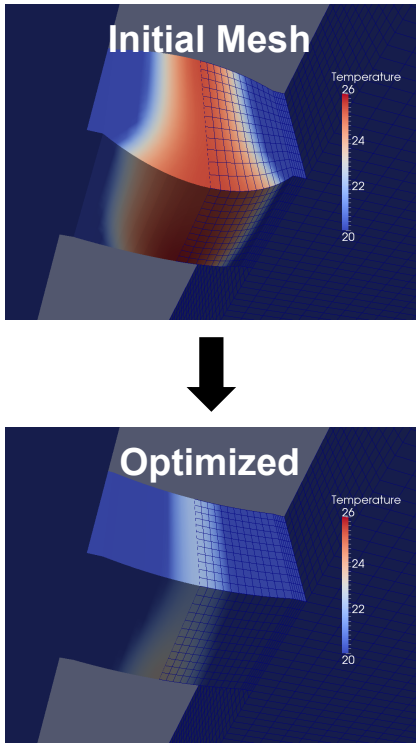


Building application codes with templated components orthogonalizes physics and embedded algorithm R&D

Spectrum of Impact for Template Component Approach

- **Derivatives – Mature capabilities**
  - Jacobians, parameter sensitivities, gradients
    - QASPR semiconductor device physics
    - Drekar MHD
    - CASL thermo-mechanics
    - QCAD quantum device modeling
    - SIERRA/Aria (fluids) and Adagio (solid mechanics)
    - LCM solid mechanics
    - Xyce electronic circuits
  - R&D focused on
    - Efficiency of AD tools (see slide #14)
    - Improving breadth of support (MPI, BLAS, …)

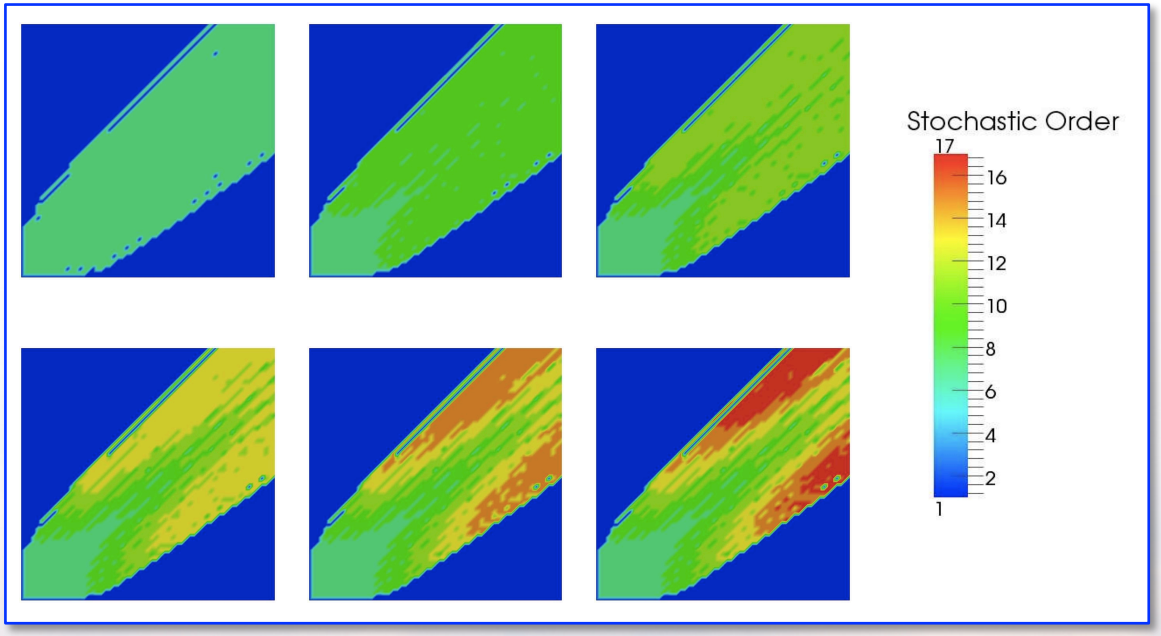
- **Embedded UQ – Basic algorithmic R&D**
  - Collaboration with USC (ASCR)
    - Stochastic Galerkin solver/preconditioners
    - Multiphysics embedded UQ
  - Embedded sampling (ASC)
  - Spatially adaptive embedded UQ (LDRD)
  - Heterogeneous multicore embedded UQ (LDRD)



