Toward a More Robust Sparse Solver

…with some ideas on resilience and scalability

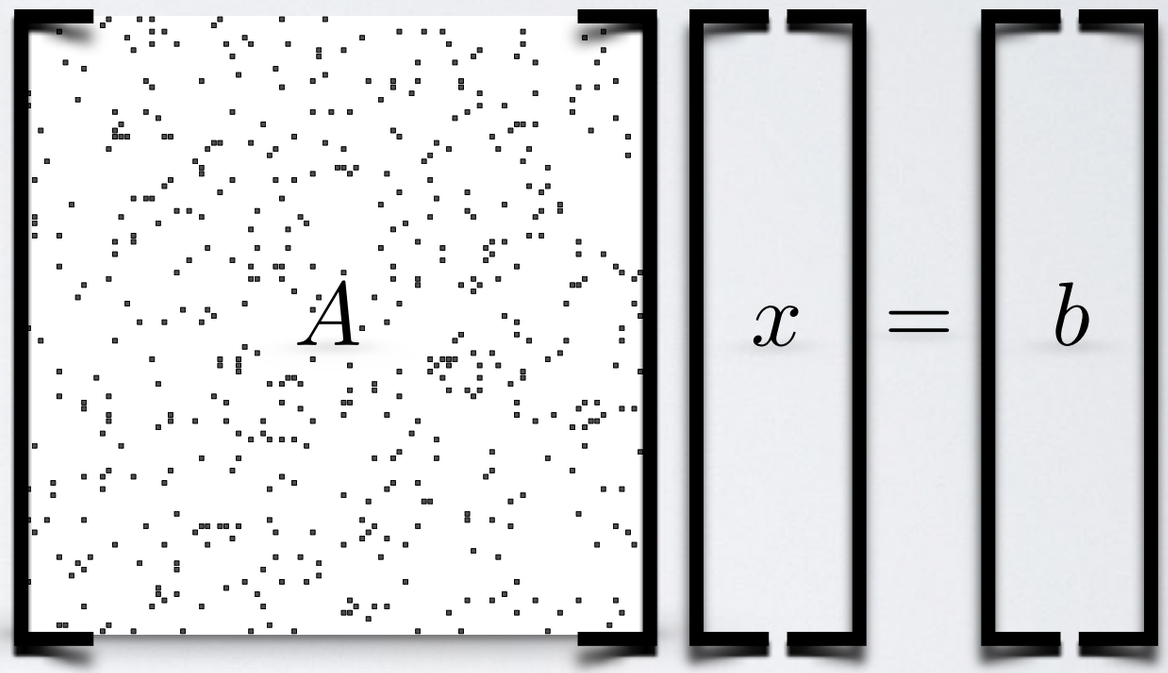
Luke Olson
CS @ Illinois

The Point

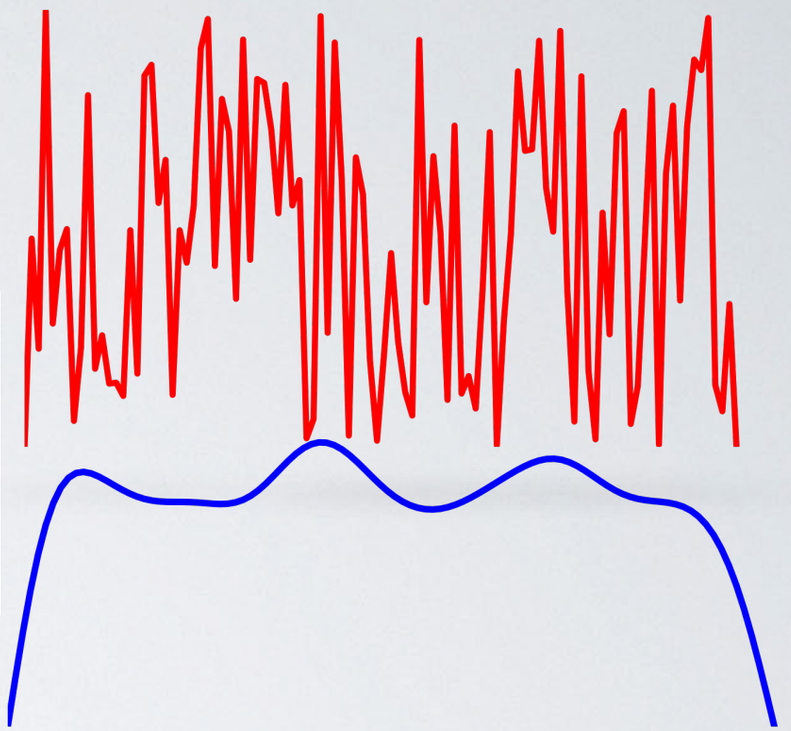
- The state of algebraic solvers (AMG)
- Thinking about MG and resilience
- Collaboration in the Joint Lab to accelerate this

Sparse Problem

- complex, non-symmetric
- non-elliptic
- preconditioning necessary
- encompass a large solve time



