Making Static Pivoting Dependable

E. Jason Riedy
EECS Department, UC Berkeley
ejr@cs.berkeley.edu

1. Motivation

For sparse LU factorization, dynamic pivoting tightly couples symbolic and numerical computation. Dynamic structural changes limit parallel scalability. Demmel and Li use *static* pivoting in distributed SuperLU for performance, but intentionally perturbing the input may lead silently to erroneous results.

Are there experimentally stable static pivoting heuristics that lead to a dependable direct solver? The answer is currently a qualified *yes*. Current heuristics fail on a few systems, but all failures are detectable.

2. Pivoting and Perturbation Heuristics

- Partial pivoting is a heuristic to limit element growth.
- Another approach: perturb pivots to keep divisors large.
- Small perturbations should not overly affect stability.

Pivoting Heuristics

Partial Pivoting

- Given: ratio $u >= 1$.
- $i_p = \mathrm{argmax}_{i=j+1..N} |a(i,j)|$.
- If $|a(i_p,j)| > u \cdot |a(j,j)|$, swap rows i_p and j.

Static Pivoting

- Given: threshold T, perturbation P.
- If $|a(j,j)| < T$, add $\pm P$ to $a(j,j)$ (preserving sign).

Perturbation Heuristics

- Norm-relative: Keep perturbations small relative to global norm. Used in SuperLU with parameter $\rho = 2^{10}\sqrt{\text{machine precision}} \approx 2^{-17}$.
- Diagonal-relative: Sparsity-preserving orderings limit changes to the diagonal, so keep perturbations small relative to original diagonal.

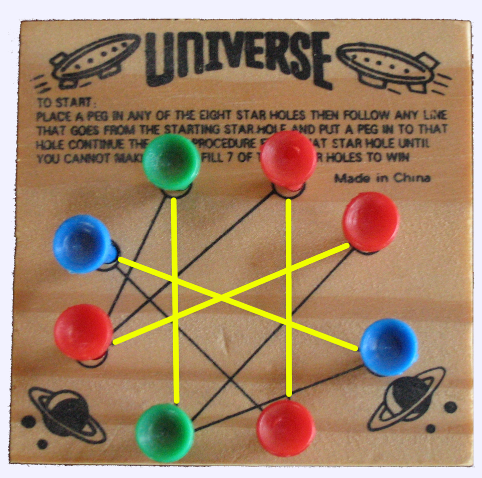
Norm-Relative

- $T = P = \rho \cdot \|A\|_1$

Diagonal-Relative

- Save original diagonal entries in d.
- At column j, $T = P = \rho \cdot |d(j)|$

3. Weighted Bipartite Matchings



- To minimize the number of perturbations during factorization, we pre-pivot to place large entries on the diagonal.
- Parallelizable auction algorithms can maximize product or sum of the diagonal; sums produce less reliable pivots.
- Quantizing diagonal to integer saves around 10% of execution time with no noticable effect on pivots.
- Approximating the objective improving time by 30% – 300%, also with little noticable effect.

4. Iterative Refinement

- Newton's method applied to $Ax = b$.
- Can use extra precision in residual and temporary solution to obtain normwise and componentwise accuracy.
- See Yozo Hida's poster for full details.

Negligible Incremental Cost: Extra-precise residuals approximately $25nnz$ fp operations in software, but no additional memory accesses. If the factoring produces a fill of over $12\times$, then extra-precise residuals are no more expensive than each step's solve.