Shape optimization of a sliding electromagnetic contact subject to Joule heating leveraging AD gradients (Salinger et al – ASC)

Templates Enable Unique Embedded UQ R&D

- **Spatially adaptive UQ of a strongly convected field in Drekar**
  - **Eric Cyr – SNL LDRD**
  - **2-D convection-diffusion with stochastically varying inlet angle**
  - **Intrusive stochastic Galerkin with spatially varying polynomial order**

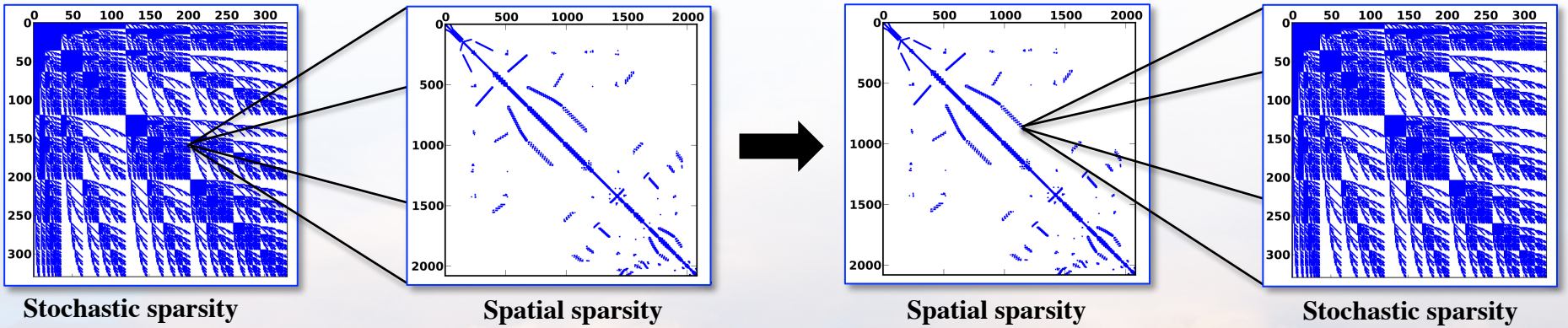


- ***Possible only through embedded approaches***
- ***Templated component approach makes this possible***

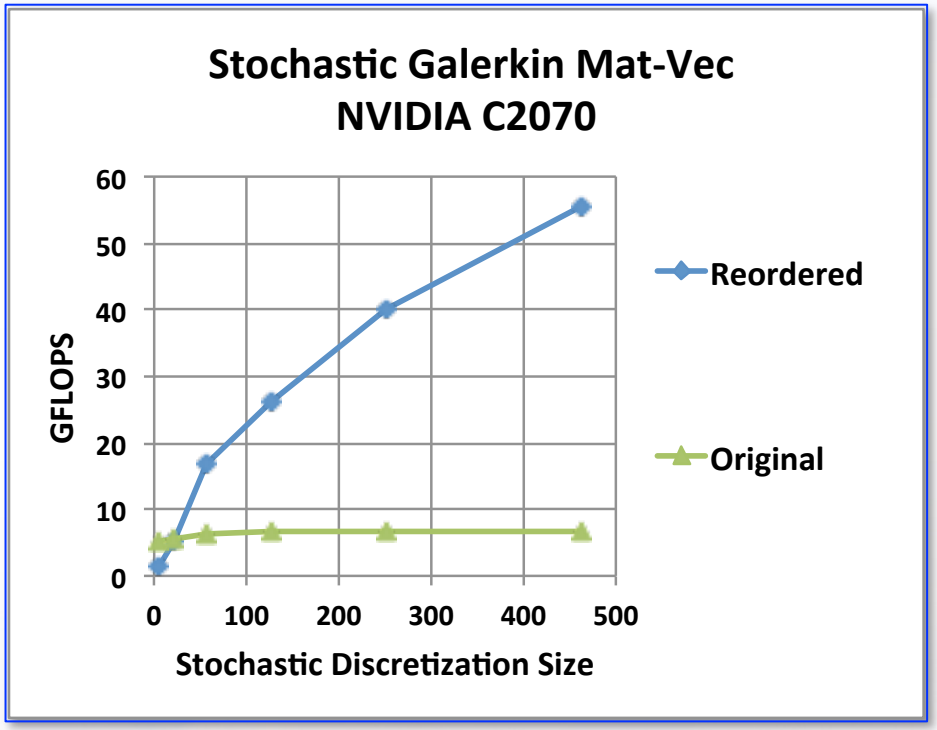
Unique Multicore R&D with Path to Impact

