Selective Protection for Sparse Iterative Solvers to Reduce the Resilience Overhead

Hongyang Sun* (Speaker), Ana Gainaru*, Manu Shantharam† , Padma Raghavan*

* Vanderbilt University, USA

† San Diego Supercomputer Center, USA

hongyang.sun@vanderbilt.edu

SBAC-PAD 2020

Scientific Computing at Large Scale

- Today's large High-Performance Computing (HPC) platforms experience multiple failures per day due to increased node/core count.

MTBF (individual node)	1 year	10 years	100 years
MTBF (platform of 10^6 nodes)	30 secs	5 mins	50 mins

* MTBF: Mean Time Between Failure

- Besides hard failures (e.g., fail-stop errors), soft faults (e.g., silent errors or silent data corruptions) also become a major threat.

Thus, protecting HPC applications from hard and soft faults plays a vital role in the integrity and efficiency of scientific computing and simulations.

Sparse Iterative Solvers and PCG

- Solving a sparse linear system:

$$\boldsymbol{Ax} = \boldsymbol{b}$$

  is central to PDE-based applications.

- We focus on the widely used PCG (Preconditioned Conjugate Gradient) algorithm to iteratively solve a sparse linear system.
- We consider system-level resilience techniques to protect PCG from soft errors with low overhead.

Algorithm 1: Preconditioned Conjugate Gradient (PCG)

Input: $A, M, b, x_0, tol, maxit$

begin

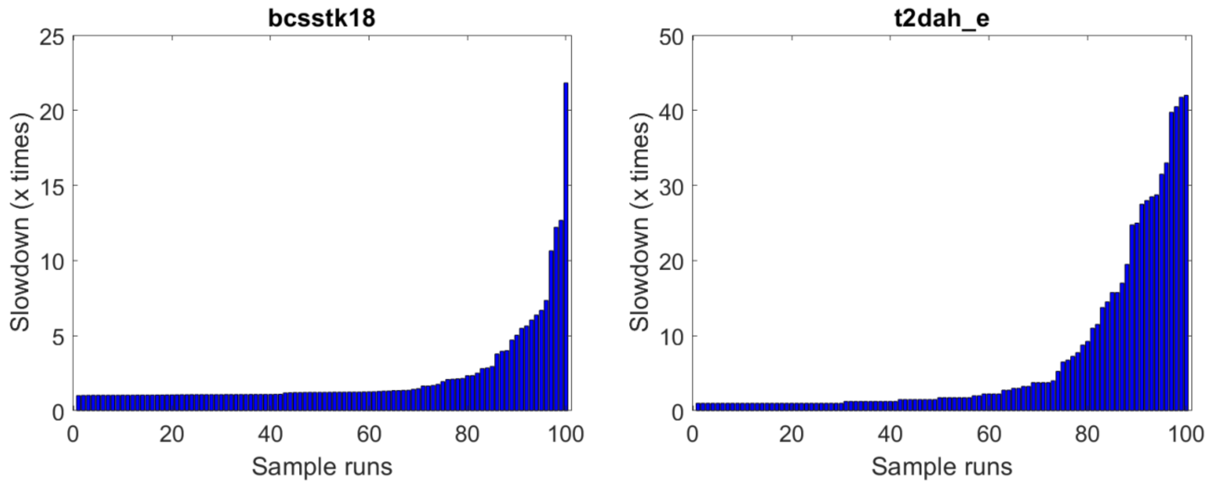
- $r_0 \leftarrow b - Ax_0;$ `// Initial residual`
- $z_0 \leftarrow M^{-1} r_0;$ `// Preconditioning`
- $p_0 \leftarrow z_0;$
- $i \leftarrow 0;$
- **while** $i < maxit$ and $\|r_i\|/\|b\| > tol$ **do**
  - $q_i \leftarrow Ap_i;$
  - $v_i \leftarrow r_i^T z_i;$
  - $\alpha \leftarrow v_i/(p_i^T q_i);$
  - $x_{i+1} \leftarrow x_i + \alpha p_i;$ `// Improve approximation`
  - $r_{i+1} \leftarrow r_i - \alpha q_i;$ `// Update residual`
  - $z_{i+1} \leftarrow M^{-1} r_{i+1};$ `// Preconditioning`
  - $v_{i+1} \leftarrow r_{i+1}^T z_{i+1};$
  - $\beta \leftarrow v_{i+1}/v_i;$
  - $p_{i+1} \leftarrow z_{i+1} + \beta p_i;$ `// New search direction`
  - $i \leftarrow i + 1;$
- **end**

