Exceptional service in the national interest

SAND2014-4517C

Sandia National Laboratories

Evaluating the Impact of SDC on the GMRES Iterative Solver

James Elliott[1,2], Mark Hoemmen[1], Frank Mueller[2]

[1]Sandia National Laboratories

[2]North Carolina State University

NC STATE UNIVERSITY
Department of Computer Science

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

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. SAND NO. 2014-1285C

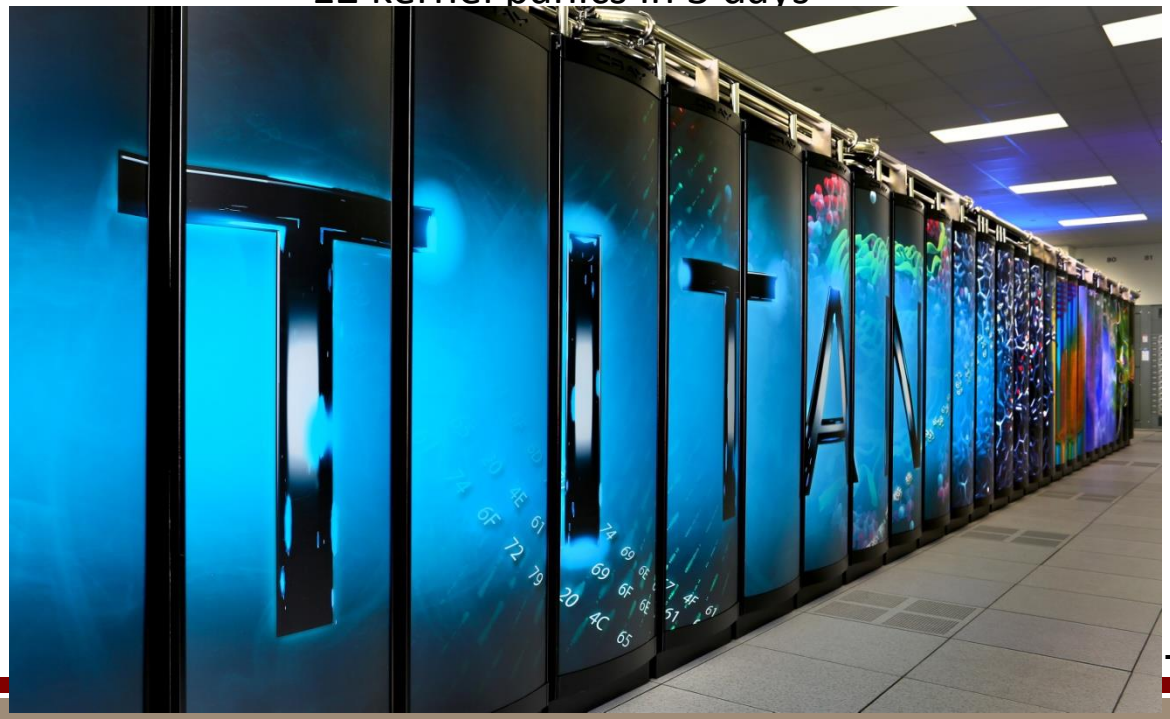
Overview

Sandia National Laboratories

- A Problem
- Myth of a Reliable Machine
- Reliability For Algorithms
- Skeptical Programming
- Results

Motivation

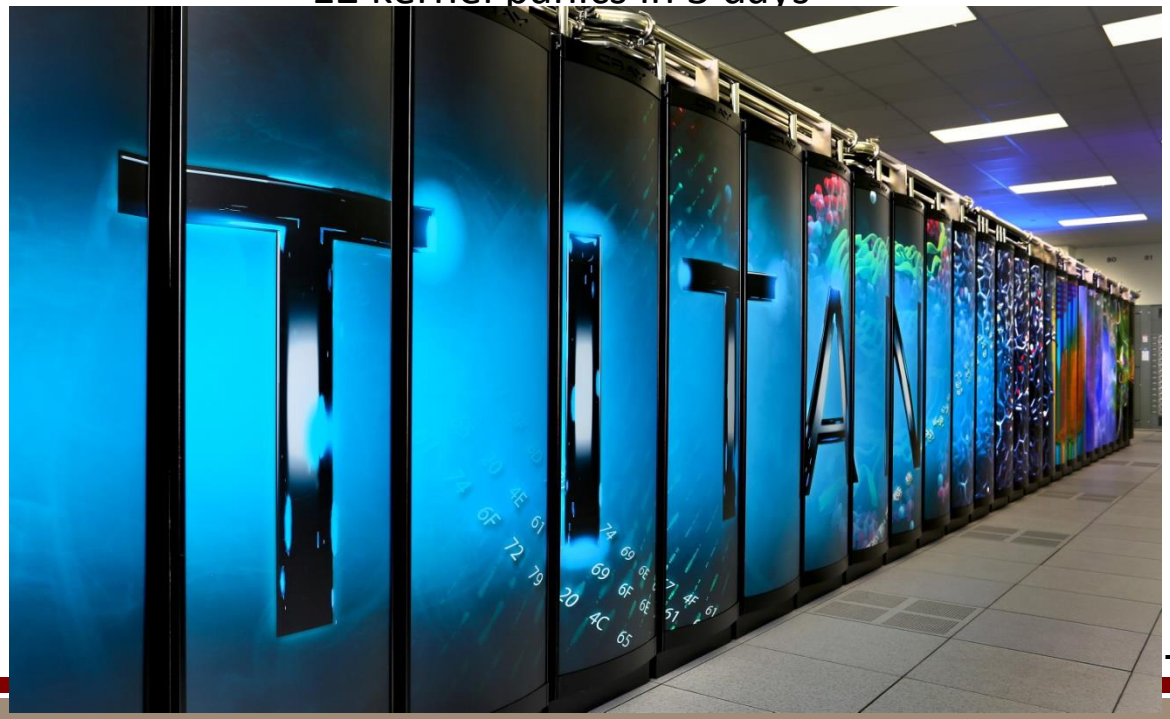
- Titan/Jaguar case study †
  - Constant stream of single bit flips
  - Double bit flip every 24 hours
  - 20 faults per hour
    - Heartbeat fault every 3 minutes
    - 12 kernel panics in 3 days



†Al Geist, Monster in the Closet, 2011

Motivation

- Titan/Jaguar case study †
  - Constant stream of single bit flips
  - Double bit flip every 24 hours
  - 20 faults per hour
    - Heartbeat fault every 3 minutes
    - 12 kernel panics in 3 days



Root cause analysis difficult …

Is it even necessary for algorithm designers!

†Al Geist, Monster in the Closet, 2011

The Problem (well one of them)

Sandia National Laboratories

Function: Puppy = make_puppy(Dog, Dog)

The Problem (well one of them)

Sandia National Laboratories

Function: Puppy = make_puppy(Dog, Dog)

The Problem (well one of them)

Sandia National Laboratories

Function: Puppy = make_puppy(Dog, Dog)

+

The Problem (well one of them)

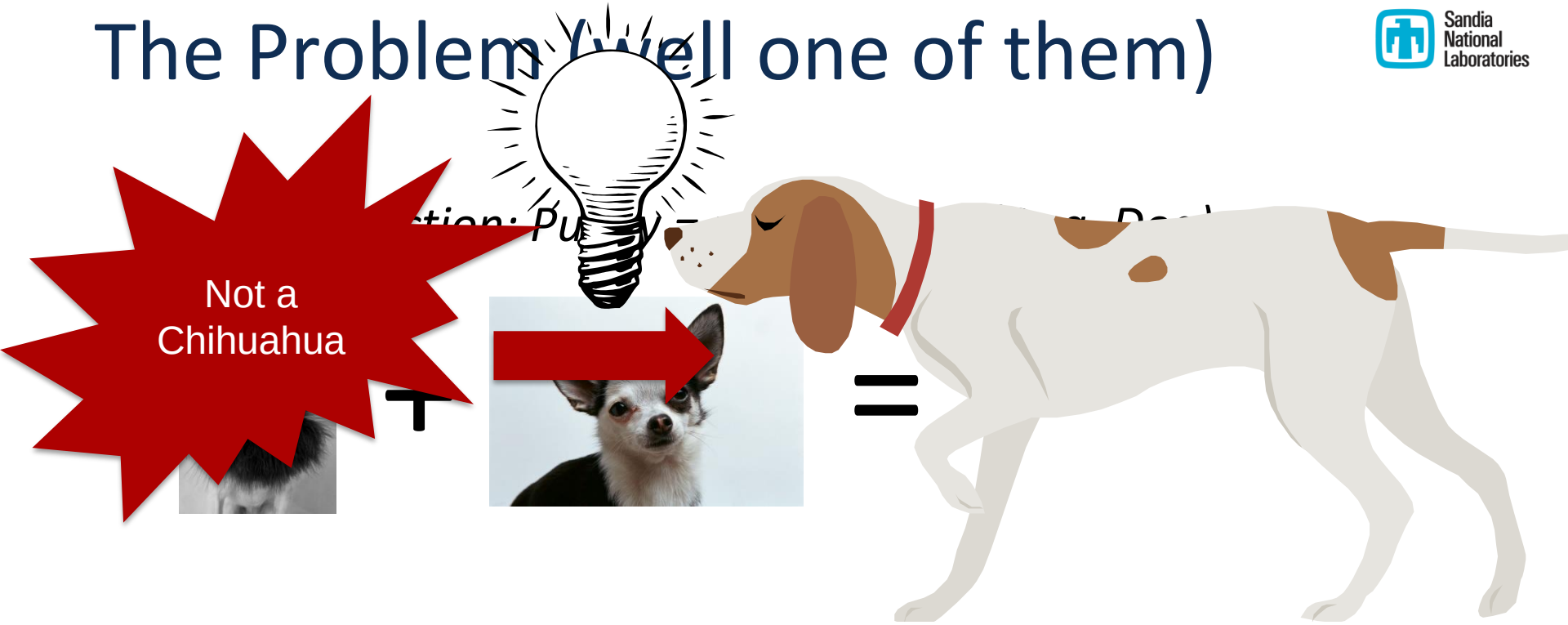
Sandia National Laboratories

Function: Puppy = make_puppy(Dog, Dog)