- UQ approaches implemented as an outer loop
  - Single-point forward simulations use very little available compute power
  - Emerging architectures leading to dramatically increased on-node parallelism

- Move UQ to the inner loop to leverage increased parallelism:
  - Phipps, Edwards, Hu – SNL LDRD
  - Embed UQ types in templated linear algebra data structures, solvers, preconditioners
  - Custom UQ-aware multicore linear algebra kernels



Unique Multicore R&D with Path to Impact



- ***Realize exascale performance through embedded UQ***
- ***Templated components make this possible***

Auxiliary Slides

Significant Visibility in External Community

- Automatic differentiation
  - 3 conference papers (ICCS 2006, AD 2008, AD 2012)
  - Program committee for AD 2008
  - Editor, organizing committee, program committee for AD 2012
- Template-based generic programming
  - 2 journal articles under revision
- Embedded/multi-physics UQ
  - 3 published/accepted journal articles, 2 under revision, 1 under review, 3 in preparation
- Numerous conference and workshop presentations

"Predictive Simulation of Complex Coupled Systems under Uncertainty"
Eric Phipps (SNL), Roger Ghanem (USC)

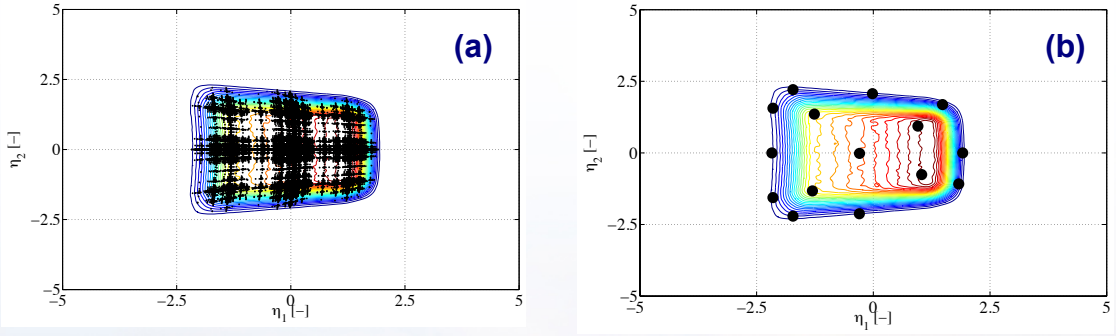
Objectives

- Develop techniques for accurate, efficient, and scalable uncertainty quantification (UQ) of complex, multiphysics, coupled systems

Impact

- Mitigate curse-of-dimensionality inherent in many large-scale multiphysics systems enabling rigorous UQ in large coupled systems

Stochastic Dimension Reduction in a Simplified Reactor Core



PDF and sample points before dimension reduction (a) and after (b) of uncertain input space for heat transfer in a coupled thermal-neutronics problem demonstrating 3 orders of magnitude reduction in computational cost

Accomplishments

- Developed computational framework for *embedded* UQ of coupled systems allowing adaptation of uncertainty representation
- Developed novel UQ approaches that leverage multiphysics structure to reduce dimensionality of uncertainty representations[1-3]
- Developed stable method for computing stochastic expansion coefficients in high dimensions[4]

1. Arnst, Ghanem, Phipps, Red-Horse, "Elements of a computational framework for stochastic coupled-systems modeling: dimension reduction," submitted to IJNME.
2. Arnst, Ghanem, Phipps, Red-Horse, "Measure transformation and efficient quadrature in reduced-dimensional stochastic analysis of coupled systems," submitted to IJNME.
3. Constantine, Phipps, "A Lanczos Method For Approximating Composite Functions," accepted to Appl. Math and Comp.
4. Constantine, Eldred, Phipps, "Sparse Pseudospectral Approximation Method," accepted to CMAME.