end

Impact of Soft Errors on PCG

- Soft errors have different impacts on the convergence of PCG.
  - SpMV ($q \leftarrow Ap$) is the most expensive operation and most impacted by errors.*
  - Errors injected in different elements of vector p cause very different slowdowns in terms of convergence speed.

Selectively protecting those elements that are more prone to errors can reduce the resilience overhead!



$$\text{Slowdown} = \frac{I_e}{I_o}$$

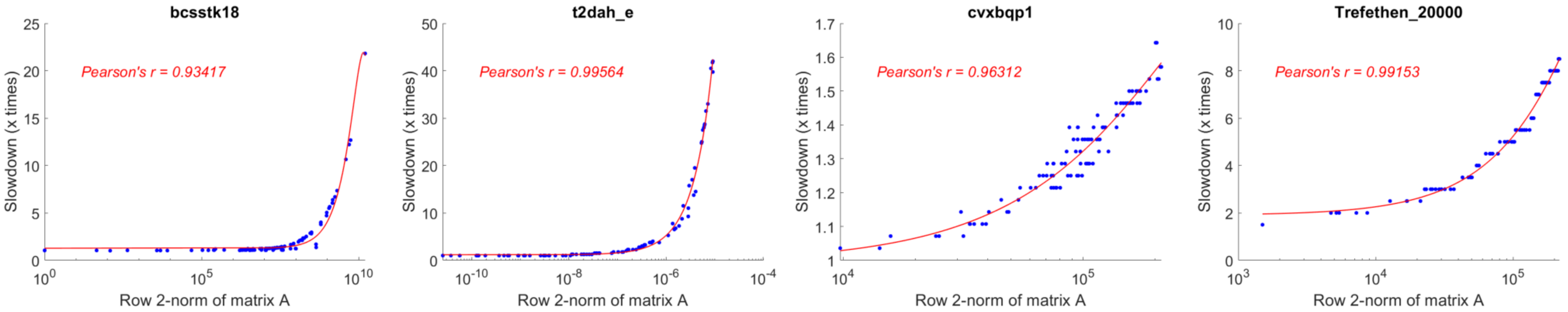
- I_e: # iter to converge w/ errors.
- I_o: # iter to converge w/o errors.

* M. Shantharam, S. Srinivasmurthy, and P. Raghavan. Characterizing the impact of soft errors on iterative methods in scientific computing, 2011.

Performance Characterization

- **Q:** How to identify those elements that are more prone to errors?
- **A:** Row 2-norm of matrix A (which can be computed offline) is strongly correlated with slowdown.