Multilevel



attenuate **high energy** quickly with with relaxation

attenuate **low energy** error through coarse-grid correction

relax $x_0 \leftarrow x_0 + \omega D^{-1} r_0$

$x_0 \leftarrow x_0 + \bar{e_0}$ **correct**

restrict $r_c = P^T r_0$

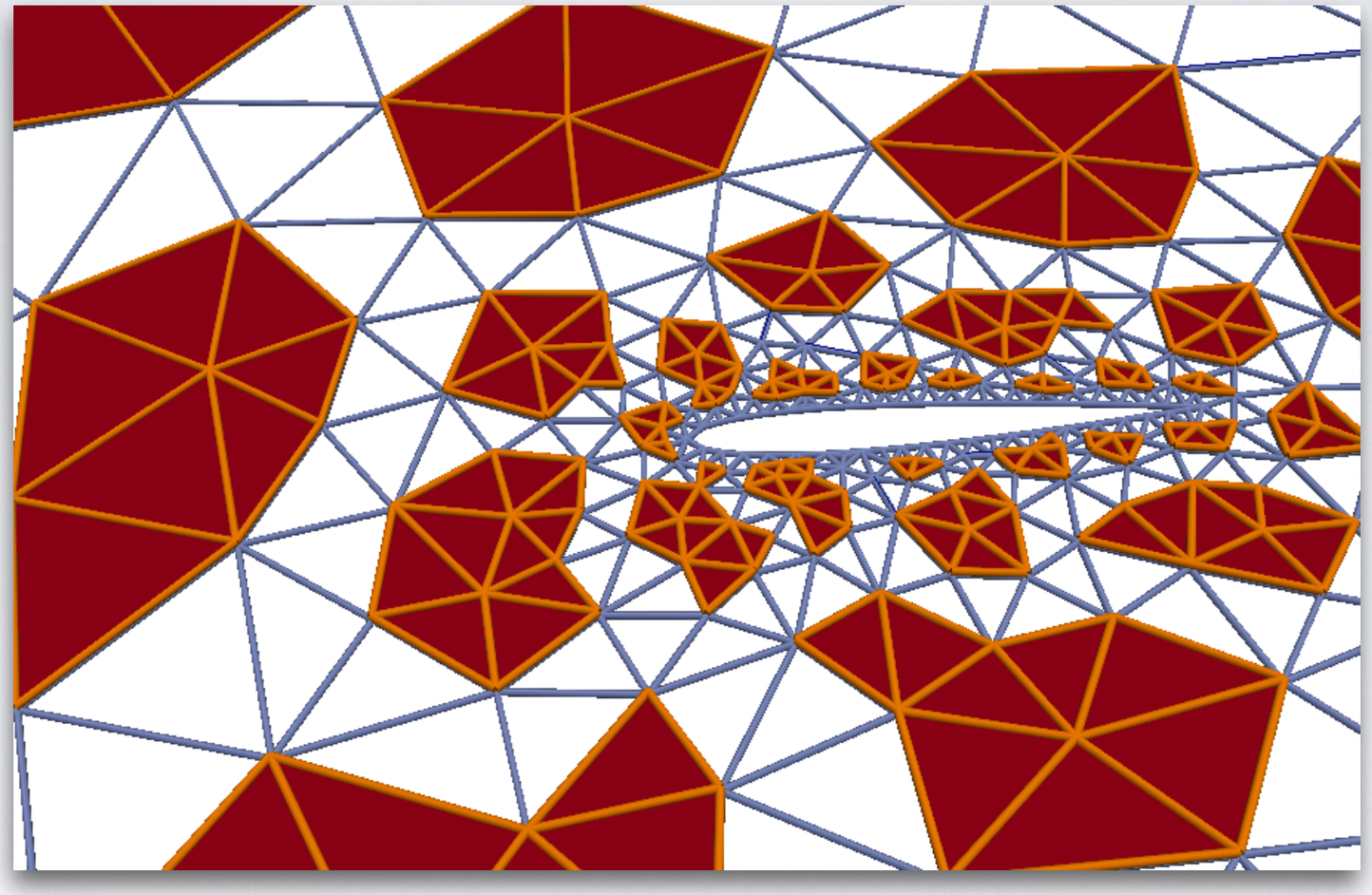
$\bar{e_0} = P e_c$ **interpolate**

$(P^T A P) e_c = r_c$

coarse solve

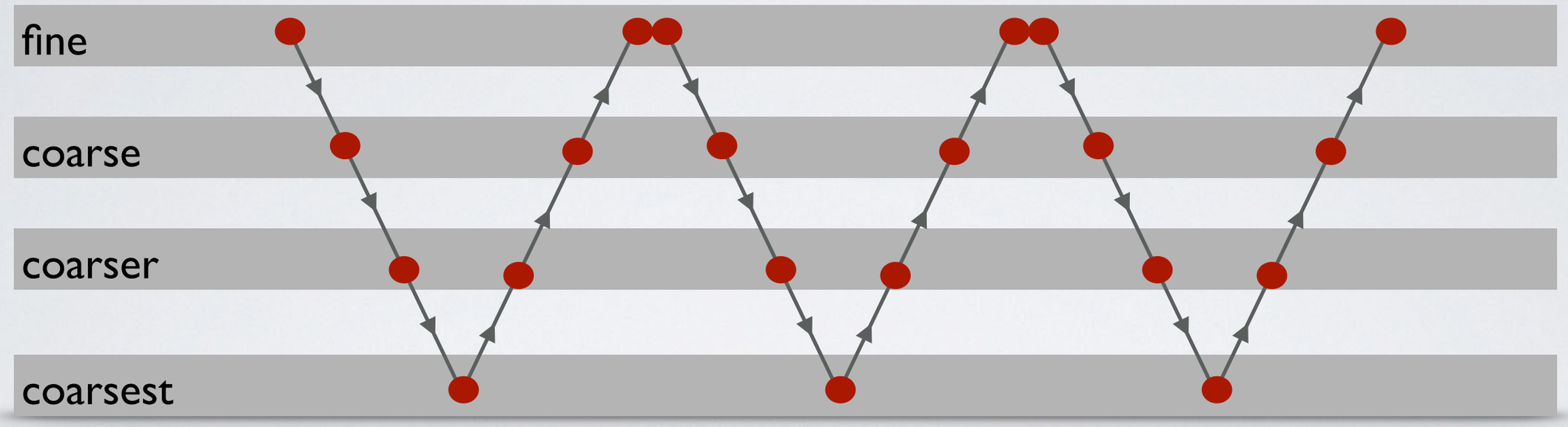
need P

Multilevel



1. Determine sense of **energy**
2. Find **strength** of edges
3. **Group** edges
4. Form **interpolation**

Interpolation



$$e_1 \leftarrow (I - P(P^T AP)^{-1} P^T A) G e_0$$

relax
residual
restrict
coarse solve
interpolate
correct

Interpolation

$$e_1 \leftarrow (I - P(P^T A P)^{-1} P^T A) G e_0$$

coarse grid correction — relax

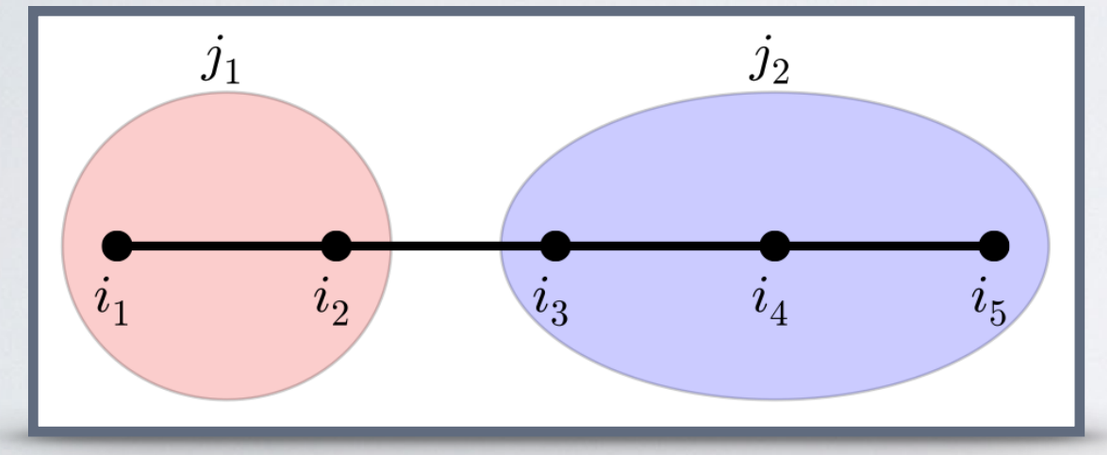
$$Ge_0 \in \mathcal{R}(P) \quad \Rightarrow \quad e_1 = 0$$

interpolation should capture what relaxation misses

- P should have low energy (low A-norm or A^*A-norm)
  1. determine sparsity pattern
  2. minimize energy column-wise (parallel)

Interpolation (basic)

