Paris
Jan. 13, 2015

Numerical resilience in linear algebra

HiePACS Inria Project
INRIA Bordeaux Sud-Ouest

Context

- HPC systems are not fault-free
- A faulty components (node, core, memory) loses all its data
- Simulations at exascale have to be resilient

Resilience: Ability to compute a correct output in presence of faults

- Context: Numerical linear algebra
- Goal: Keep converging in presence of fault
- Method: Recover-restart strategy without Checkpoint

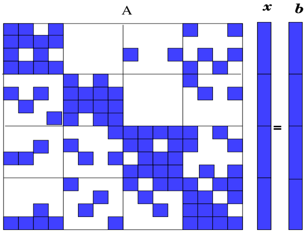
Framework

Objectives

- Explore fault-tolerant schemes with less/no overhead
- Numerical algorithms to deal with overhead issue

Faults in this work

- Detected corrupted memory space (node crashes, damaged memory pages, uncorrected bit-flip, . . .)



$$Ax = b$$

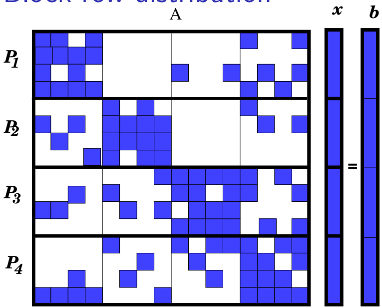
We attempt to design fault tolerant solver for sparse linear system

Two classes of iterative methods

- Stationary methods (Jacobi, Gauss-Seidel, ...)
- Krylov subspace methods (CG, GMRES, Bi-CGStab, ...)

- Krylov methods have attractive potential for Extreme-scale

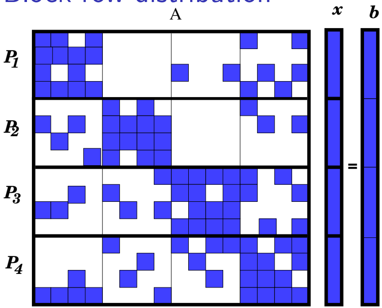
Block row distribution



We distinguish two categories of data:

- Static data
- Dynamic data

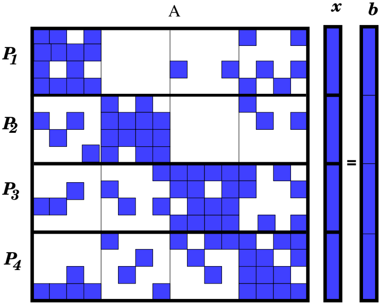
Block row distribution



We distinguish two categories of data:

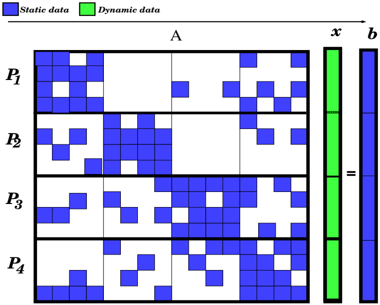
- Static data
- Dynamic data

Block row distribution



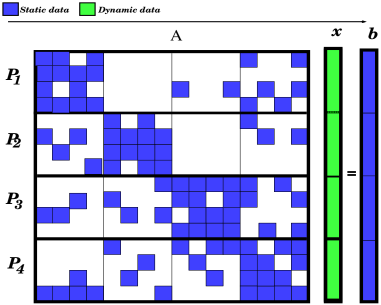
We distinguish two categories of data:

- Static data
- Dynamic data



We distinguish two categories of data:

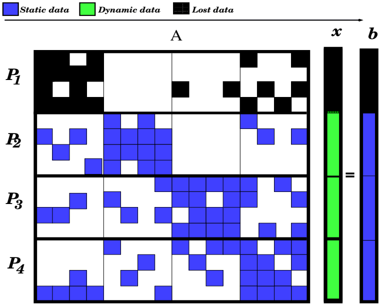
- Static data
- Dynamic data



We distinguish two categories of data:

- Static data
- Dynamic data

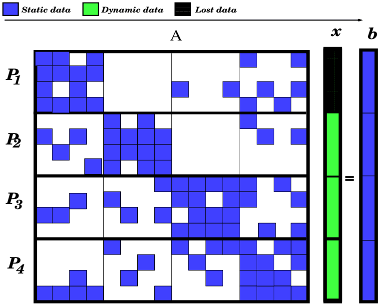
Let's assume that P_1 fails



We distinguish two categories of data:

- Static data
- Dynamic data

Let's assume that P_1 fails

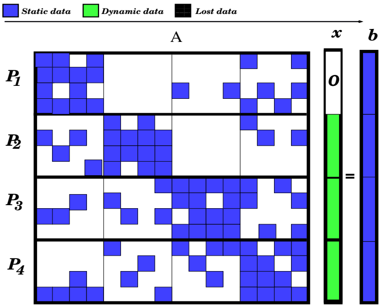


We distinguish two categories of data:

- Static data
- Dynamic data

Let's assume that P_1 fails

- Failed processor is replaced
- Static data are restored



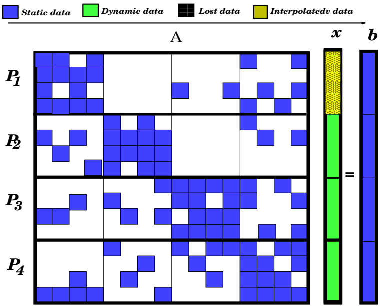
We distinguish two categories of data:

- Static data
- Dynamic data

Let's assume that P_1 fails

- Failed processor is replaced
- Static data are restored

Reset: Set (x_1) to initial value



We distinguish two categories of data:

- Static data
- Dynamic data

Let's assume that P_1 fails

- Failed processor is replaced
- Static data are restored

Our algorithms aim at recovering x_1 and restart

Interpolation methods

Fault in linear system

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

Interpolation methods

Fault in linear system

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} ? \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$ How to recover x_1?

Interpolation methods

Fault in linear system

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} ? \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

How to recover x_1?

Linear Interpolation (LI) [Langou, Chen, Bosilca, Dongarra, SISC, 2007]

Solve $A_{11}x_1 = b_1 - A_{12}x_2$

Interpolation methods

Fault in linear system

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} ? \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$ How to recover x_1?

Linear Interpolation (LI) [Langou, Chen, Bosilca, Dongarra, SISC, 2007]

Solve $A_{11}x_1 = b_1 - A_{12}x_2$

Least Squares Interpolation (LSI)

$$\begin{pmatrix} A_{11} \\ A_{21} \end{pmatrix} x_1 + \begin{pmatrix} A_{21} \\ A_{22} \end{pmatrix} x_2 = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

$$x_1 = \underset{x}{argmin} \left\| \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} - \begin{pmatrix} A_{11} \\ A_{21} \end{pmatrix} x - \begin{pmatrix} A_{12} \\ A_{22} \end{pmatrix} x_2 \right\|_2$$

Impact of fault rate

Preconditioned GMRES (Kim1 - 2 % data lost)

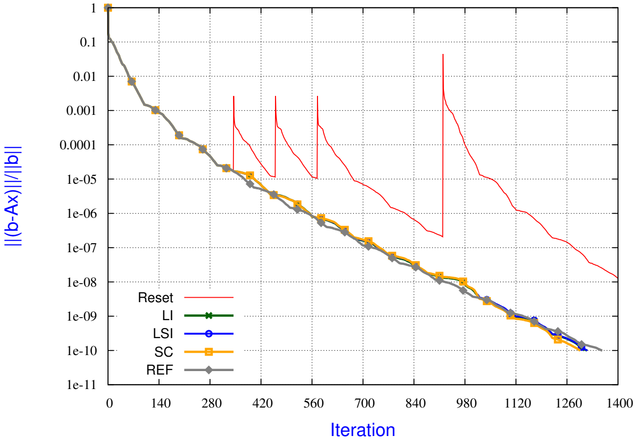


Figure: 4 faults

Impact of fault rate

Preconditioned GMRES (Kim1 - 2 % data lost)

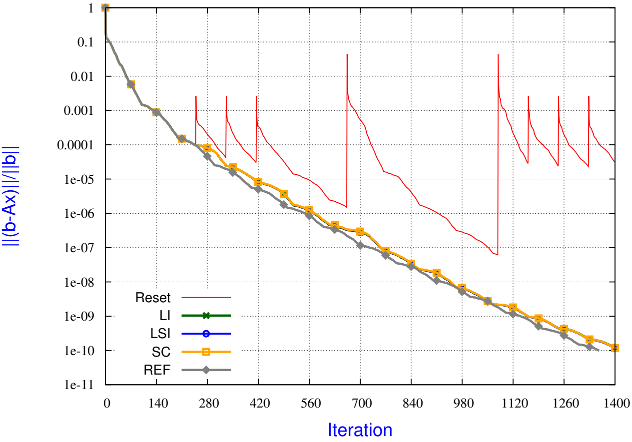


Figure: 8 faults

Impact of fault rate

Preconditioned GMRES (Kim1 - 2 % data lost)