5. Summary

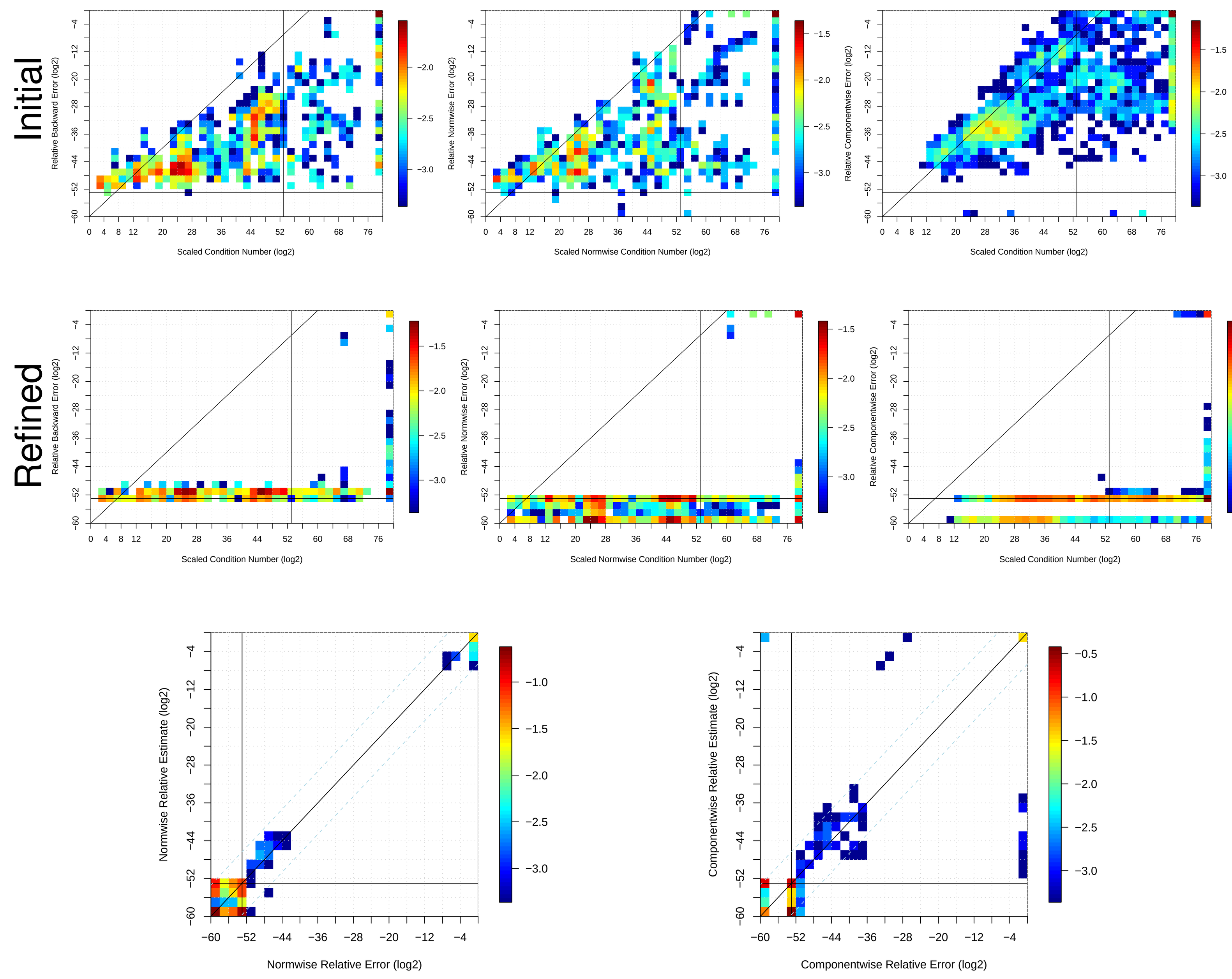
Out of 231 large matrices from UF Collection (N from 2500 to 213 360):

- Extra-precise refinement failed to stabilize five matrices for the best normrelative perturbations and two for diagonal relative (Zhao2, av41092).
- All failed from pivot growth $> 10^8$; simple to check.
- These systems were the only error estimate failures.
- Note: Only worried about condition numbers less than $\approx {}^1/_{\text{machine precision}}$, depending on non-zeros per row / col.
- Little variation

Failures are rare and detectable. Careful scaling should reduce pivot growth in current failure cases.