- Set the sparsity pattern from aggregation



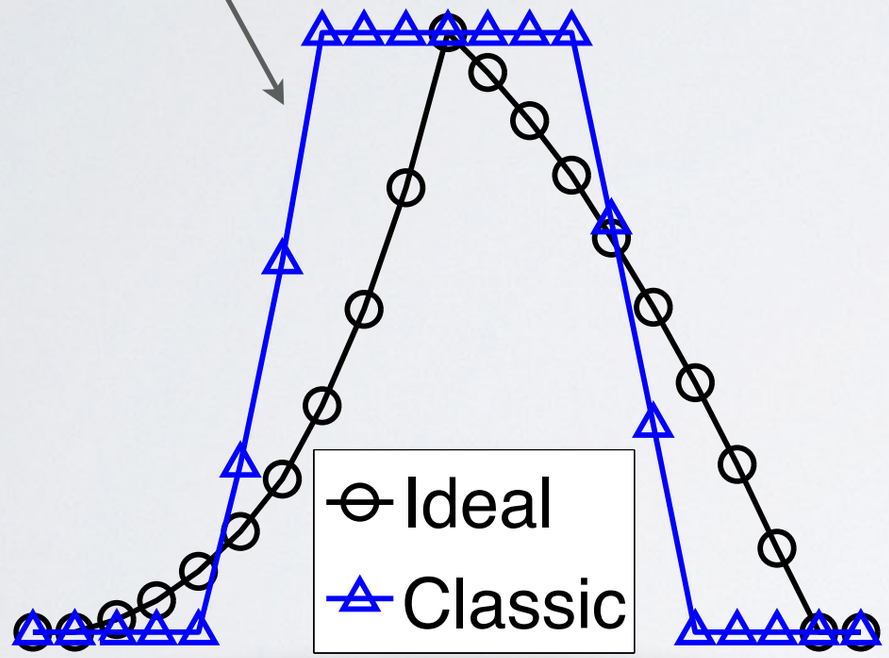
$$P = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

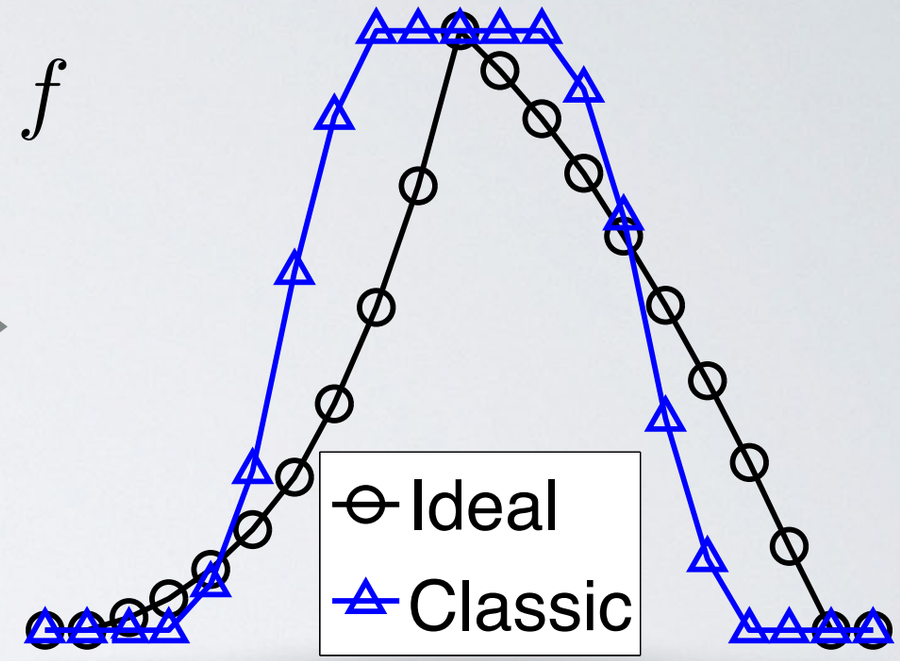
$$P \leftarrow (I - \omega D^{-1} A) P$$

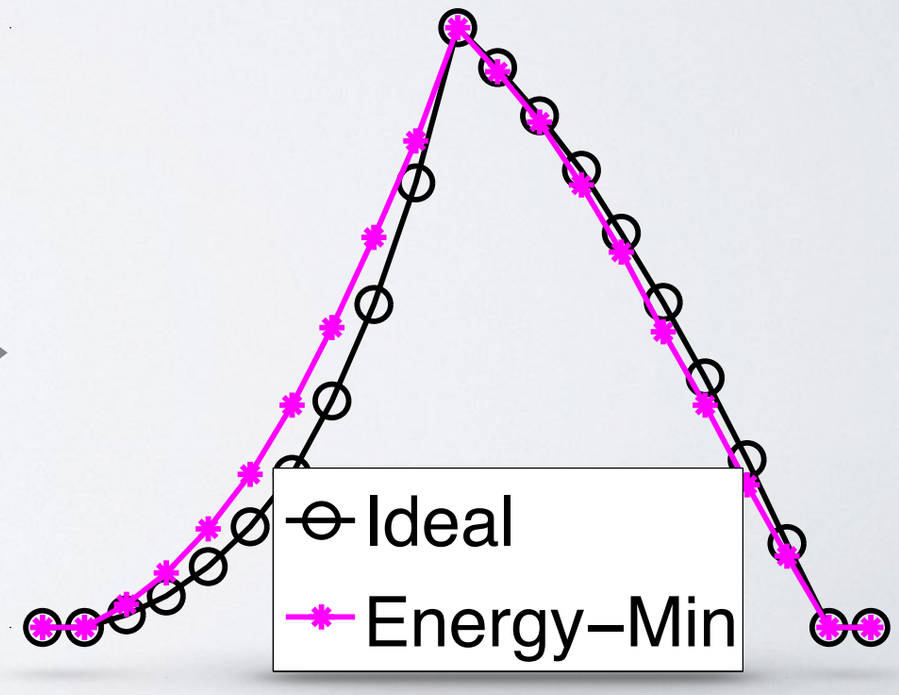
- reduce energy
- improve accuracy
- increase complexity

Range of Interpolation

$$-u_{xx} + \sin(x)u_x = f$$







Toward General Interpolation

- Want P so that $u_{low} \in \mathcal{R}(P)$

- Grow sparsity pattern along strong edges:

$$S^k P_{tent}$$

- Minimize residual:

$$AP_j = 0 \quad \text{for each column } j$$

- Constraint the minimization with

$$Pu_{low}^c = u_{low}$$

Toward General Interpolation

- Hermitian (and positive definite): use CG

$$AP_j = 0 \Leftrightarrow \min \|P_j\|_A$$
$$R = P^*$$

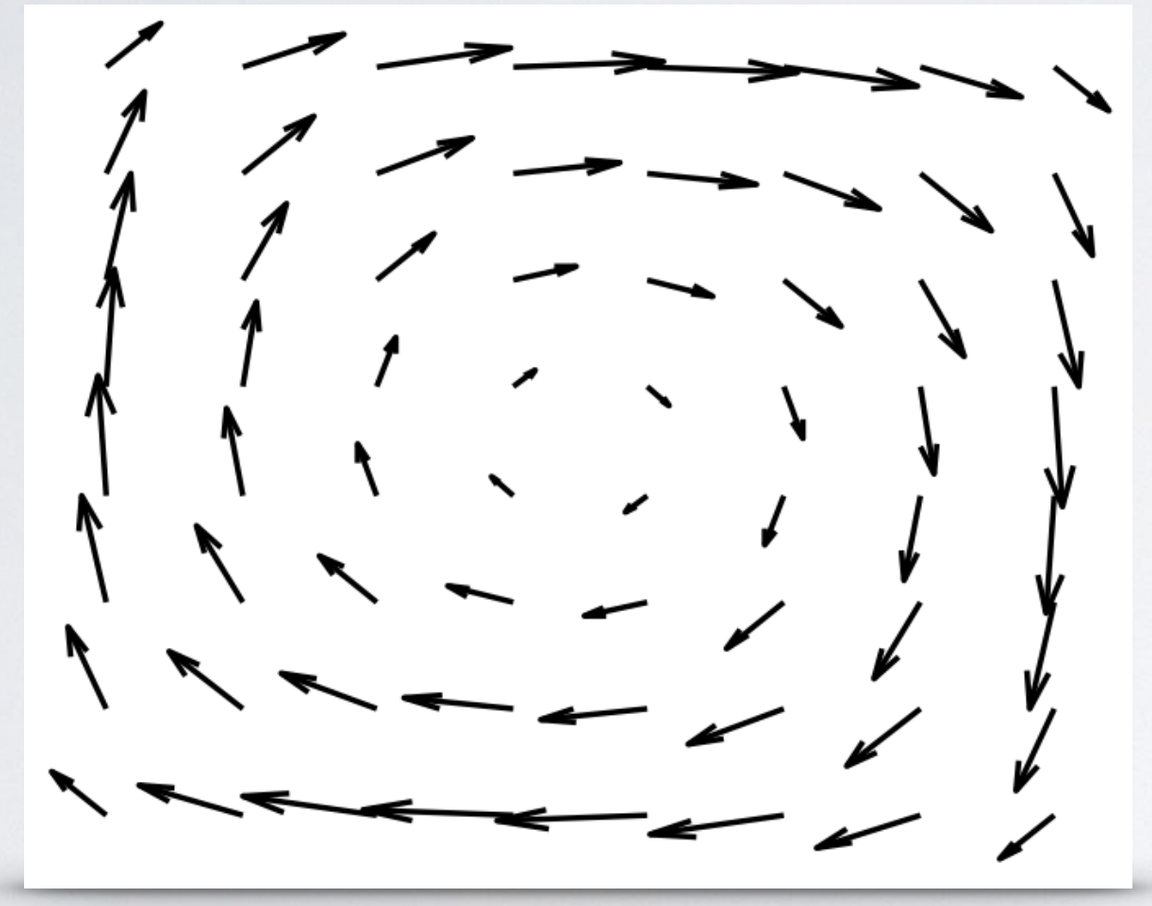
- Non-Hermitian: use GMRES

$$AP_j = 0 \Leftrightarrow \min \|P_j\|_{A^*A}$$
$$A^*R_j^* = 0 \Leftrightarrow \min \|R_j^*\|_{AA^*}$$

- Range of interpolation/restriction target "right"/"left" low-energy

- Cost is comparable to that of standard smoothing

Example: recirculating flow



h	std.	opt.
1/64	>150	24
1/128	>150	28
1/256	>150	33
1/512	>150	33

iterations

What interpolation gives us

- Sense of optimality (or a sense of *violating* optimality)

$$u \in \mathcal{R}(P) \quad \Rightarrow \quad \|u\|_A \ll 1$$

- Energetic stability

$$\|P(P^T P)^{-1} P^T\|_A$$

uniformly bounded gives convergence

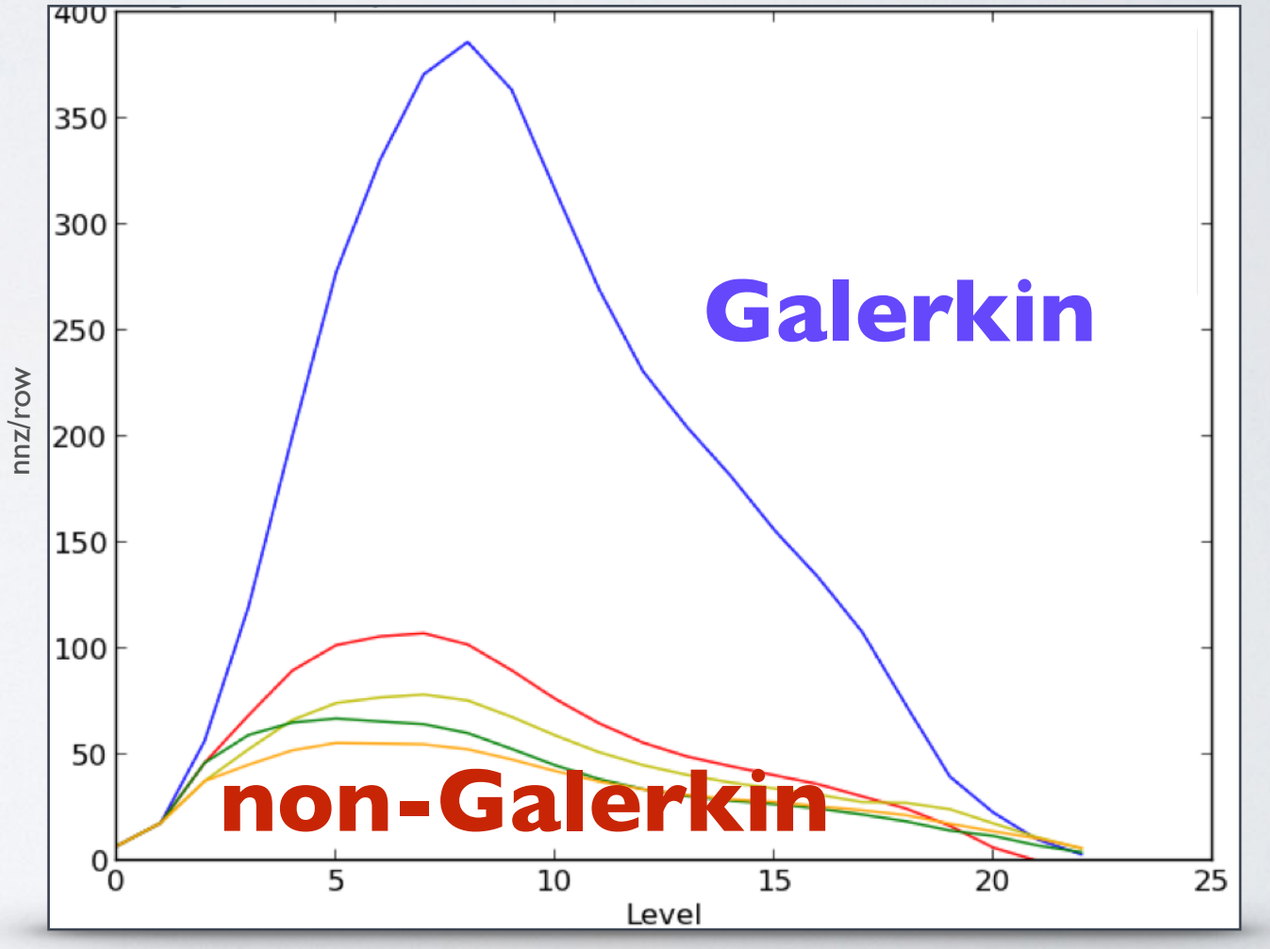
Non-Galerkin Coarsening

- Galerkin coarsening

$$A_{\text{coarse}} = P^T A_{\text{fine}} P$$

- High growth in non-zeros per row:
  - high complexity
  - high data dependence between compute nodes