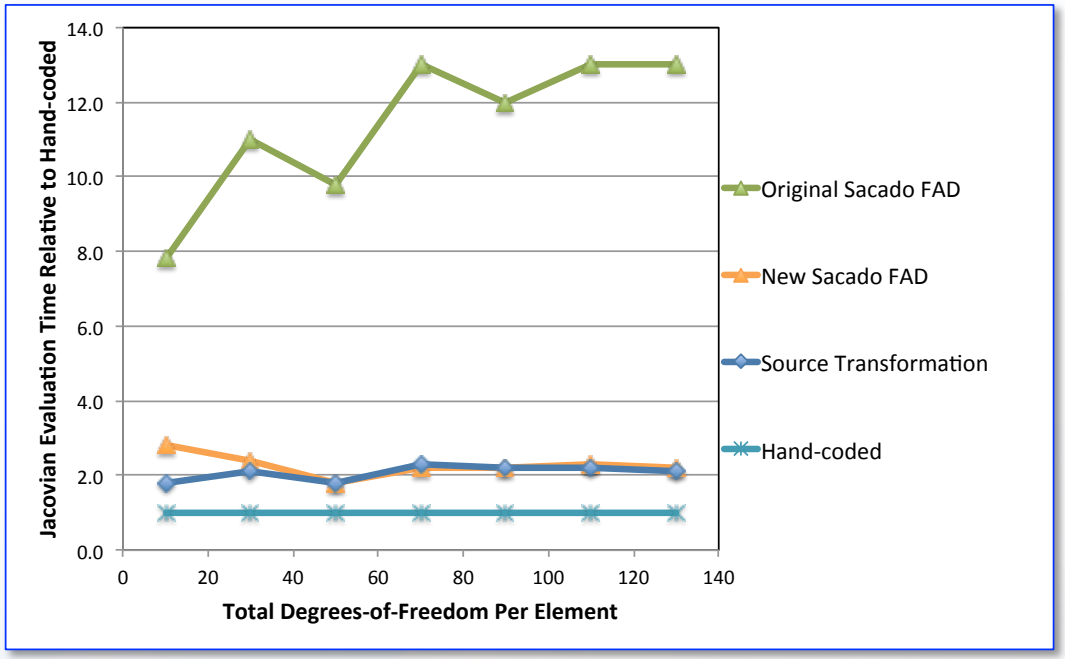
AD Provides Analytic Derivatives Without Hand-Coding

- How does Automatic Differentiation (AD) work?
  - Derivatives at operation level known
  - Chain rule
- What does AD compute?
  - Forward mode derivatives (Jacobians, Jacobian-vector products)
  - Reverse mode derivatives (Gradients, Jacobian-transpose products)
  - High order univariate Taylor polynomials
- How is it implemented?
  - Source transformation (Fortran)
  - Operator overloading (C++)
- Multiple derivative components propagated simultaneously
  - Big cache benefit, potential for multi-core

Our AD Research is Distinguished by Tools & Approach for Large-Scale Codes

- Many AD tools and research projects
  - × Most geared towards Fortran (ADIFOR, OpenAD)
  - × Most C++ tools are slow (ADOL-C)
  - × Most applied in black-box fashion
- Sacado: Operator overloading AD tools for C++ applications
  - ✓ Multiple highly-optimized AD data types
  - ✓ Transform to template code & instantiate on Sacado AD types
  - ✓ Apply AD only at the "element level"
- This is the only successful, sustainable approach for large-scale C++ codes!
- Directly impacting QASPR through Charon
  - ✓ Analytic Jacobians and parameter derivatives