+

=

The Problem (Well one of them)



The Problem (well one of them)

Two things happened:

1. You noticed a fault, therefore …
2. You can respond (correct the fault)

Detection and Correction

The Problem (well one of them)

Sandia National Laboratories

How to detect?

- Enforce fine grain correctness (systems approach)
  - ✓ **Do calculations multiple times and vote**
  - × **Ignores numerical properties of algorithms**

The Problem (well one of them)

Sandia National Laboratories

How to detect?

- Enforce fine grain correctness (systems approach)
  - ✓ **Do calculations multiple times and vote**
  - × **Ignores numerical properties of algorithms**
- Convergence (numerical error dampening)
  - ✓ **Iterative methods have proven properties to converge to a correct solution given preconditions/assumptions about the inputs**
  - × **Convergence promise invalidated if assumptions/preconditions are invalidated at any point.**

The Problem (well one of them)

Sandia National Laboratories

How to detect?

- Enforce fine grain correctness (systems approach)
  - ✓ **Do calculations multiple times and vote**
  - × **Ignores numerical properties of algorithms**
- Convergence (numerical error dampening)
  - ✓ **Iterative methods have proven properties to converge to a correct solution given preconditions/assumptions about the inputs**
  - × **Convergence promise invalidated if assumptions/preconditions are invalidated at any point.**
- Can we do algorithm fault tolerance while still exploiting the numerical properties?

Myth of a Reliable Machine

Sandia National Laboratories

- Computers are inherently **unreliable**
- We design software and hardware that work given some tolerance
  - Windows crashes…but not too often
  - Screens flicker…but faster than the eye can see
  - Video games draw frames…but can skip some
- Numerical Mathematics has adapted
  - IEEE spec mandates behavior, standardizes representation
  - Integrity of approximations can be proven given assumptions promised by floating point

Myth of a Reliable Machine

Sandia National Laboratories

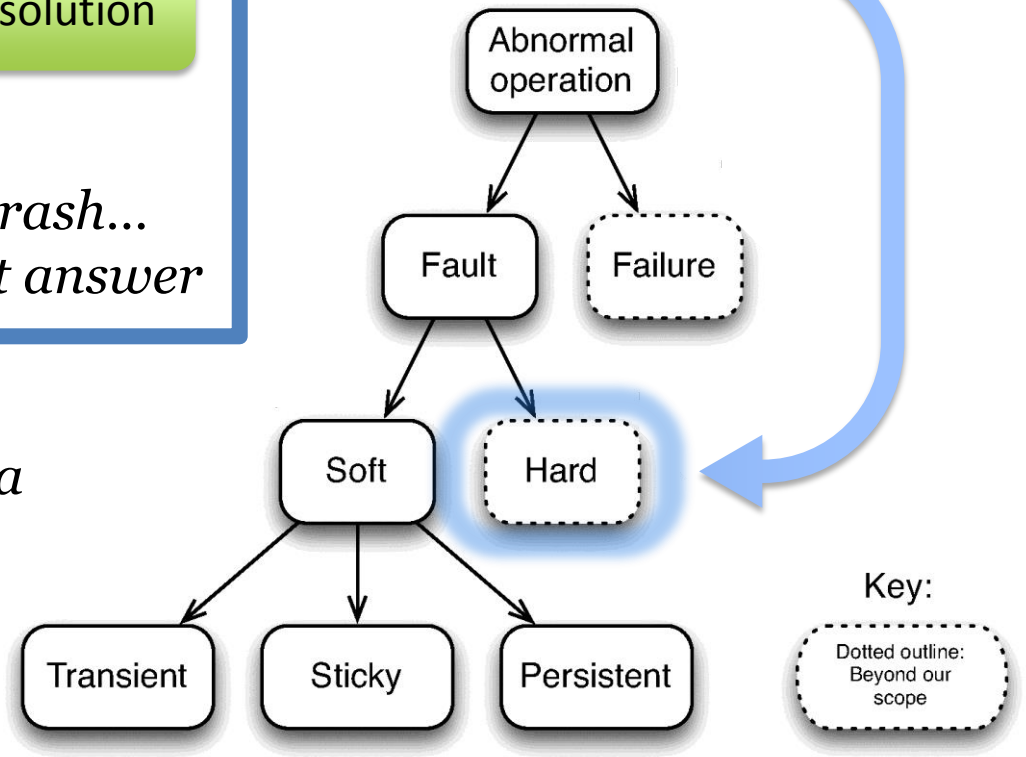
Current algorithms assume ***numerical reliability***

Reliable Machine

start → operations → solution

As long the application doesn't crash…
we will get the right answer

Hard Faults: Not in scope of this work



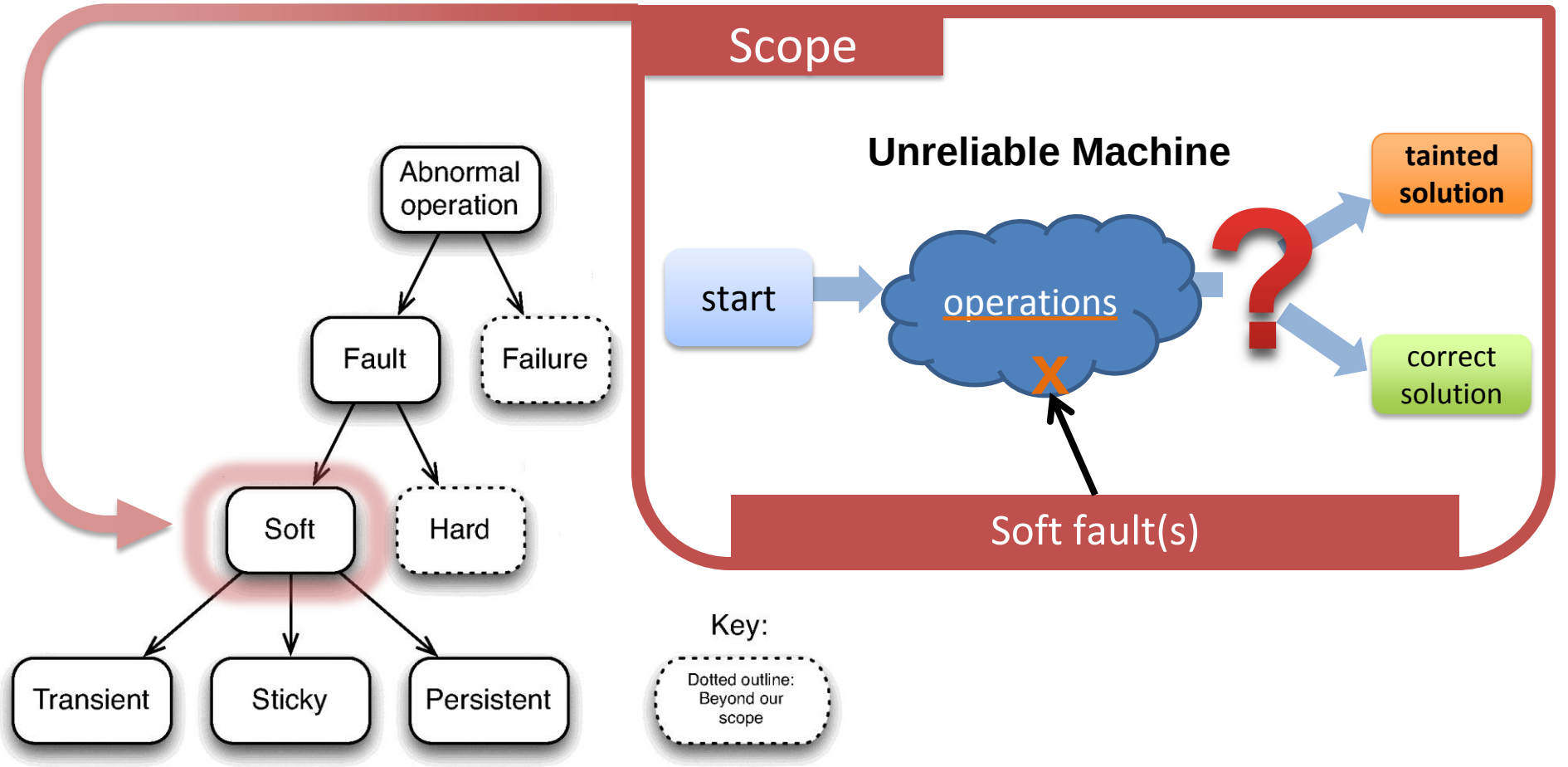
Machine reliability addressed via checkpoint/restart, replication, algorithm based approaches