$$\|A_{i*}\|_2 = \sqrt{\sum_{j=1}^{N} A_{i,j}^2} \ .$$

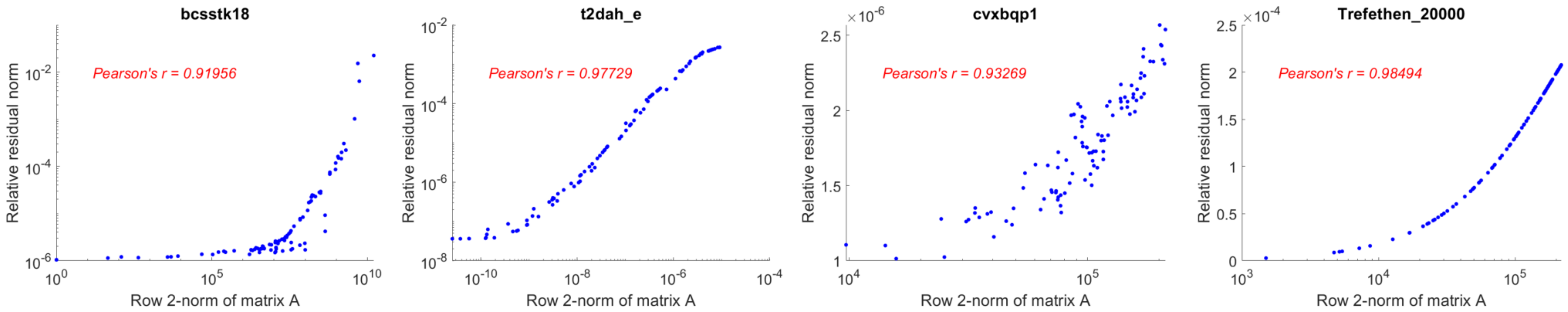


Performance Characterization

- Further, 2-norm of matrix A is strongly correlated two important convergence indicators in PCG (i.e., relative residual norm and A-norm of errors).

$$relative\ residual\ norm = \|r_i\|/\|b\|$$

$$A\text{-}norm\ of\ errors = \sqrt{(x_i - \hat{x})^T A (x_i - \hat{x})}$$

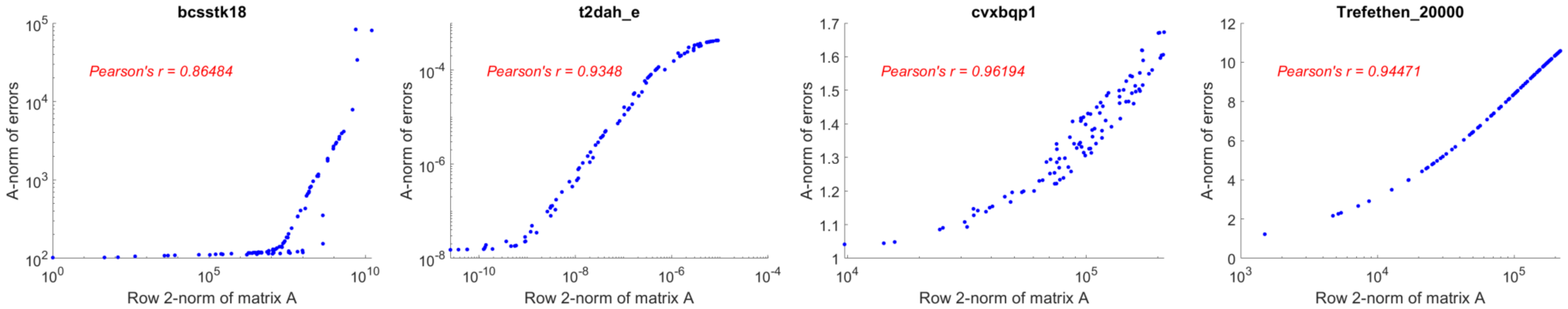


Performance Characterization

- Further, 2-norm of matrix A is strongly correlated two important convergence indicators in PCG (i.e., relative residual norm and A-norm of errors).