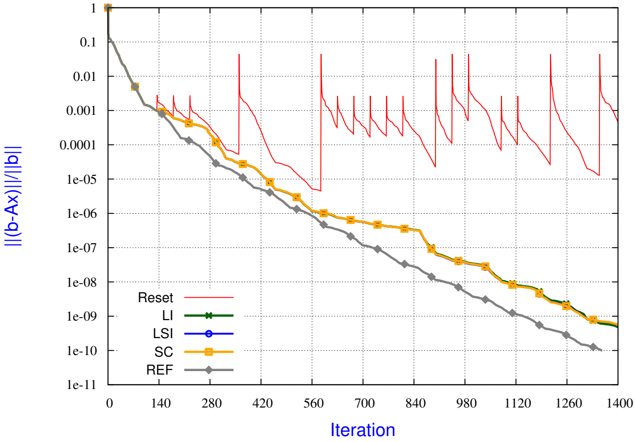


Figure: 17 faults

Impact of fault rate

Preconditioned GMRES (Kim1 - 2 % data lost)

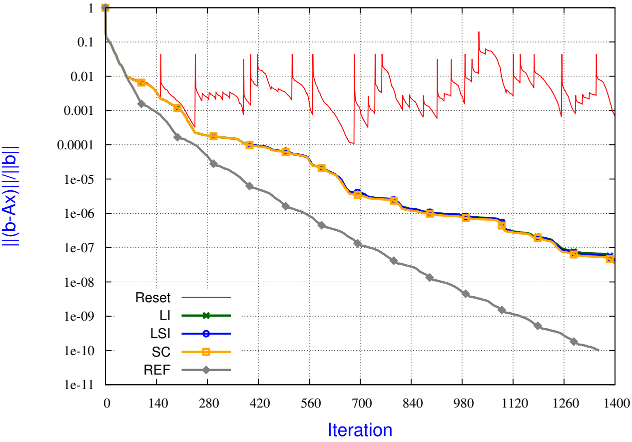


Figure: 40 faults

Impact of lost data volume

Preconditioned GMRES(100) (Averous/epb3 - 10 faults)

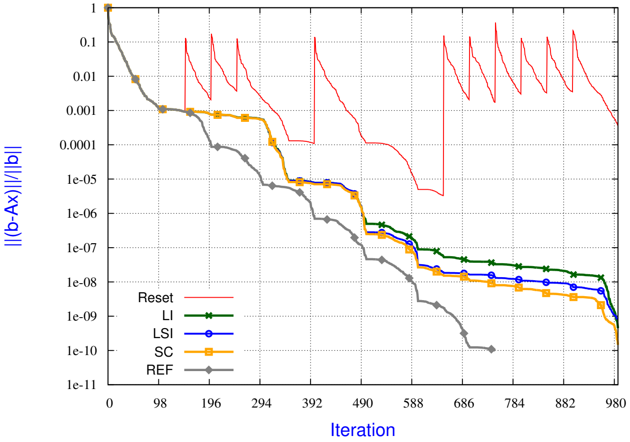


Figure: 3 % data lost

Impact of lost data volume

Preconditioned GMRES(100) (Averous/epb3 - 10 faults)

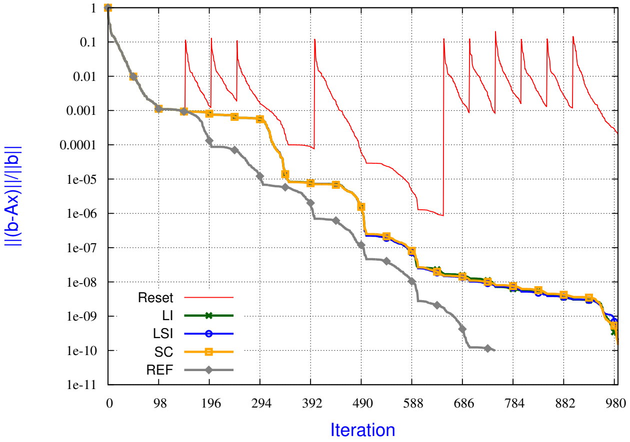


Figure: 0.8 % data lost

Impact of lost data volume

Preconditioned GMRES(100) (Averous/epb3 - 10 faults)