Myth of a Reliable Machine

Sandia National Laboratories

Numerical reliability:

numerical operations may be unreliable



Reliability for Algorithms

Sandia National Laboratories

Numerical Method = Theory + Data

(algorithm) (with assumptions)

- **Two Categories:**
  - **Data:** data required theoretically by the algorithm (Preconditions/Assumptions)

    E.g., for GMRES: A, x0, orthonormal subspace of A, right-hand side
  - **Meta-data:** "stuff" required to implement the algorithm

    E.g., loop counters, sparse data structures (indices), code

Regardless of who implements GMRES or in what language, the data can always be assumed (otherwise we would not be considering GMRES!)
Language and Implementation details may result in vastly different meta-data.

- Most extensible research: **focus on faults in data.**

Reliability for Algorithms

Sandia National Laboratories

- Result of silent fault in **data**:
  - **Silent Data Corruption (SDC)**
- Result of silent transient fault in **meta-data**
  - Faults in code:
    - Weird behavior
    - Perform wrong operation (e.g., add instead of divide) **= SDC** (output is "tainted")
  - Faults in loop counters, indices (if no segfault/crash)
    - Operate on incorrect data **= SDC** (output is "tainted")
    - Iterate too much or too little **= SDC** (output is "tainted")
- Perhaps more

Key: corrupted data model *many* types of faults!

Big Idea: Bounded Error

Sandia National Laboratories

- Focus on ensuring **bounded error,** *rather than* ***detecting and correcting <u>all errors</u>*.**

Big Idea: Bounded Error

Sandia National Laboratories

- Focus on ensuring **bounded error,** *rather than* ***detecting and correcting <u>all errors</u>*.**
- Use preconditions/assumptions on inputs with algorithms' theoretical basis to derive invariants.

Big Idea: Bounded Error

Sandia National Laboratories

- Focus on ensuring **bounded error,** *rather than* ***detecting and correcting all errors*****.**
- Use preconditions/assumptions on inputs with algorithms' theoretical basis to derive invariants.
- Invariant checks are silly in a reliable environment
  - If unreliable – **invariants serve as cheap detectors** that the algorithm is in a **theoretically impossible state**.

Big Idea: Bounded Error

Sandia National Laboratories

- Focus on ensuring **bounded error,** *rather than* ***detecting and correcting all errors*.**
- Use preconditions/assumptions on inputs with algorithms' theoretical basis to derive invariants.
- Invariant checks are silly in a reliable environment
  - If unreliable – **invariants serve as cheap detectors** that the algorithm is in a **theoretically impossible state**.
- No promise to detect/correct all errors
  - One piece of the resilient algorithm pie
- We call this Skeptical Programming.
  - **Be skeptical of computations performed on an unreliable machine**

Skeptical Programming

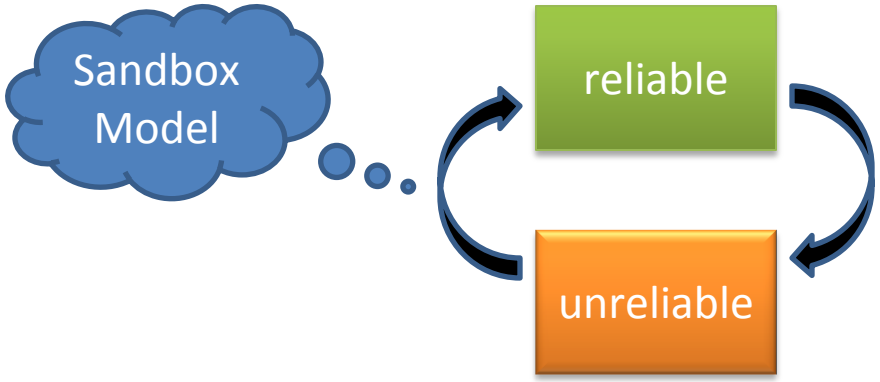
Sandia National Laboratories

- GMRES' flops are spent in orthogonalization
  - **Project linear system**
    - **(sparse matrix vector product)**
  - **Form orthonomal Krylov subspace**
    - **(dot products)**
  - **Least squares solve**
    - **Small local matrix operation**
  - **Optional: Solution Update / Explicit Residual Calculation (Ax-b)**

Reliability

Looking from the Reliable Set:

- **Sandbox Model**



Operate unreliably, but do not trust the results.

- **Selective Reliability:**

  What operations need to be reliable?

Universe of **operations** inherently ***unreliable***

Unreliable ops:
Preconditioner call (and all its operations)

Reliable ops:
Flexible GMRES, (currently all collectives)

Selective Reliability

Sandia National Laboratories

Algorithm 1 Flexible GMRES (FGMRES)

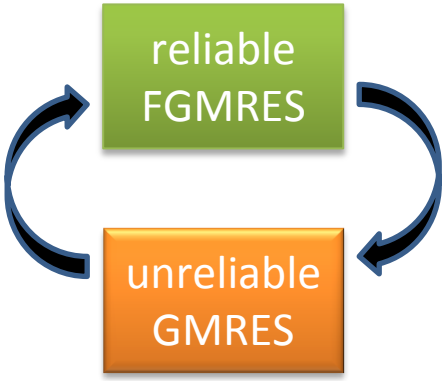
Input: Linear system $Ax = b$ and initial guess x_0
Output: Approximate solution x_m for some $m \geq 0$

1: $\mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$ ▷ Unpreconditioned initial residual
2: $\beta := \|\mathbf{r}_0\|_2$, $\mathbf{q}_1 := \mathbf{r}_0/\beta$
3: **for** $j = 1, 2, \ldots$ until convergence **do**
4: Solve $\mathbf{q}_j = \mathbf{M}_j\mathbf{z}_j$ ▷ Apply current preconditioner
5: $\mathbf{v}_{j+1} := \mathbf{A}\mathbf{z}_j$ ▷ Apply the matrix $\mathbf{A}$
6: **for** $i = 1, 2, \ldots, k$ **do** ▷ Orthogonalize
7: $h_{i,j} := \mathbf{q}_i \cdot \mathbf{v}_{j+1}$
8: $\mathbf{v}_{j+1} := \mathbf{v}_{j+1} - h_{i,j}\mathbf{q}_i$
9: **end for**
10: $h_{j+1,j} := \|\mathbf{v}_{j+1}\|_2$
11: Update rank-revealing decomposition of $\mathbf{H}(1{:}j, 1{:}j)$
12: **if** $\mathbf{H}(j+1, j)$ is less than some tolerance **then**
13: **if** $\mathbf{H}(1{:}j, 1{:}j)$ not full rank **then**
14: Did not converge; report error
15: **else**
16: Solution is $\mathbf{x}_{j-1}$ ▷ Happy breakdown
17: **end if**
18: **else**
19: $\mathbf{q}_{j+1} := \mathbf{v}_{j+1}/h_{j+1,j}$
20: **end if**
21: $\mathbf{y}_j := \arg\min_y \|\mathbf{H}(1{:}j+1, 1{:}j)\mathbf{y} - \beta\mathbf{e}_1\|_2$
22: $\mathbf{x}_j := \mathbf{x}_0 + [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_j]\mathbf{y}_j$ ▷ Compute solution update
23: **end for**

M_j are the preconditioners:

$$\mathbf{z}_j = \text{gmres}(\mathbf{A}, \mathbf{q}_j)$$

M_j represents using GMRES as a preconditioner… inside FGMRES.