$$relative\ residual\ norm = ||r_i||/||b||$$

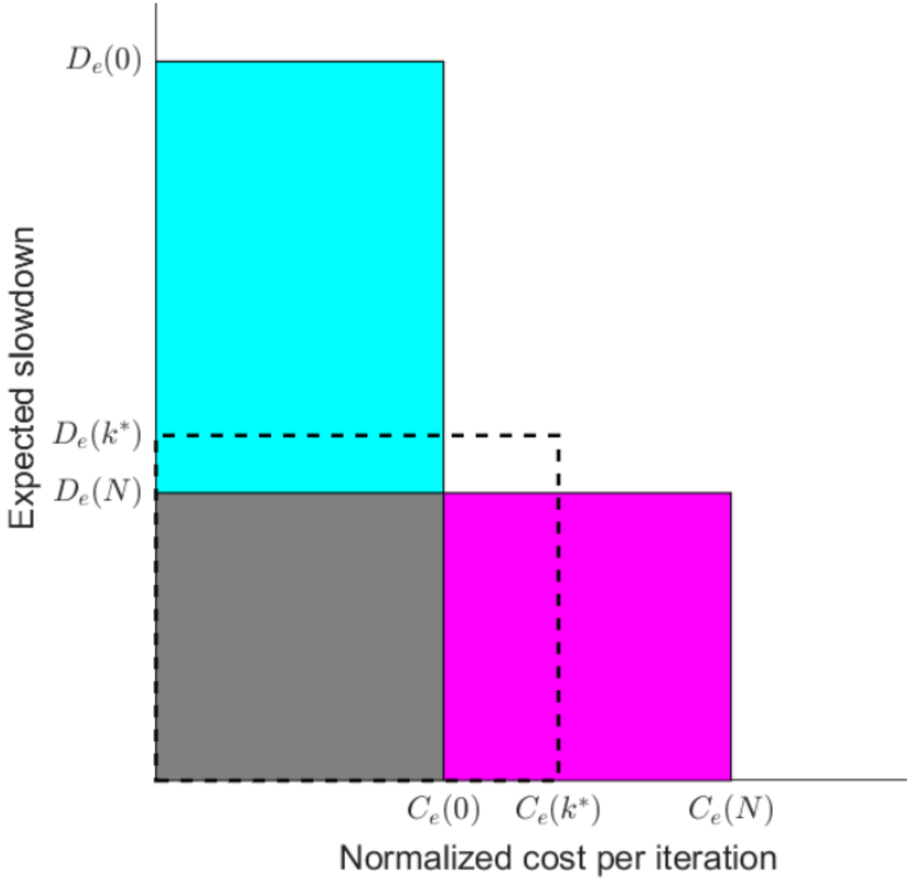
$$A\text{-}norm\ of\ errors = \sqrt{(x_i - \hat{x})^T A (x_i - \hat{x})}$$



Selective Protection Scheme

- Only elements with corresponding high 2-norms in matrix A need to be protected (at system level by duplicate computation), and in the event of soft errors, the iteration can be re-computed.

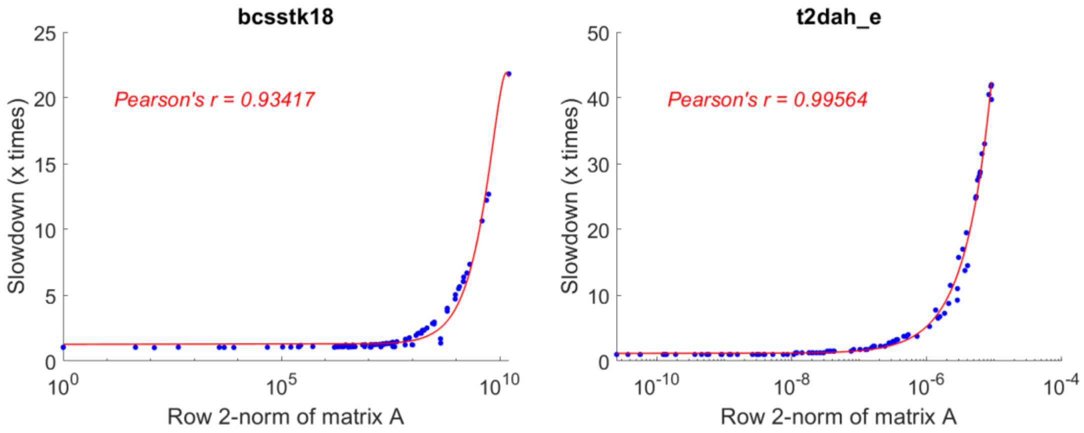
- **Q:** How many elements to protect to optimize the resilience overhead?
  - **Full protection:** 100% overhead but no slowdown (magenta area).
  - **Zero protection:** 0% overhead but large slowdown (cyan area).
  - **Optimal protection:** $x\%$ overhead with a factor of y slowdown (i.e., dashed rectangle with min area)?