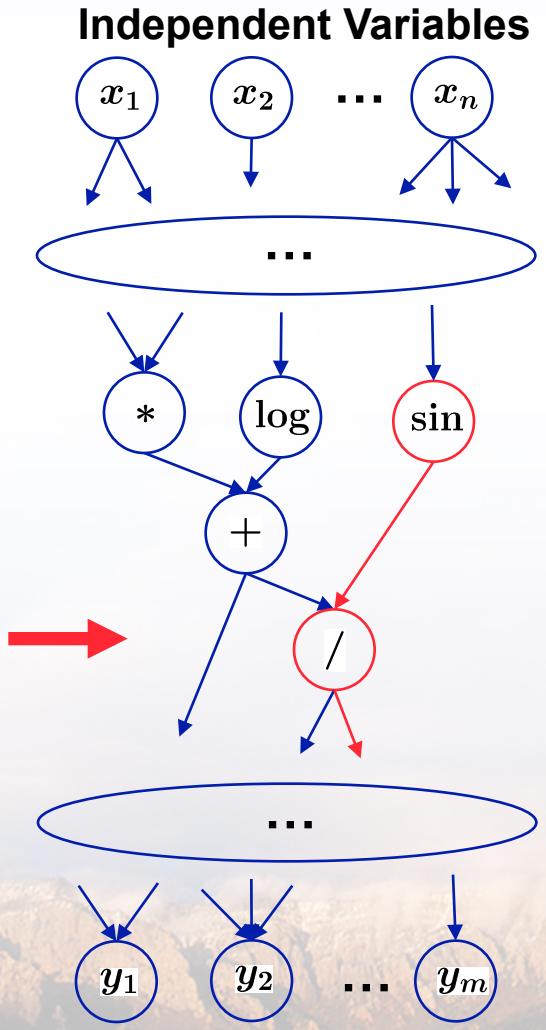
Our AD Tools Perform Extremely Well



- Comparison of Sacado element-based forward AD approach for computing analytic Jacobian to source transformation and hand-coding
- Simple set of representative PDEs
  - Total degrees-of-freedom = number of nodes x number of PDEs for each element
- 6x improvement in Sacado Forward AD tools (FAD) from 2004 to 2012
- Comparison to source transformation demonstrates operator overloading overhead is zero
- 2x cost relative to hand-coded, optimized Jacobian (very problem dependent)

Verification of Automatic Differentiation

- Verification of the AD tools
  - Unit-test with respect to known derivatives
  - Composite tests
    - Compare to other tools
    - Compare to hand-derived
    - Compare to finite differences
- Verification of AD in application code
  - Compiler drastically simplifies this
  - All of the standard hand-coded verification techniques
    - Compare to finite differences
    - Nonlinear convergence



Compiler type mechanism will not allow breaking the chain from independent to dependent variables

Embedded Stochastic Galerkin UQ Methods

- **Steady-state stochastic problem (for simplicity):**

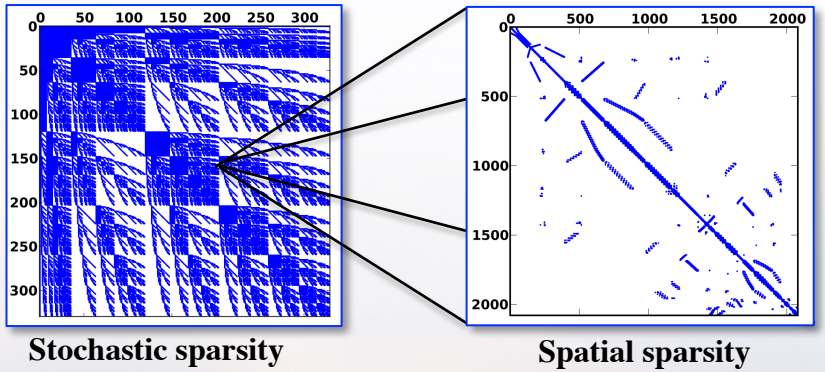
$$\text{Find } u(\xi) \text{ such that } f(u,\xi) = 0,\ \xi : \Omega \to \Gamma \subset R^M, \text{ density } \rho$$

- **Stochastic Galerkin method (Ghanem and many, many others…):**

$$\hat{u}(\xi) = \sum_{i=0}^{P} u_i \psi_i(\xi) \to F_i(u_0, \ldots, u_P) = \frac{1}{\langle \psi_i^2 \rangle} \int_\Gamma f(\hat{u}(y), y)\psi_i(y)\rho(y)dy = 0, \quad i = 0, \ldots, P$$

  - Multivariate orthogonal basis of total order at most N – (generalized polynomial chaos)
- **Method generates new coupled spatial-stochastic nonlinear problem (intrusive)**

$$0 = F(U) = \begin{bmatrix} F_0 \\ F_1 \\ \vdots \\ F_P \end{bmatrix}, \quad U = \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_P \end{bmatrix} \quad \frac{\partial F}{\partial U} :$$



Stochastic sparsity Spatial sparsity

- **Advantages:**
  - Many fewer stochastic degrees-of-freedom for comparable level of accuracy
- **Challenges:**
  - Computing SG residual and Jacobian entries in large-scale, production simulation codes
  - Solving resulting systems of equations efficiently, particularly for nonlinear problems