Theoretical reason to believe this approach

Skeptical Programming

Sandia National Laboratories

Algorithm 1: GMRES

for $l = 1$ to do
  $\mathbf{r} := \mathbf{b} - \mathbf{A}\mathbf{x}^{(j-1)}$
  $\mathbf{q}_1 := \mathbf{r} / ||r||_2$
  for $j = 1$ to *restart* do
    $\mathbf{w}_0 := \mathbf{A}\mathbf{q}_j$
    for $i = 1$ to j do
      $h_{i,j} := \mathbf{q}_i \cdot \mathbf{w}_{i-1}$
      $\mathbf{w}_i := \mathbf{w}_{i-1} - h_{i,j}\mathbf{q}_i$
    end
    $h_{j+1,j} := ||\mathbf{w}||_2$
    $\mathbf{q}_{j+1} := \mathbf{w}/h_{j+1,j}$
    Find $\mathbf{y} = \min ||\mathbf{H}_j\mathbf{y} - ||\mathbf{b}||\,\mathbf{e}_1||_2$
    Evaluate convergence criteria
    Optionally, compute $\mathbf{x}_j = \mathbf{Q}_j\mathbf{y}$
  end
end

Skeptical Programming

Sandia National Laboratories

Algorithm 1: **GMRES**

for $l = 1$ **to** **do**
  $\mathbf{r} := \mathbf{b} - \mathbf{A}\mathbf{x}^{(j-1)}$
  $\mathbf{q}_1 := \mathbf{r} / ||r||_2$
  for $j = 1$ **to** *restart* **do**
    $\mathbf{w}_0 := \mathbf{A}\mathbf{q}_j$
    for $i = 1$ **to** j **do**
      $h_{i,j} := \mathbf{q}_i \cdot \mathbf{w}_{i-1}$
      $\mathbf{w}_i := \mathbf{w}_{i-1} - h_{i,j}\mathbf{q}_i$
    end
    $h_{j+1,j} := ||\mathbf{w}||_2$
    $\mathbf{q}_{j+1} := \mathbf{w}/h_{j+1,j}$
    Find $\mathbf{y} = \min ||\mathbf{H}_j\mathbf{y} - ||\mathbf{b}||\,\mathbf{e}_1||_2$
    Evaluate convergence criteria
    Optionally, compute $\mathbf{x}_j = \mathbf{Q}_j\mathbf{y}$
  end
end

Orthogonalization
Builds upper Hessenburg matrix (H)

Skeptical Programming

Sandia National Laboratories

Algorithm 1: GMRES

for $l = 1$ to do
  $\mathbf{r} := \mathbf{b} - \mathbf{A}\mathbf{x}^{(j-1)}$
  $\mathbf{q}_1 := \mathbf{r} / ||r||_2$
  for $j = 1$ to *restart* do
    $\mathbf{w}_0 := \mathbf{A}\mathbf{q}_j$
    for $i = 1$ to j do
      $h_{i,j} := \mathbf{q}_i \cdot \mathbf{w}_{i-1}$
      $\mathbf{w}_i := \mathbf{w}_{i-1} - h_{i,j}\mathbf{q}_i$
    end
    $h_{j+1,j} := ||\mathbf{w}||_2$
    $\mathbf{q}_{j+1} := \mathbf{w} / h_{j+1,j}$
    Find $\mathbf{y} = \min ||\mathbf{H}_j\mathbf{y} - ||\mathbf{b}|| \, \mathbf{e}_1||_2$
    Evaluate convergence criteria
    Optionally, compute $\mathbf{x}_j = \mathbf{Q}_j\mathbf{y}$
  end
end

Theoretical Bounds on the Arnoldi Process

$$\|\mathbf{w}_0\| = \|\mathbf{A}\mathbf{q}_j\| \leq \|\mathbf{A}\|_2 \|\mathbf{q}_j\|_2$$
$$\|\mathbf{w}_0\| \leq \|\mathbf{A}\|_2 \leq \|\mathbf{A}\|_F$$

From isometry of orthogonal projections,

$$|h_{i,j}| \leq \|\mathbf{A}\|_F$$

- $h_{i,j}$ form Hessenberg Matrix
- Bound only computed once, valid for entire solve
- If triggered … (see results)

Skeptical Programming and Selective Reliability

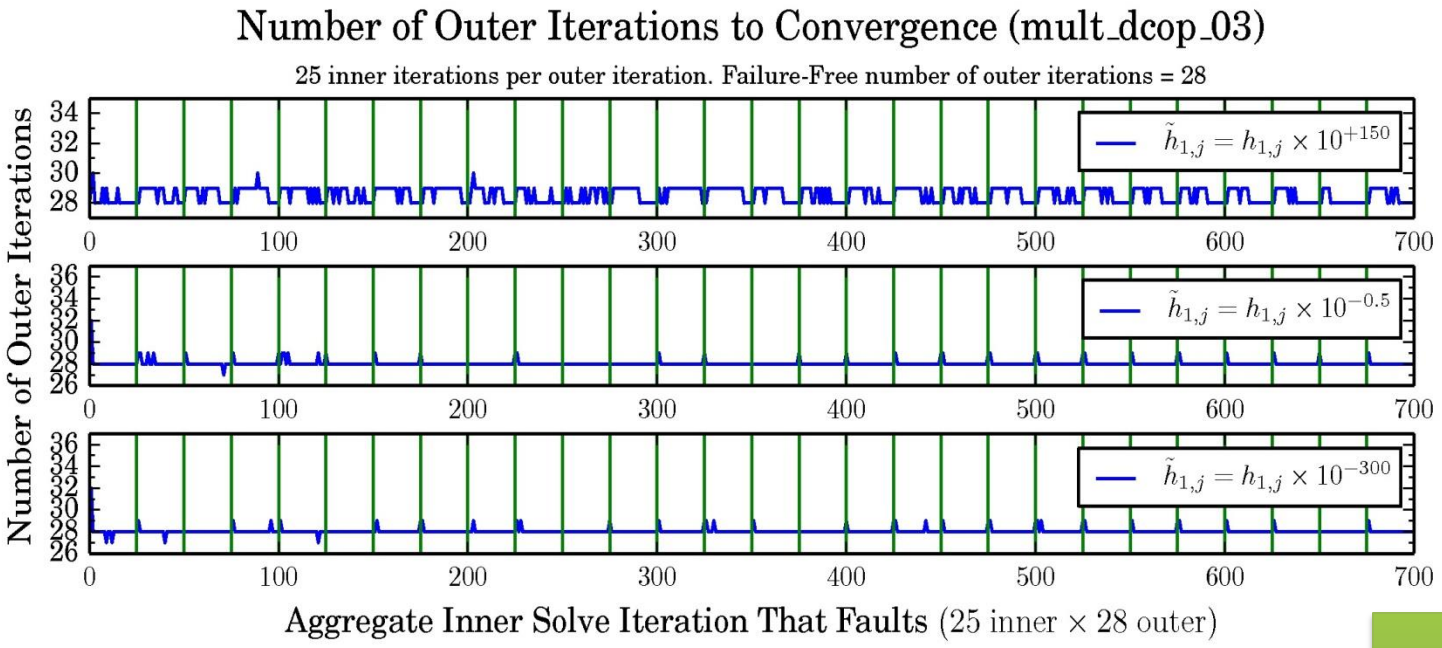
Sandia National Laboratories

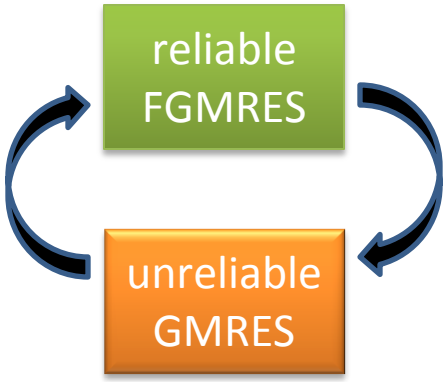
Skeptical Programming not end-all solution…

enhances Fault Tolerance

- Prototyped using FT-GMRES (using Trilinos Framework)
  - Flexible GMRES preconditioned by GMRES
- Selective Reliability
  - Outer solver (everything but preconditioner) is reliable
  - Preconditioner can be unreliable
- Sandbox
  - Unreliability of preconditioner is “sandboxed”
- Skeptical Programming
  - Errors are “more bounded” than if no checks used
  - Checks add little overhead