Selective Protection Scheme

1. Performance Prediction

- Profile matrix A with a small number *m* of sample runs (e.g., *m = 20*) by injecting errors in random elements.
- Use polynomial regression to fit a function *F* that maps row 2-norm of matrix A to convergence slowdown.



2. Analytical Modeling

- Expected slowdown (by protecting k elements with highest row 2-norms):

$$D_e(k) = \frac{1}{N}\left(k + \sum_{i=k+1}^{N} F(a_{\sigma(i)})\right)$$

- Normalized cost per iteration:

$$C_e(k) = 1 + \frac{k}{N}$$

- Expected overhead can be minimized offline in *O(N)* time with the following analytical model:

$$H_e(k) = D_e(k) \cdot C_e(k) - D_o \cdot C_o$$
$$= \frac{1}{N}\left(k + \sum_{i=k+1}^{N} F(a_{\sigma(i)})\right)\left(1 + \frac{k}{N}\right) - 1$$

Experimental Setup

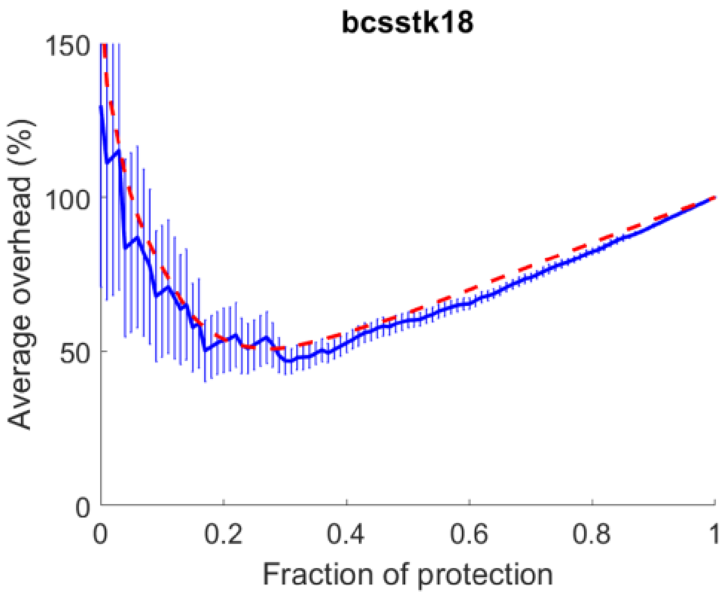
- **Matrices:**
  - ❑ 20 sparse matrices selected from the SuiteSparse Matrix Collection.
- **PCG algorithm:**
  - ❑ Incomplete Cholesky factorization as preconditioner with threshold dropping (10^{-3}).
  - ❑ Initial guess $x_0 = \mathbf{0}$ (all zeros).
  - ❑ RHS vector $b = A \cdot \mathbf{1}$.
- **Soft Errors:**
  - Injected in vector p of SpMV.
  - Random magnitude in 1st iteration.
- Experiments conducted in Matlab and results averaged over 100 runs.