10K d.o.f. per node
512 nodes (12 cores)



average non-zeros per row

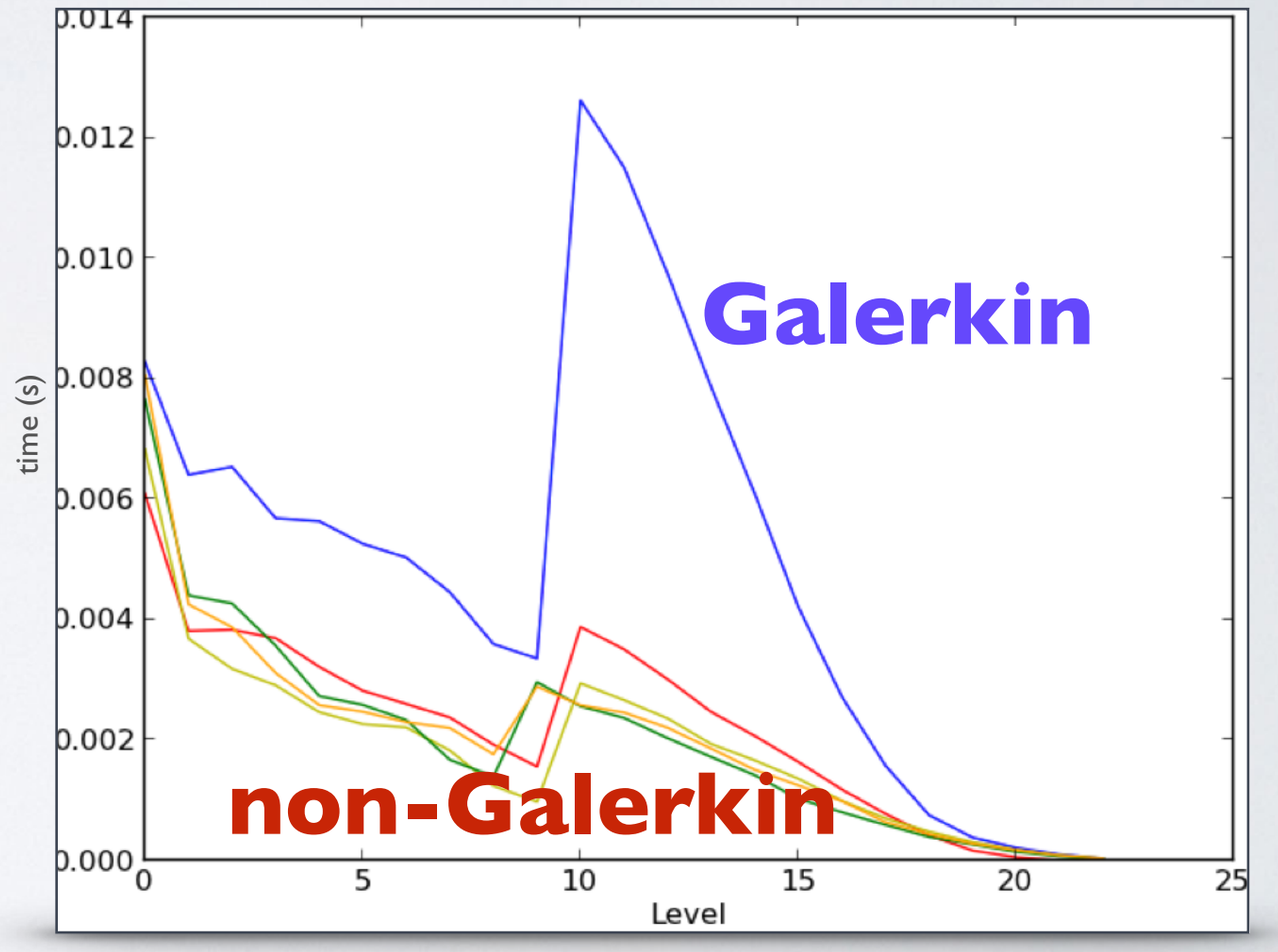
Non-Galerkin Coarsening

- Galerkin coarsening

$$A_{\text{coarse}} = P^T A_{\text{fine}} P$$

- High growth in non-zeros per row:
  - high complexity
  - high data dependence between compute nodes

10K d.o.f. per node
512 nodes (12 cores)



time per level

Non-Galerkin Coarsening

- Basic idea* is to take

$$A_c = P^T A_f P$$

$$\|I - \hat{A}_c^{-1} A_c\| \rightarrow min$$

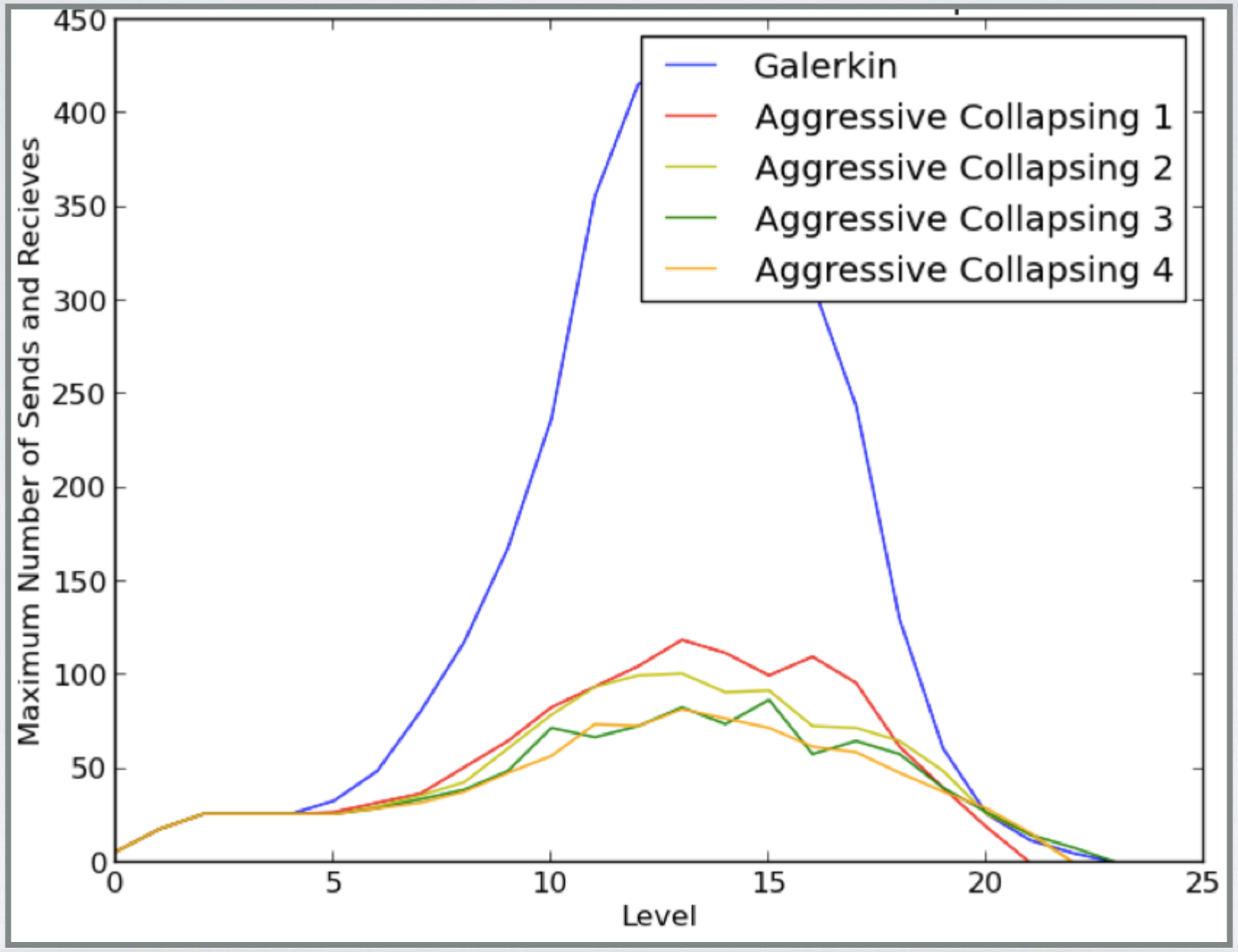
* Falgout, Schroder. *Non-Galerkin Coarse Grids for Algebraic Multigrid*. LLNL 2013.