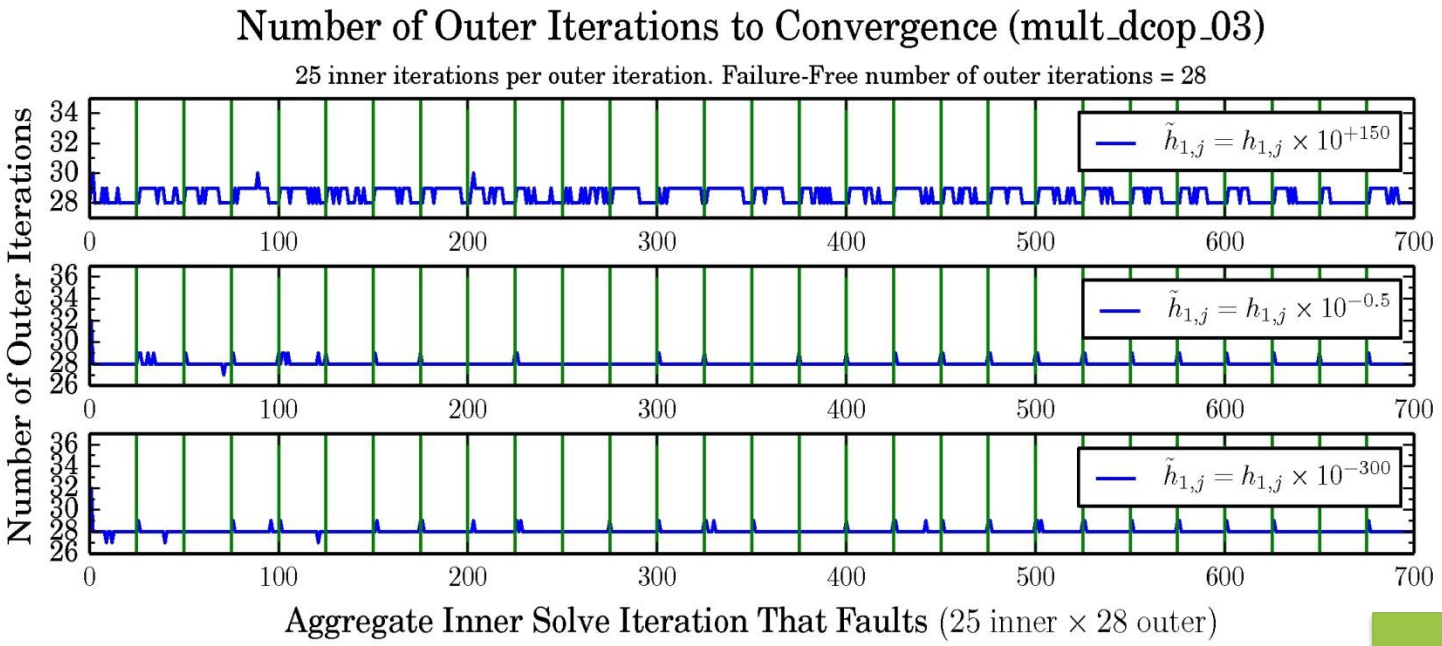
Prototype



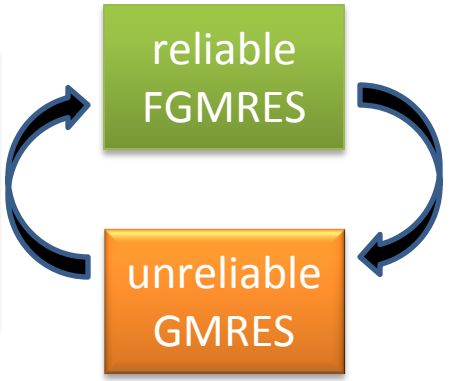


Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Prototype



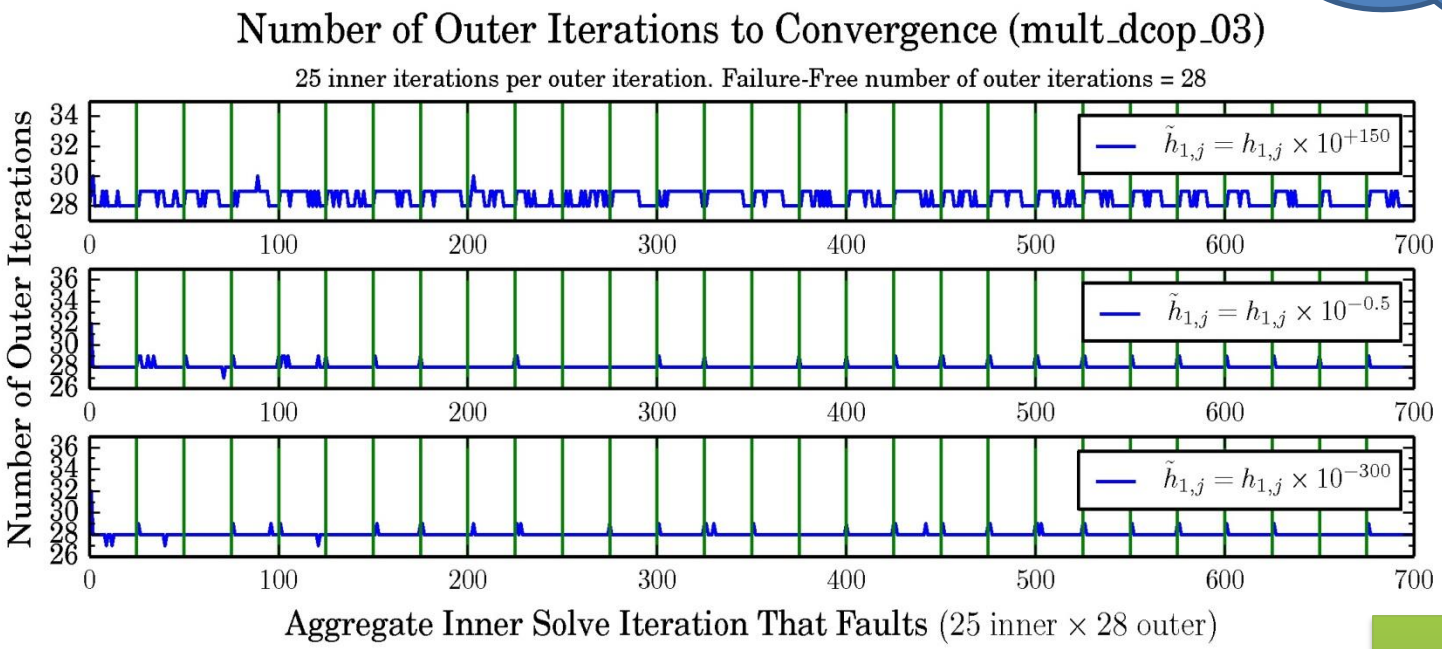
Failure Free: **700 orthgonalization calls**
28 outer iterations to converge
25 inner iterations per outer
x-axis:
introducing single fault in x-th orthogonalization



Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Prototype

700 experiments per fault type



Failure Free: **700 orthgonalization calls**
28 outer iterations to converge
25 inner iterations per outer
x-axis:
introducing single fault in x-th orthogonalization

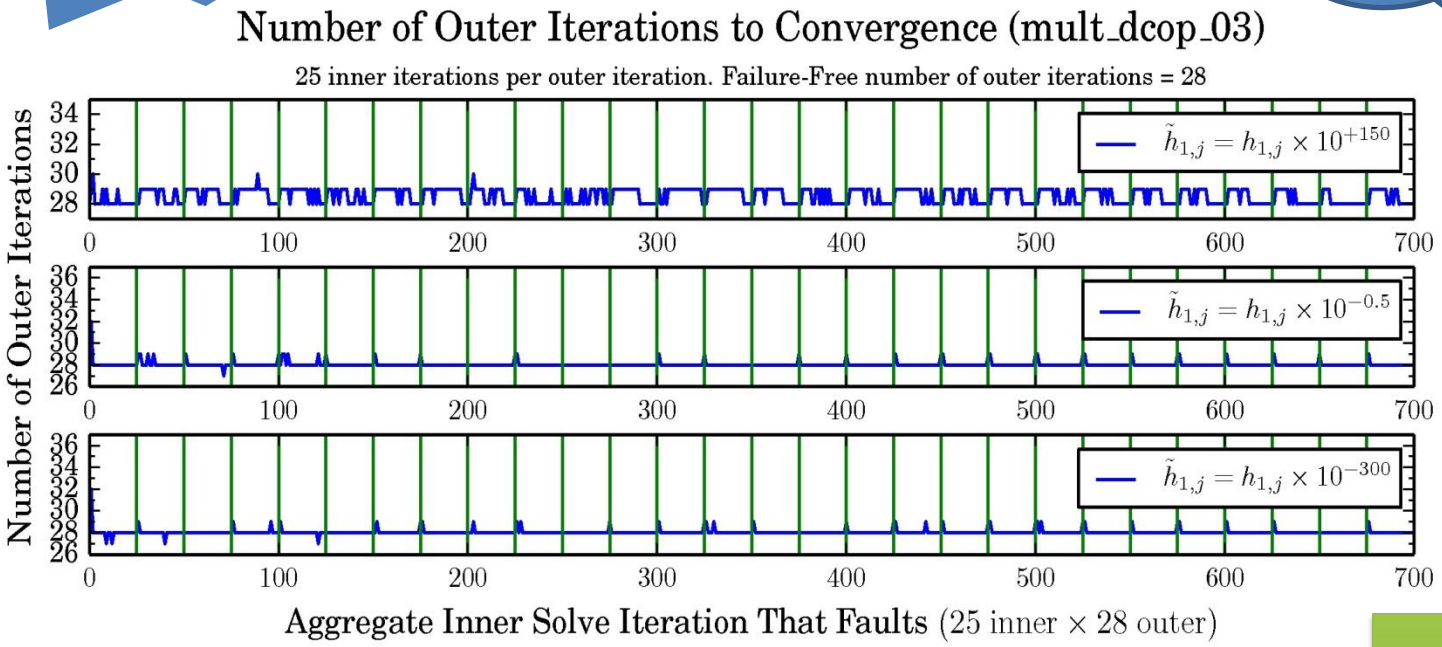
reliable FGMRES

unreliable GMRES

Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Inject ***single*** fault
Study impact