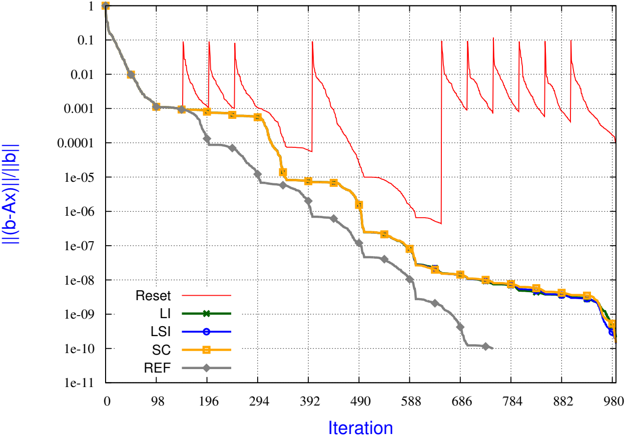


Figure: 0.2 % data lost

Impact of lost data volume

Preconditioned GMRES(100) (Averous/epb3 - 10 faults)

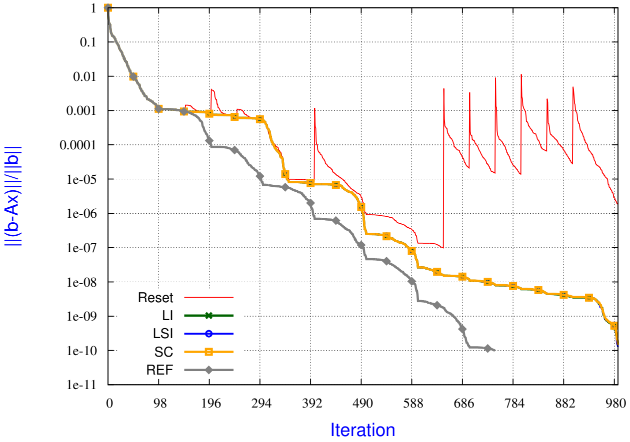


Figure: 0.001 % data lost

Recovery-restart for eigensolvers

Fault in eigenproblem

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \lambda \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

Recovery-restart for eigensolvers

Fault in eigenproblem

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} ? \\ x_2 \end{pmatrix} = \lambda \begin{pmatrix} ? \\ x_2 \end{pmatrix}$$

How to recover x_1?

Linear Interpolation (LI)

Solve the linear system $\left(A_{11} - \lambda I_1\right) x_1 = -A_{12} x_2$

Least Squares Interpolation (LSI)

$$\begin{pmatrix} A_{11} \\ A_{21} \end{pmatrix} x_1 + \begin{pmatrix} A_{21} \\ A_{22} \end{pmatrix} x_2 = \lambda \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$x_1 = \underset{x}{argmin} \left\| \begin{pmatrix} A_{11} - \lambda I_1 \\ A_{21} \end{pmatrix} x + \begin{pmatrix} A_{12} \\ A_{22} - \lambda I_2 \end{pmatrix} x_2 \right\|_2$$

Jacobi-Davidson method

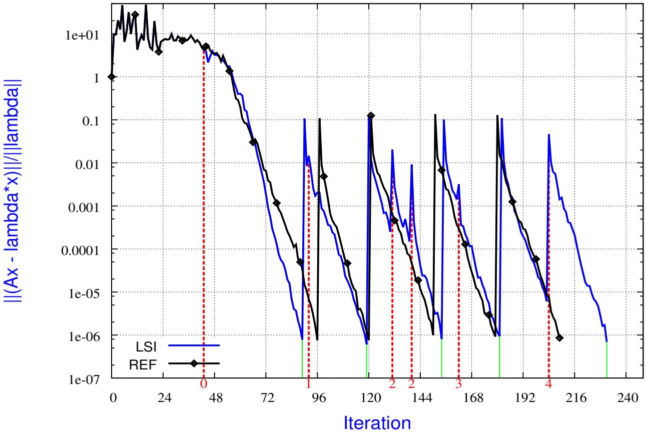


Figure: Jacobi-Davidson method with 5 faults - 1 % lost data. Convergence history using LSI and Checkpoint of current iterate

Concluding remarks

Summary

- Our techniques preserve some of the key monotonicy of Krylov solvers but lack of robustness of LI for non-SPD problems
- The restarting effect remains reasonable within the GMRES context
- No fault, no overhead

Related projects:

- ANR RESCUE (ROMA, Grand Large)
- FP7 Exa2CT (soft error)

Merci for your attention

Questions ?

https://team.inria.fr/hiepacs/