- $\hat{A}_c$ has improved sparsity

1. Determine a sparsity pattern

2. Enforce collapsing to maintain spectral equivalence

Non-Galerkin Coarsening



- potential: lower corruption, few (potential) faults in MPI

Why think about MG resilience

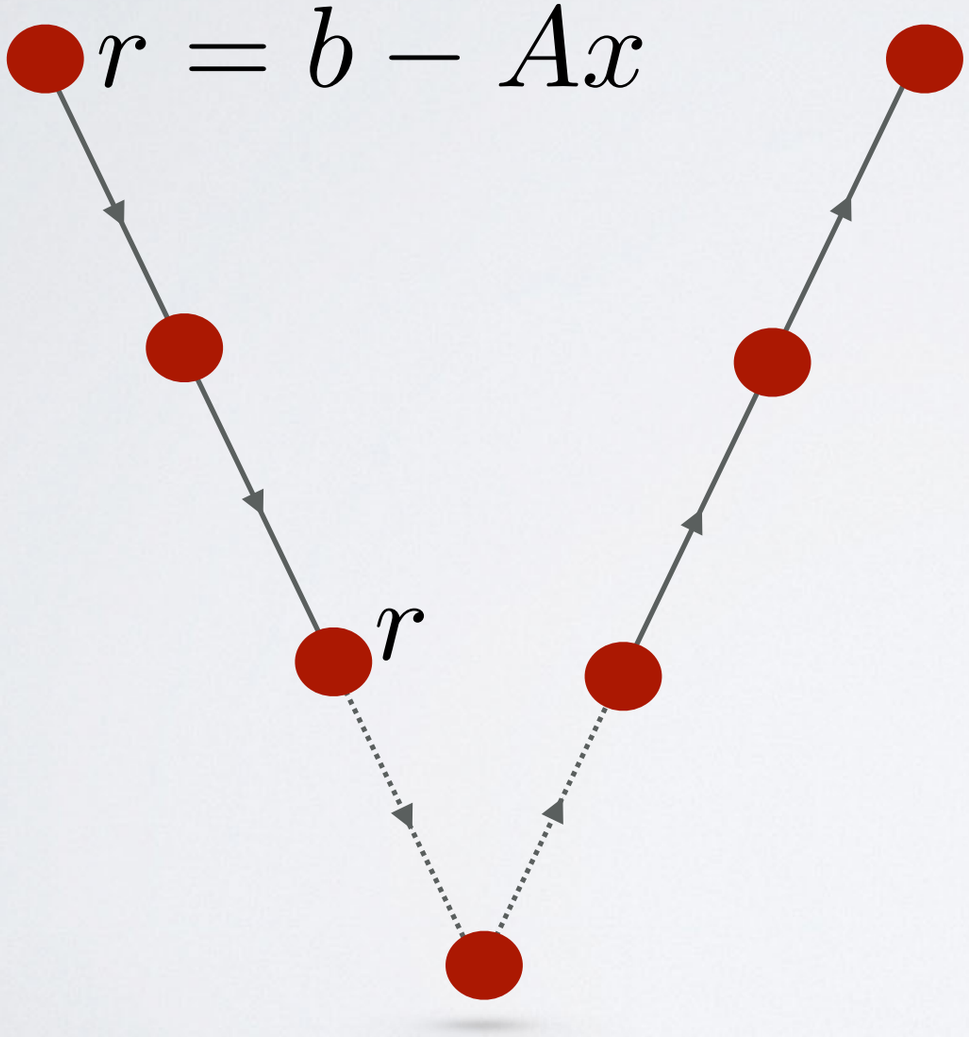
- Large amount of time spent in solving $Ax = b$
    1. opportunity for silent detection
    2. opportunity for less aggressive recovery
    3. refocus on convergence

- A good solver will attenuate some errors, recognize others
- Optimized components → more detection
- Natural hierarchy → micro-checkpointing