700
experiments per
fault type



Failure Free: **700 orthgonalization calls**
28 outer iterations to converge
25 inner iterations per outer
x-axis:
introducing single fault in x-th orthogonalization

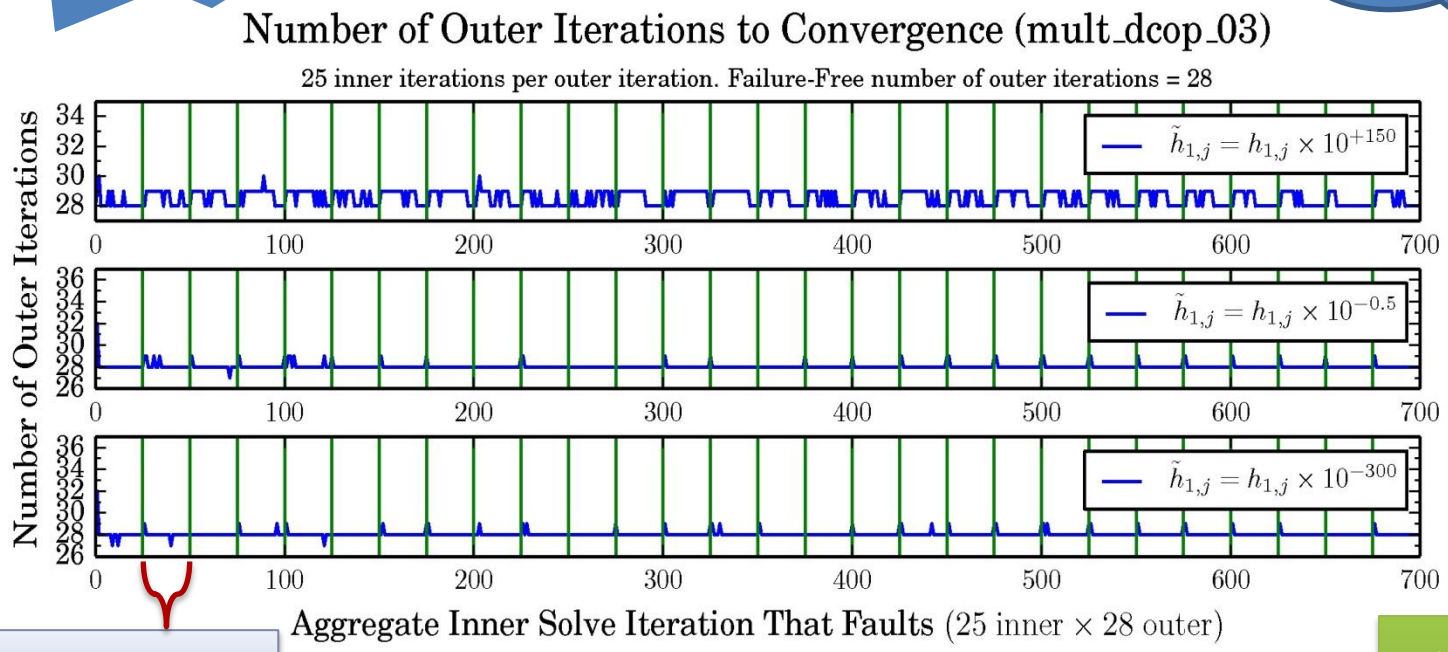
reliable
FGMRES

unreliable
GMRES

Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Inject ***single*** fault
Study impact

700
experiments per
fault type



one inner solve

Failure Free: **700 orthgonalization calls**
28 outer iterations to converge
25 inner iterations per outer
x-axis:
introducing single fault in x-th orthogonalization

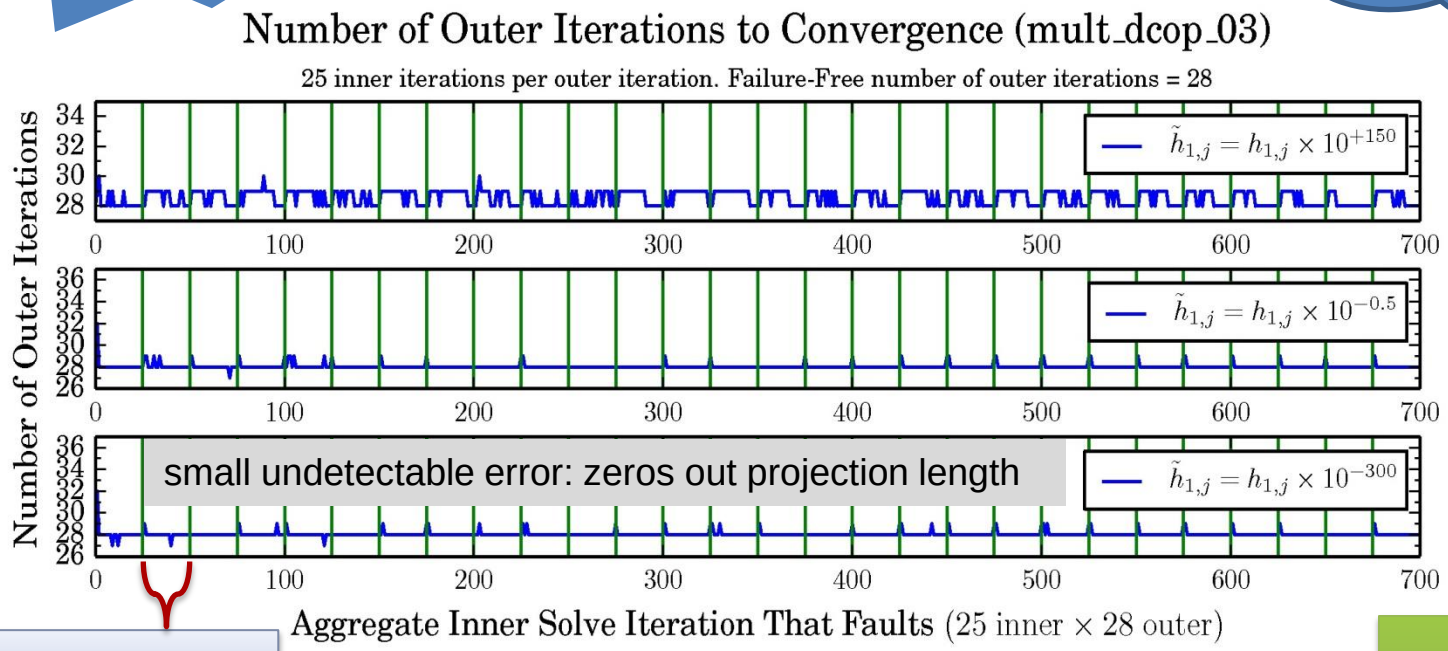
reliable
FGMRES

unreliable
GMRES

Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Inject ***single*** fault
Study impact

700
experiments per
fault type



one inner solve

Failure Free: **700 orthgonalization calls**
28 outer iterations to converge
25 inner iterations per outer
x-axis:
introducing single fault in x-th orthogonalization