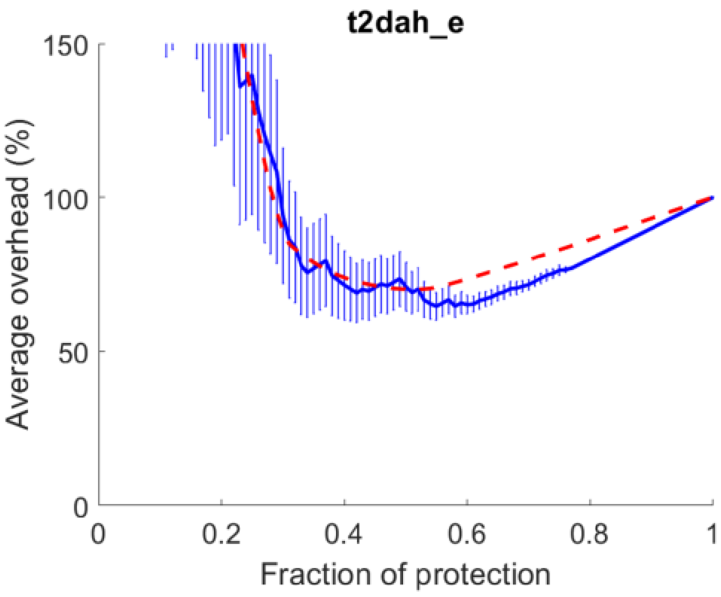
Table I. 20 matrices from the SuiteSparse Matrix Collection [1].

Id	*Matrix*	*N*	*nnz*	*Density*
1	t2dah_e	11445	176117	0.13%
2	bcsstk18	11948	149090	0.1%
3	cbuckle	13681	676515	0.36%
4	Pres_Poisson	14822	715804	0.33%
5	gyro_m	17361	340431	0.11%
6	nd6k	18000	6897316	2.1%
7	bodyy5	18589	128853	0.037%
8	raefsky4	19779	1316789	0.34%
9	Trefethen_20000	20000	554466	0.14%
10	msc23052	23052	1142686	0.22%
11	bcsstk36	23052	1143140	0.22%
12	wathen100	30401	471601	0.051%
13	vanbody	47072	2329056	0.11%
14	cvxbqp1	50000	349968	0.014%
15	ct20stif	52329	2600295	0.095%
16	thermal1	82654	574458	0.0084%
17	m_t1	97578	9753570	0.1%
18	2cubes_sphere	101492	1647264	0.016%
19	G2_circuit	150102	726674	0.0032%
20	pwtk	217918	11524432	0.024%

Experimental Results

1. *Our performance prediction and analytical models accurately capture the resilience overhead of various fractions of selective protection.*