Trying a fault

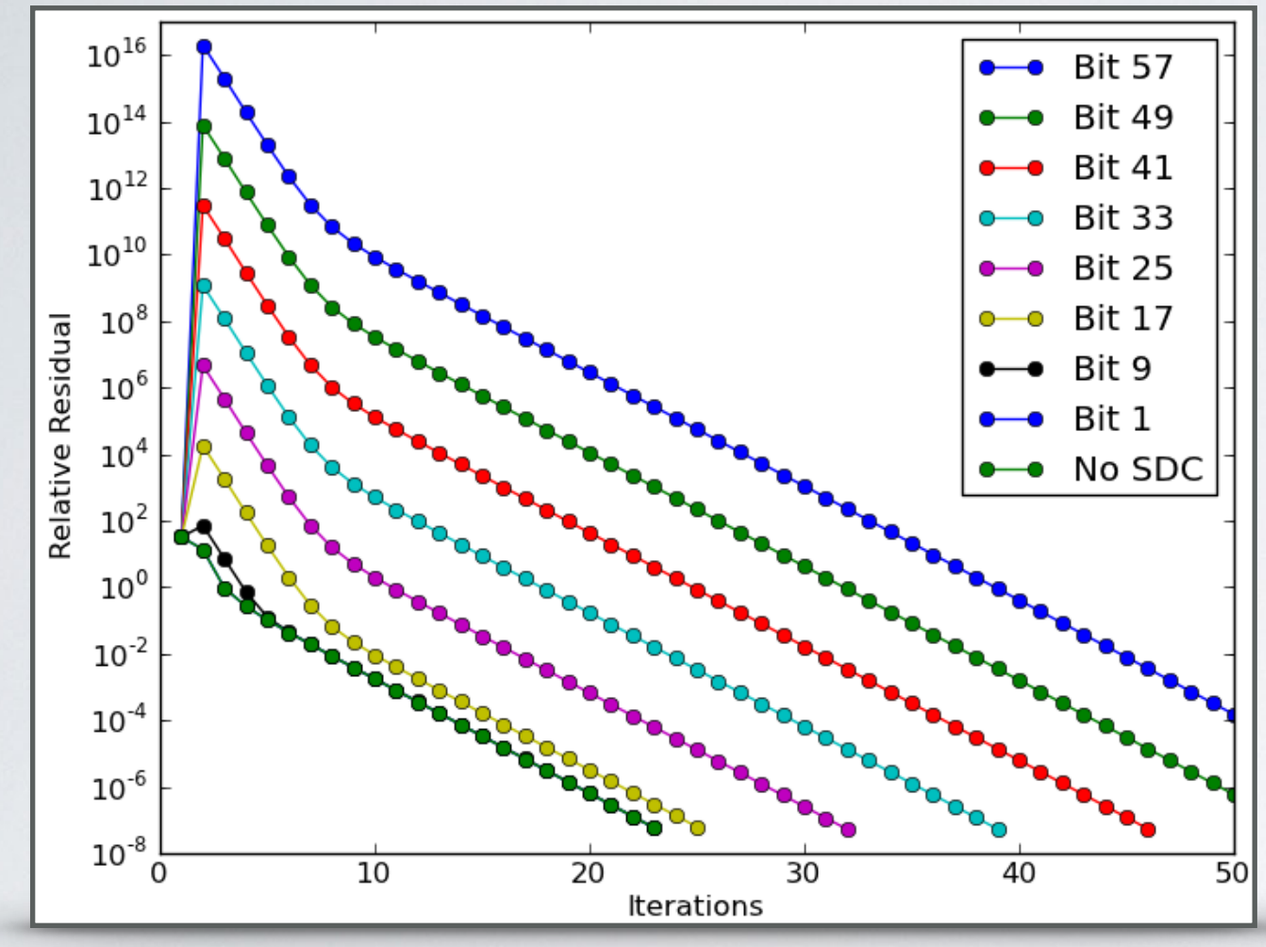
- Jon Calhoun's fault injection framework
  - LLVM IR bitcode
  - fault model to inject in arithmetic, pointers, control, etc

Trying a fault

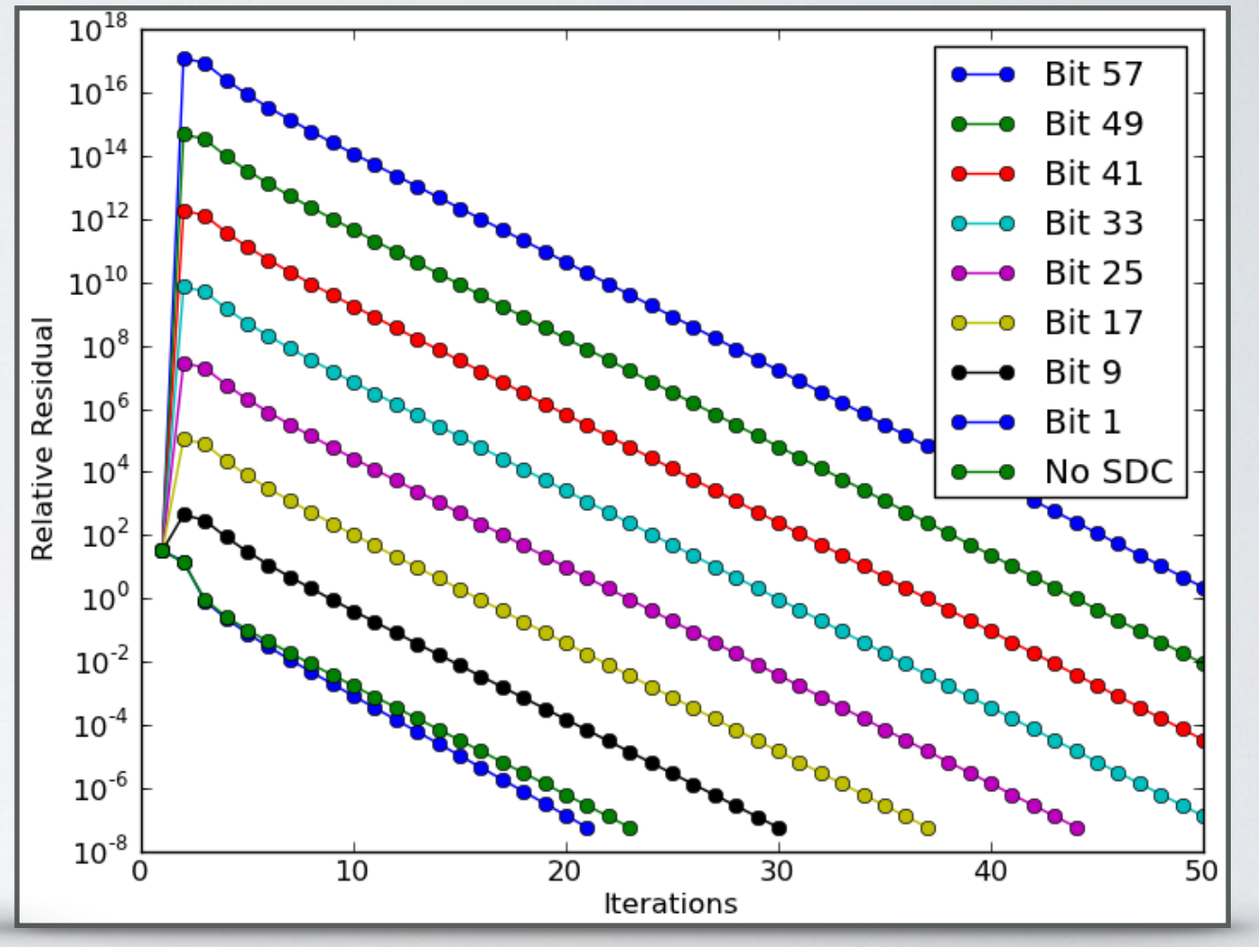


- Example: residual corruption in bit **m**
- 2D Poisson
  - results in an arithmetic error
  - other corruptions possible
    - pointer errors most common, but easily manifest as a SEG

Trying a fault



Finest level



4th level

- bit injected into residual in the second iteration

What happens

- Error propagates with the mat-vec Ax

- Coarse levels intensify fault (relative to amount of data)
  - less likely and cheaper to detect

- if a fault is detected **and it is severe**
  then recover, adjust

- Need accurate measure of **silent** errors
  - handle persistent errors
  - recover from transient ones