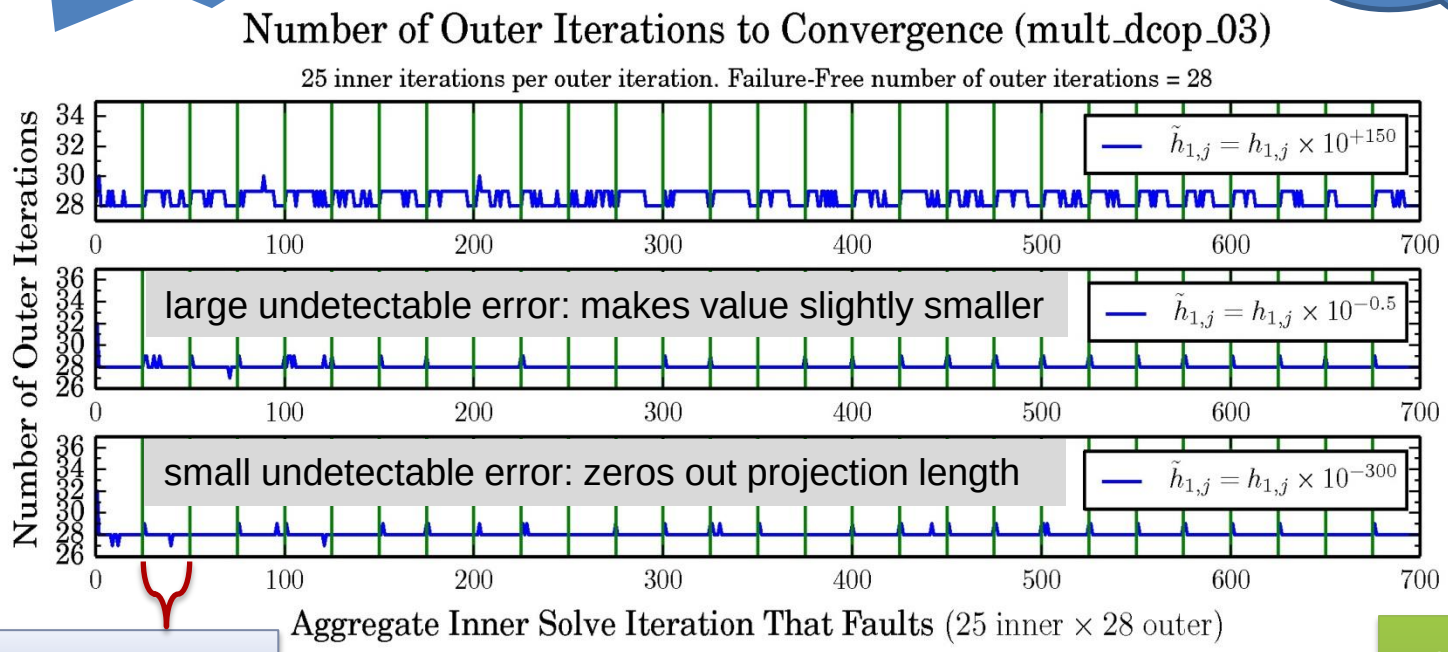
reliable
FGMRES

unreliable
GMRES

Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Inject ***single*** fault
Study impact

700
experiments per
fault type



one inner solve

Failure Free: **700 orthgonalization calls**
28 outer iterations to converge
25 inner iterations per outer
x-axis:
introducing single fault in x-th orthogonalization

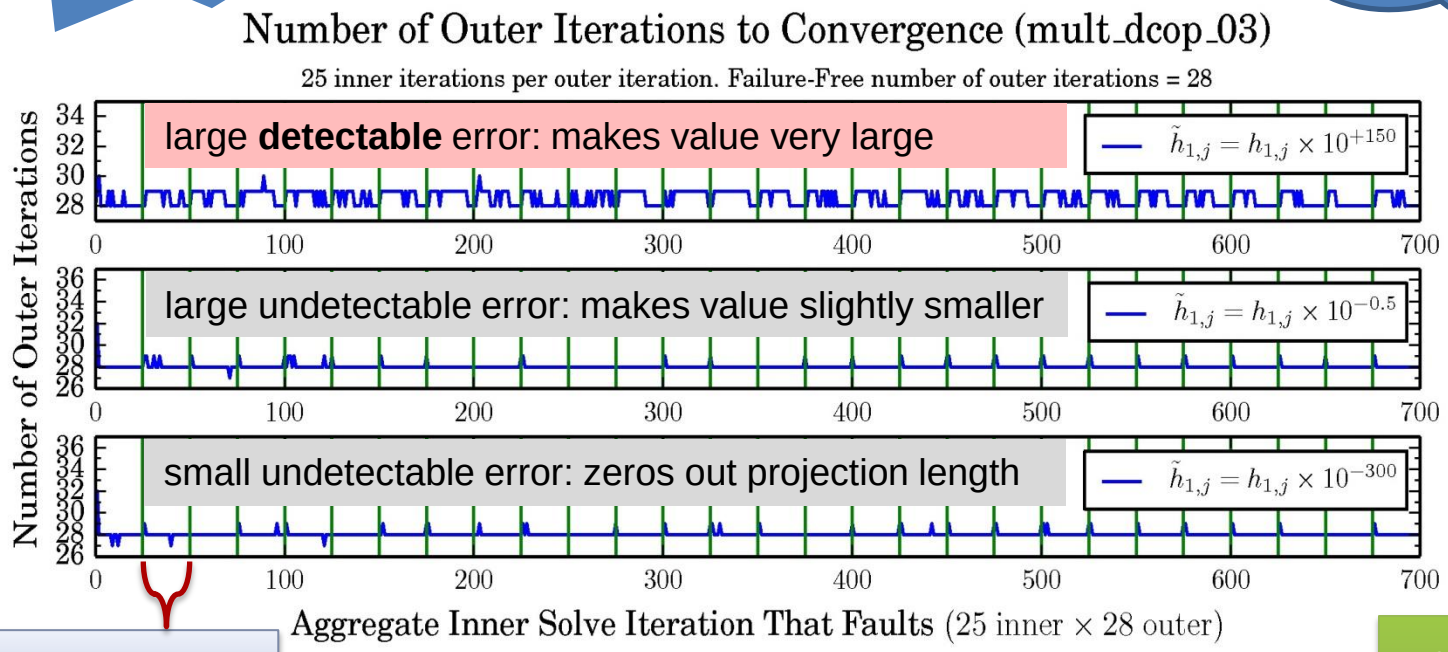
reliable
FGMRES

unreliable
GMRES

Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Inject ***single*** fault
Study impact

700
experiments per
fault type



one inner solve

Failure Free: **700 orthgonalization calls**
28 outer iterations to converge
25 inner iterations per outer
x-axis:
introducing single fault in x-th orthogonalization

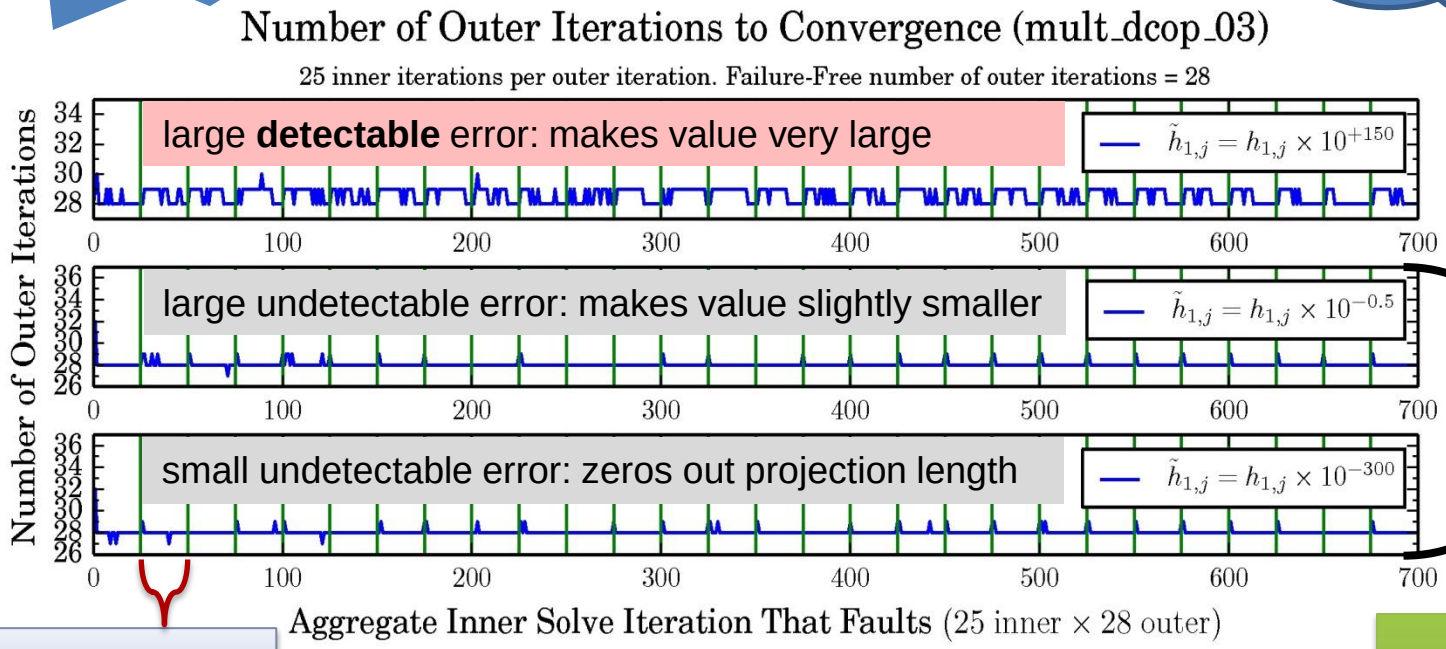
reliable
FGMRES

unreliable
GMRES

Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Inject ***single*** fault
Study impact