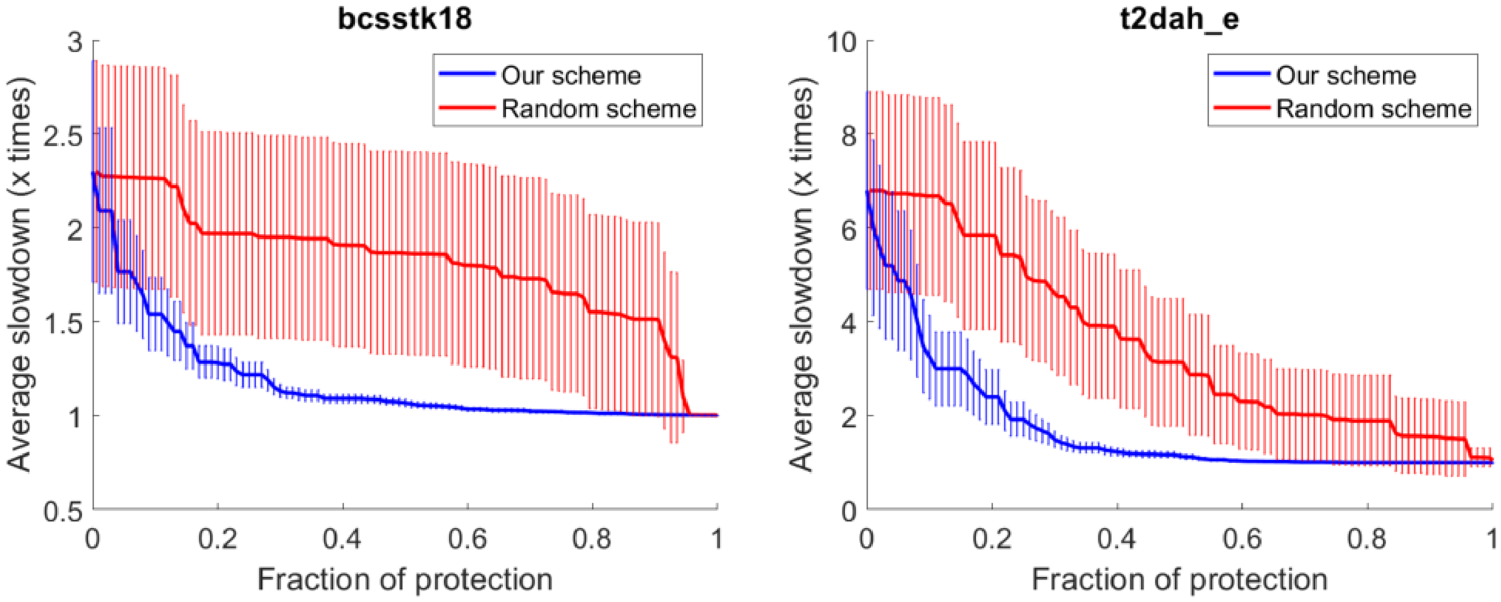




- **red --- line: predicted overhead using our analytical model.**
- **Blue — line: average experimental overhead with 95% confidence interval.**

Experimental Results

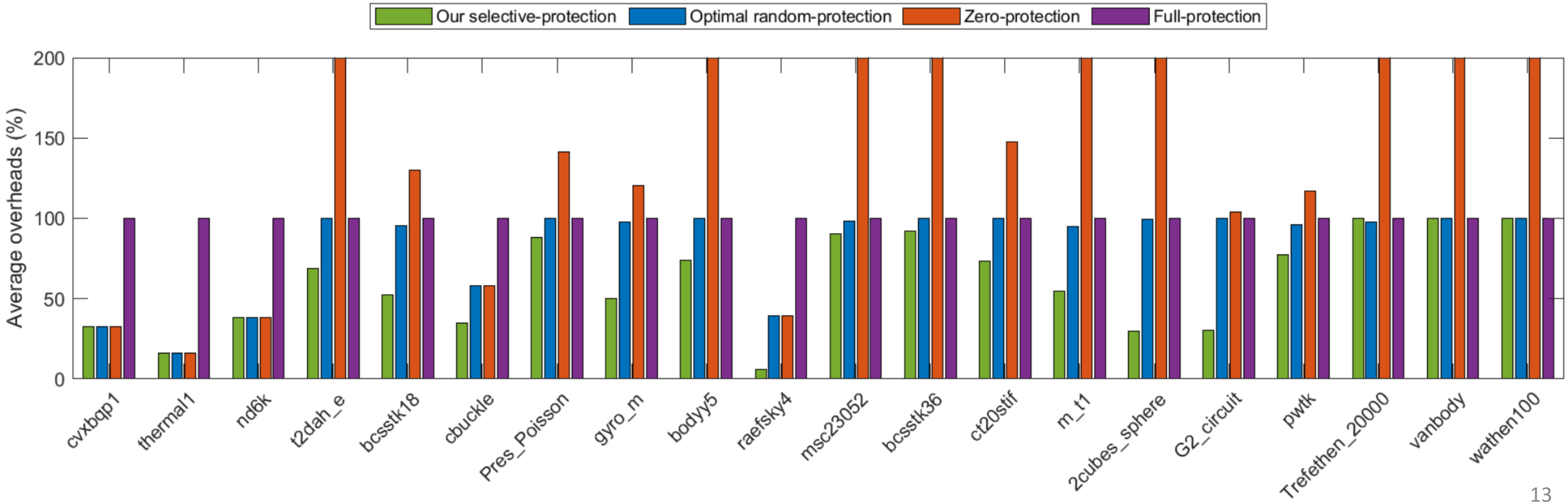
2. *Our selective protection scheme is more effective and targeted than a random selective protection strategy* in terms of reducing the average slowdown (and variance) in the event of soft errors.*



* J. Sloan, R. Kumar, and G. Bronevetsky. Algorithmic approaches to low overhead fault detection for sparse linear algebra, 2012.