Stokhos: Trilinos Tools for Embedded Stochastic Galerkin UQ methods

- Eric Phipps, Chris Miller, Habib Najm, Bert Debusschere, Omar Knio

http://trilinos.sandia.gov

- Tools for describing SG discretization
  - Stochastic bases, quadrature rules, etc…
- C++ operator overloading library for automatically evaluating SG residuals and Jacobians
  - Replace low-level scalar type with orthogonal polynomial expansions
  - Leverages Trilinos Sacado automatic differentiation library

$$a = \sum_{i=0}^{P} a_i \psi_i,\ b = \sum_{j=0}^{P} b_j \psi_j,\ c = ab \approx \sum_{k=0}^{P} c_k \psi_k,\ c_k = \sum_{i,j=0}^{P} a_i b_j \frac{\langle \psi_i \psi_j \psi_k \rangle}{\langle \psi_k^2 \rangle}$$

- Tools forming and solving SG linear systems
  - SG matrix operators
  - Stochastic preconditioners
  - Hooks to Trilinos parallel solvers and preconditioners
- Provides tools for investigating embedded UQ methods in large-scale applications

Generating SG Residual/Jacobian Entries Through Automatic Differentiation (AD)

- Trilinos package Sacado provides AD capabilities to C++ codes
  - AD relies on known derivative formulas for all intrinsic operations plus chain rule
  - AD data types & overloaded operators
  - Replace scalar type in application with Sacado AD data types

- Similar approach can be used to apply SG projections in an operation by operation manner

$$\text{Given } a(y) = \sum_{i=0}^{P} a_i \psi_i(y),\ b = \sum_{i=0}^{P} b_i \psi_i(y),\ \text{find } c(y) = \sum_{i=0}^{P} c_i \psi_i(y)$$

$$\text{such that } \int_{\Gamma} \big(c(y) - \phi(a(y), b(y))\big)\psi_i(y)\rho(y)dy = 0, \quad i = 0, \ldots, P$$

  - Simple formulas for addition, subtraction, multiplication, division
  - Transcendental operations are more difficult

SG Projections of Intermediate Operations

- **Addition/subtraction**

$$c = a \pm b \Rightarrow c_i = a_i \pm b_i$$

- **Multiplication**

$$c = a \times b \Rightarrow \sum_i c_i \psi_i = \sum_i \sum_j a_i b_j \psi_i \psi_j \rightarrow c_k = \sum_i \sum_j a_i b_j \frac{\langle \psi_i \psi_j \psi_k \rangle}{\langle \psi_k^2 \rangle}$$

- **Division**

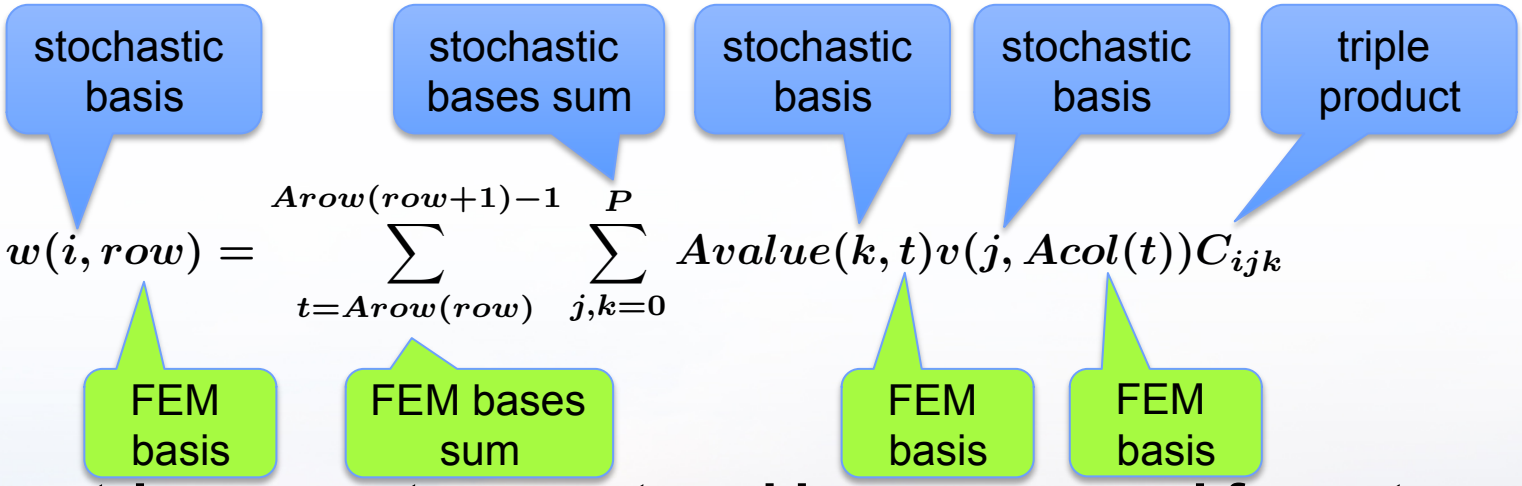
$$c = a/b \Rightarrow \sum_i \sum_j c_i b_j \psi_i \psi_j = \sum_i a_i \psi_i \rightarrow \sum_i \sum_j c_i b_j \langle \psi_i \psi_j \psi_k \rangle = a_k \langle \psi_k^2 \rangle$$