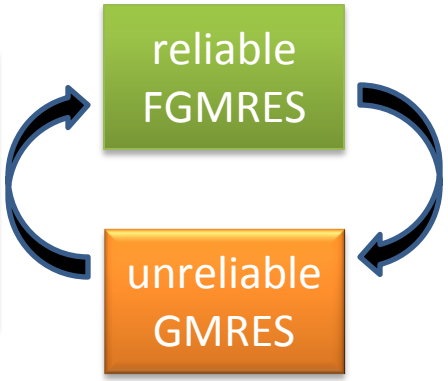
700
experiments per
fault type



Smaller than bound

one inner solve

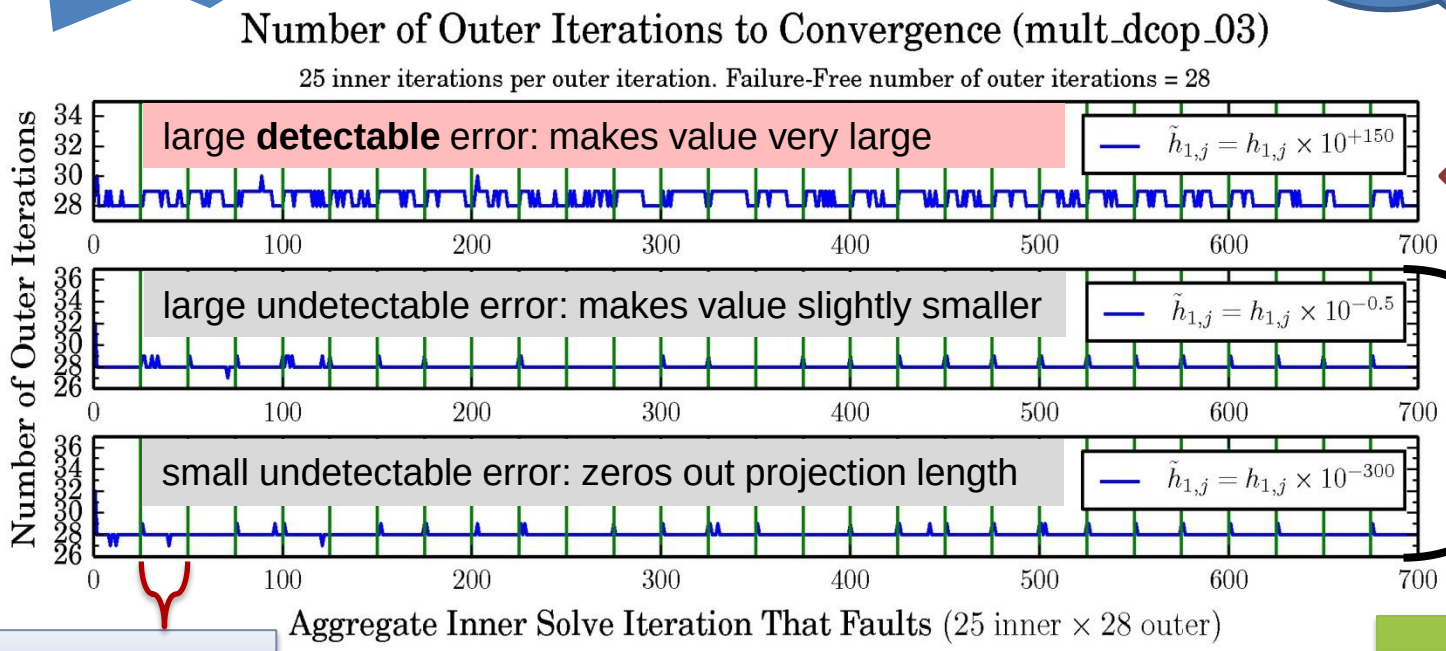
Failure Free: **700 orthgonalization calls**
28 outer iterations to converge
25 inner iterations per outer
x-axis:
introducing single fault in x-th orthogonalization



Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Inject ***single*** fault
Study impact

700
experiments per
fault type



Larger than bound

Smaller than bound

one inner solve

Failure Free: **700 orthgonalization calls**
28 outer iterations to converge
25 inner iterations per outer
x-axis:
introducing single fault in x-th orthogonalization

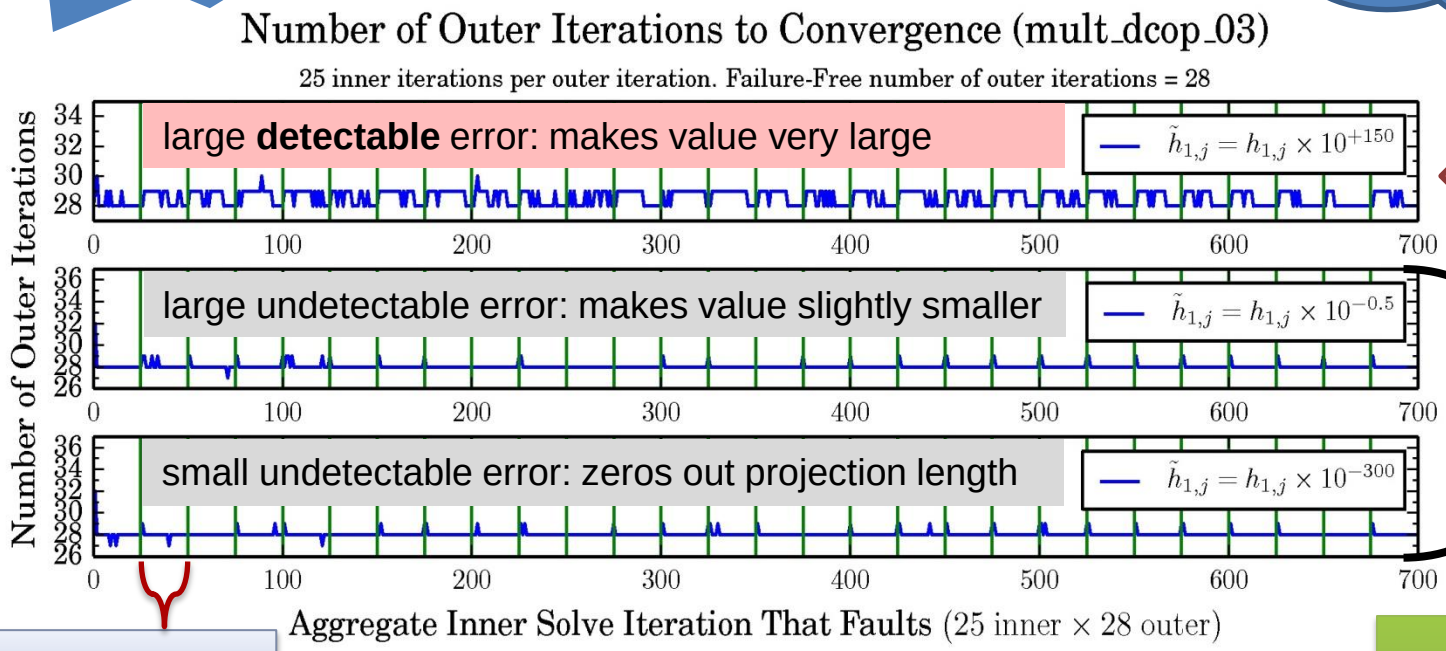
reliable FGMRES

unreliable GMRES

Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Inject ***single*** fault
Study impact

700
experiments per
fault type



Larger than bound

Smaller than bound

one inner solve

Failure Free: **700 orthgonalization calls**
28 outer iterations to converge
25 inner iterations per outer

reliable FGMRES

unbounded error = high variability in time to solution
bounded error = low variability, in time to solution

Prototype Design: Fault Tolerant GMRES iterative solver
Flexible GMRES (FGMRES) creates sandbox
Implemented using Trilinos

Future [Current] Work

Sandia National Laboratories

- Reactive approaches
  - Rollback solver and restart
  - Abort solver (assumes some outer layer is there to save us)
  - Patch current solver iteration (rollback no restart)
- Theoretical
  - Bound arbitrary preconditioners, e.g., $\|\mathbf{w}_0\| = \|\mathbf{A}\mathbf{M}^{-1}\mathbf{q}_j\|$
  - Evaluate nested solvers vs. restarted solvers
    - Skeptical Programming enhances both
- Reliable solver design
  - Redundancy and iterative methods
    - Running a solver twice may not provide any resilience, e.g.,
      - a fault may manifest as longer run time or stagnation rather than incorrect result
    - Smarter redundancy
      - Orthogonalize twice vs. redundant orthogonalizations (use theoretical bounds to vote)

Conclusion

Sandia National Laboratories

- **Be adversarial** – silent faults may be rare, but are expected to increase
- Focus on **theoretically sound approaches** – so we not only get the correct answer, but have reason to believe scientists with different data will also get the correct answer!
- Determine bounds, and study worst-case error injection
- We can make progress…

even if the universe is unreliable!

Questions/Comments: {jjellio,mhoemme}@sandia.gov mueller@cs.ncsu.edu

Papers: http://arxiv.org/find/cs/1/au:+Elliott_J

http://www4.ncsu.edu/~jjellio3/

Acknowledgements

This research was supported by the Consortium for Advanced Simulation of Light Water Reactors (http://www.casl.gov), an Energy Innovation Hub (http://www.energy.gov/hubs) for Modeling and Simulation of Nuclear Reactors under U.S. Department of Energy Contract No. DE-AC05-00OR22725.

This work was supported in part by grants from National Science Foundation (awards 1058779 and 0958311).

Sandia National Laboratories is a multiprogram 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.