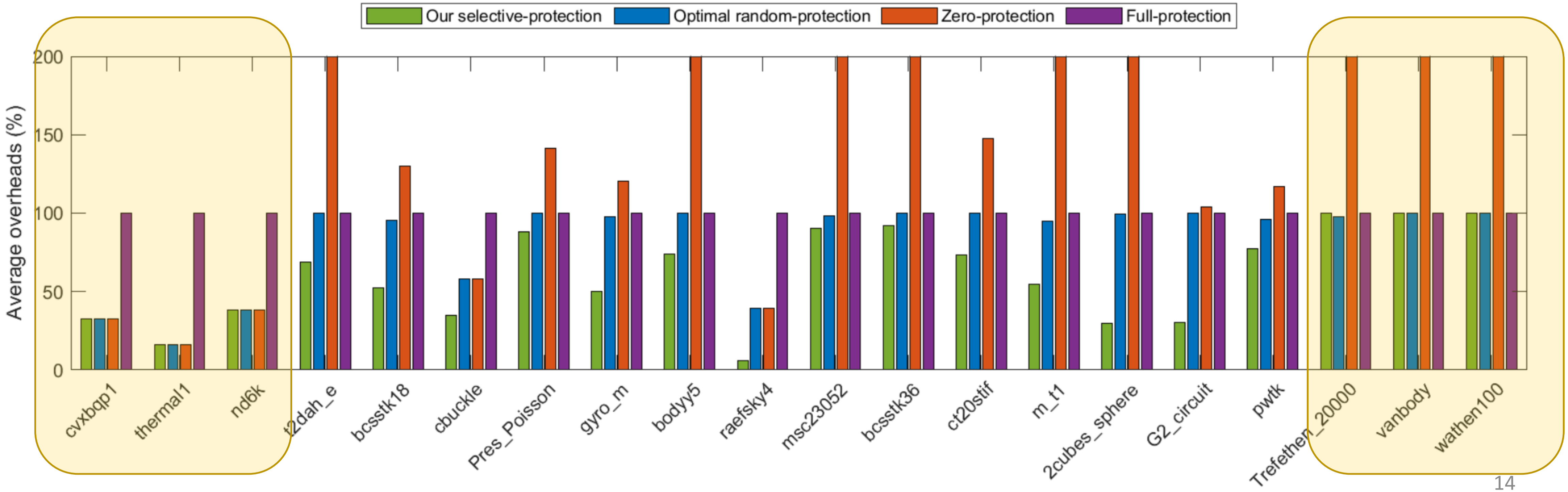
Evaluation Results

3. *Our selective protection scheme significantly reduces resilience overhead (by 32.6% on average and up to 70.2%) for the set of matrices compared to baseline schemes (i.e., full protection, zero protection, random protection).*



Evaluation Results

4. *For some matrices (e.g., left-most three), zero-protection performs equally well, due to the small impact of soft errors on all elements.*
5. *For some other matrices (e.g., right-most three), full-protection performs equally well, due to the large impact of soft errors on most elements.*



Summary

- Soft errors have very different impacts on the convergence of PCG.
- The slowdown caused by a soft errors strongly correlates with corresponding row 2-norm of underlying sparse matrix A.
- Our selective protection scheme (performance prediction + analytical modeling) significantly reduces the resilience overhead.

Future Work

- Selective protection for other iterative solvers than PCG (e.g., GMRES).
- Application-level protection instead of system level (e.g., ABFT).
- Variable error rates and implication on selective protection.