- **Several approaches for transcendental operations**
  - **Taylor series and time integration (Fortran UQ Toolkit by Najm, Debusschere, Ghanem, Knio)**
  - **Tensor product and sparse-grid quadrature (Dakota)**

- **These ideas allow the implementation of Sacado "AD" types for intrusive stochastic Galerkin methods**
  - **Easy transition once code is setup to use AD**

Commuted SG Matrix Orthogonal Polynomial Multiply

- Two level algorithm
  - Outer: traditional CRS matrix-vector multiply algorithm
  - Inner: orthogonal polynomial multiply

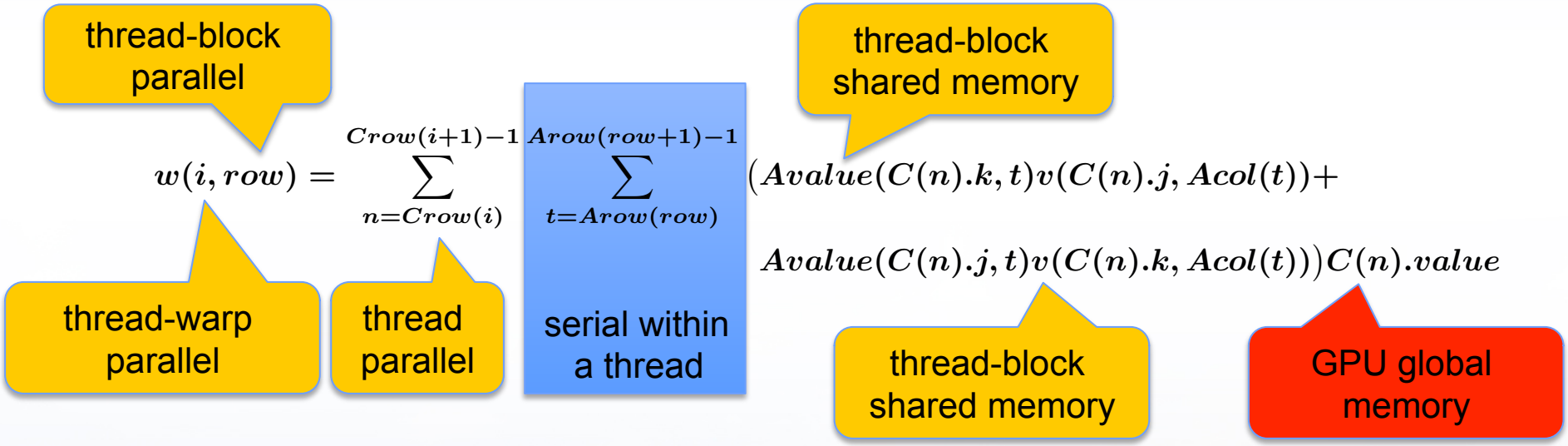


- Symmetric sparse tensor stored in compressed format:

$$w(i,row) = \sum_{t=Arow(row)}^{Arow(row+1)-1} \sum_{n=Crow(i)}^{Crow(i+1)-1} (Avalue(C(n).k,t)v(C(n).j,Acol(t)) + Avalue(C(n).j,t)v(C(n).k,Acol(t)))C(n).value$$

- Opportunities for iteration concurrency: row , i , t , n

Manycore-GPU with Inner Polynomial Multiply: Two-level Concurrency



- **Multiple levels of concurrency:**
  - **Each row owned by a thread-block**
  - **Each warp within a thread-block owns an "i"**
  - **Warps within a thread perform polynomial multiply in parallel, executing CRS loop serially**
- **Currently sparse tensor stored in GPU global memory**