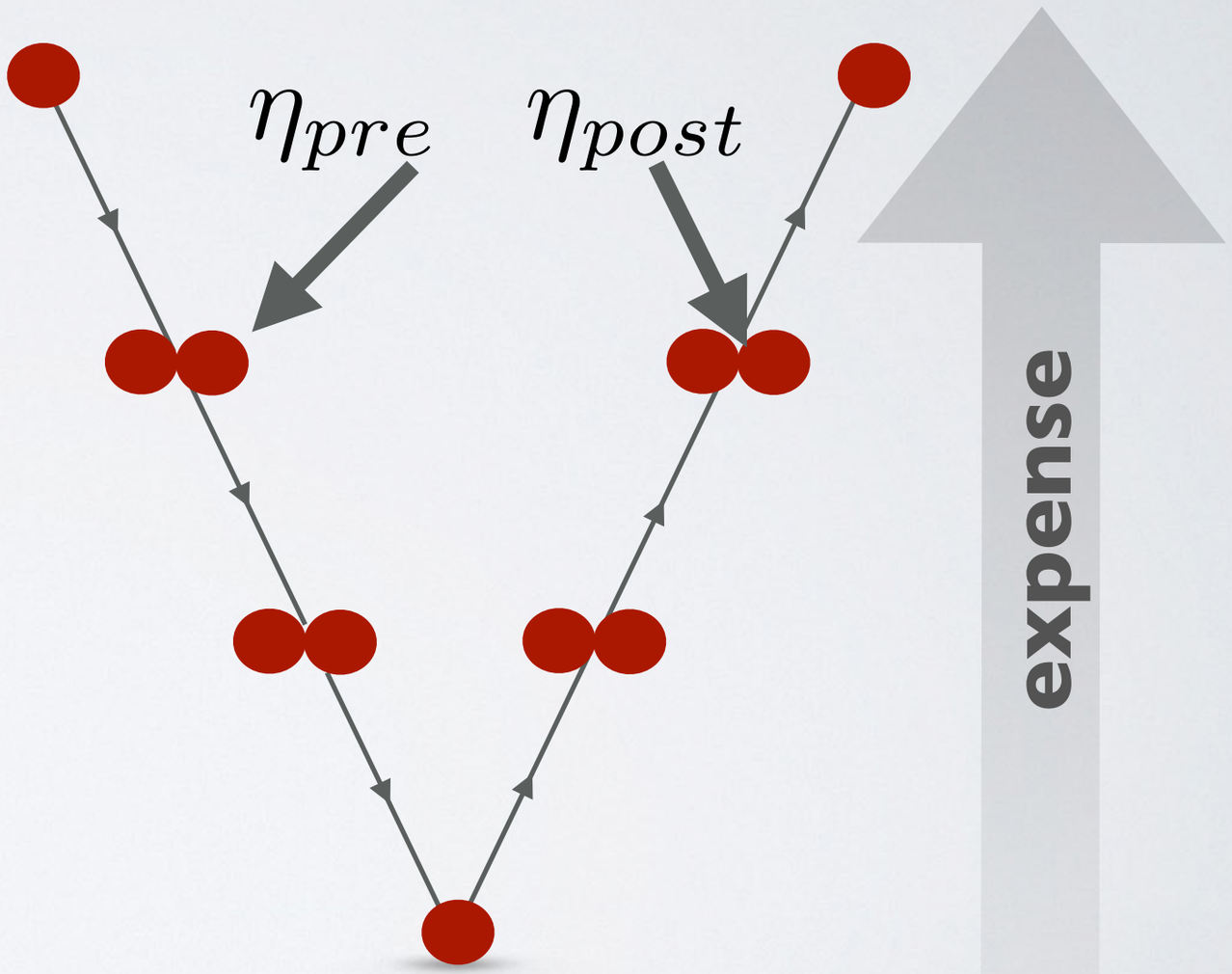
measures

- energy metric (expensive, accurate):

$$\eta = \langle Ax, x \rangle - 2\langle b, x \rangle$$

- residual metric (cheap,, bound):

$$\langle r, r \rangle$$



measures

Metric	Probability of encounter
Convergence	94%
segfault	24%
energy check	21%
residual check	3%
loop assertion	6%

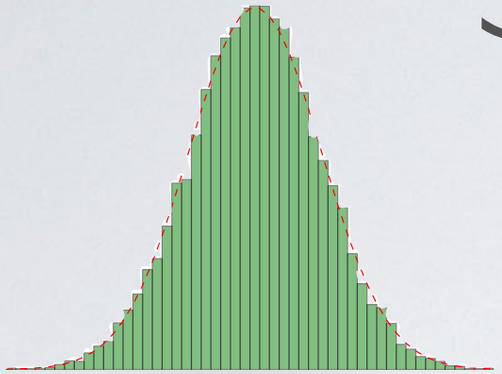
level	overhead
0	18%
1	6%
2	<1%
3	11%

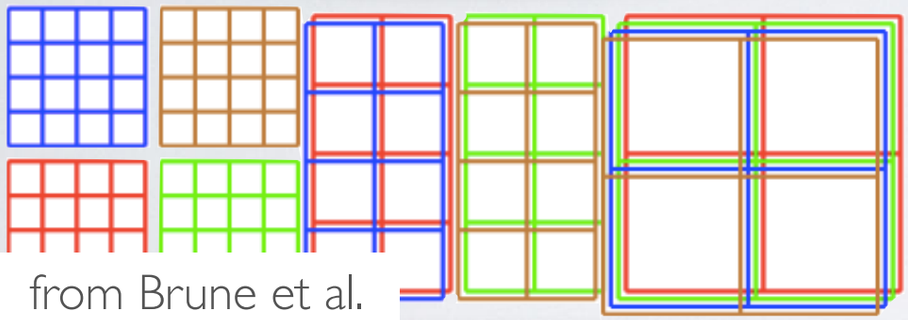
- 100 tests
- aniso diff. 40k x 40k
- injection rate ~ 1-2 per solve

measures

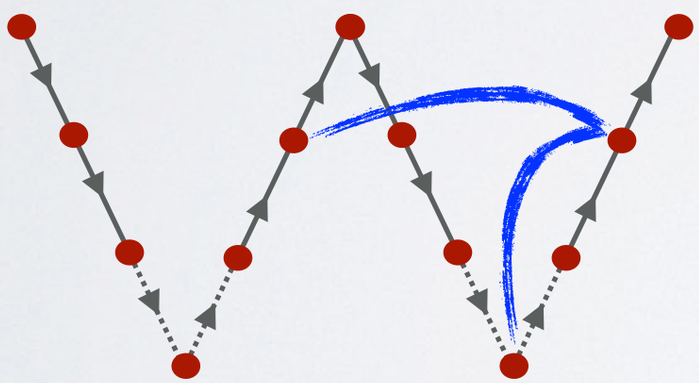
- How do we effectively measure stability locally?
- Do the measures capture the right silent errors (efficiently)?

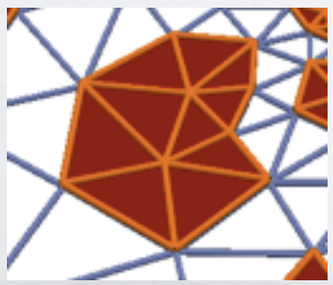
Some directions





from Brune et al.





- Fault model to match data
  - potential collaboration: Cappello (ANL/INRIA)
- Redundancy and recovery
  - potential collaboration: Brown et al (ANL)
- Algorithm recovery with micro checkpointing
  - potential collaboration: Cappello (ANL/INRIA)
- Extend interpolation and non-Galerkin methodology to DD-type
  - potential collaboration: Brown (ANL), Nataf (LJLL), Grigori (INRIA)

- this is work with
  - Jon Calhoun (fault injection, analysis)
  - Amanda Bienz (non-Galerkin scaling)
  - Andrew Reisner (redundancy models)
  - others @ Illinois, LLNL, ANL