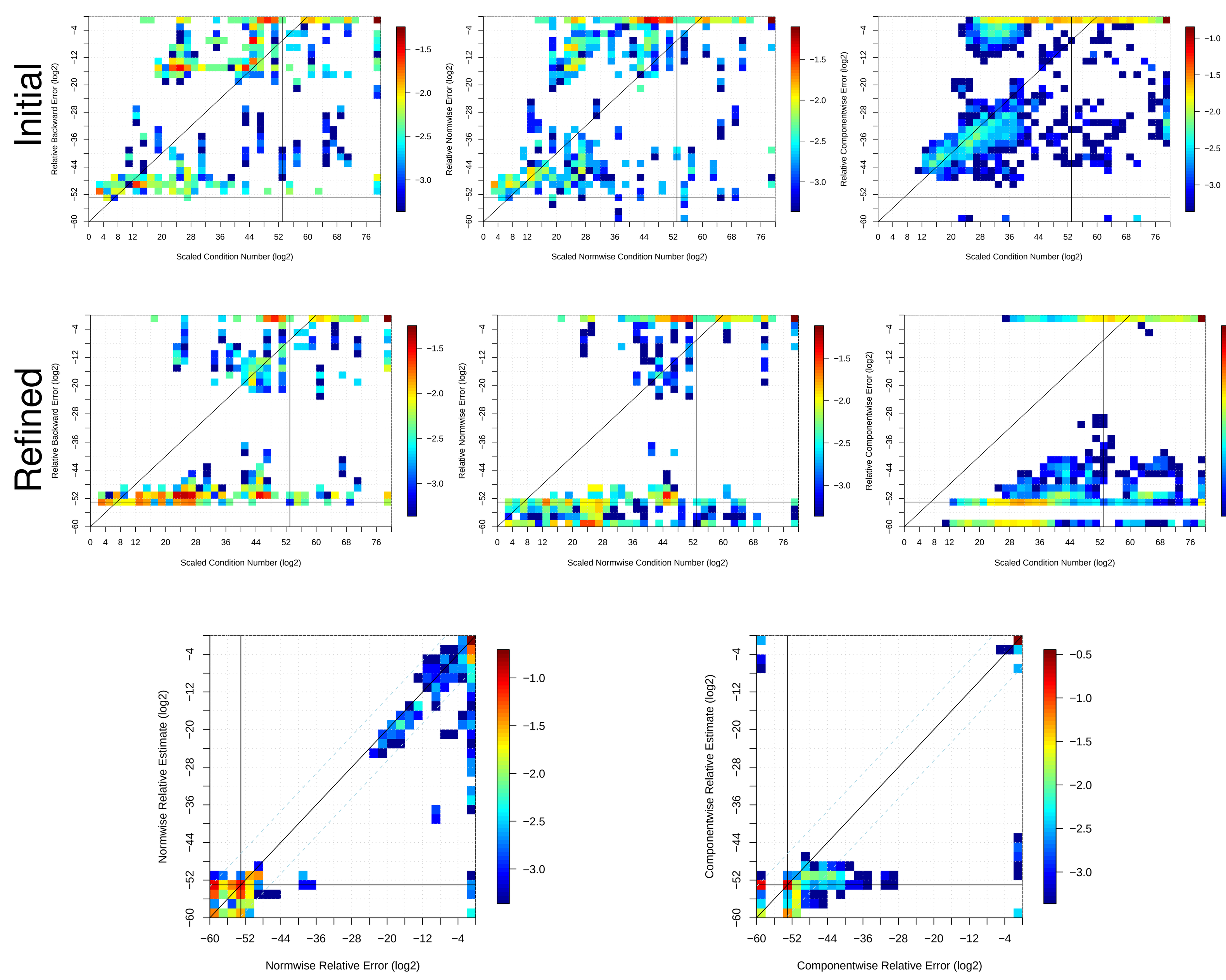
Partial Pivoting

$\text{berr} = \max \frac{|b-Ax|}{|b|+|A|\cdot|x|}$ $\quad \text{err}_{\text{nrm}} = \frac{\|x-x_{\text{true}}\|_\infty}{\|x_{\text{true}}\|_\infty}$ $\quad \text{err}_{\text{cmp}} = \max \frac{|x-x_{\text{true}}|}{|x_{\text{true}}|}$



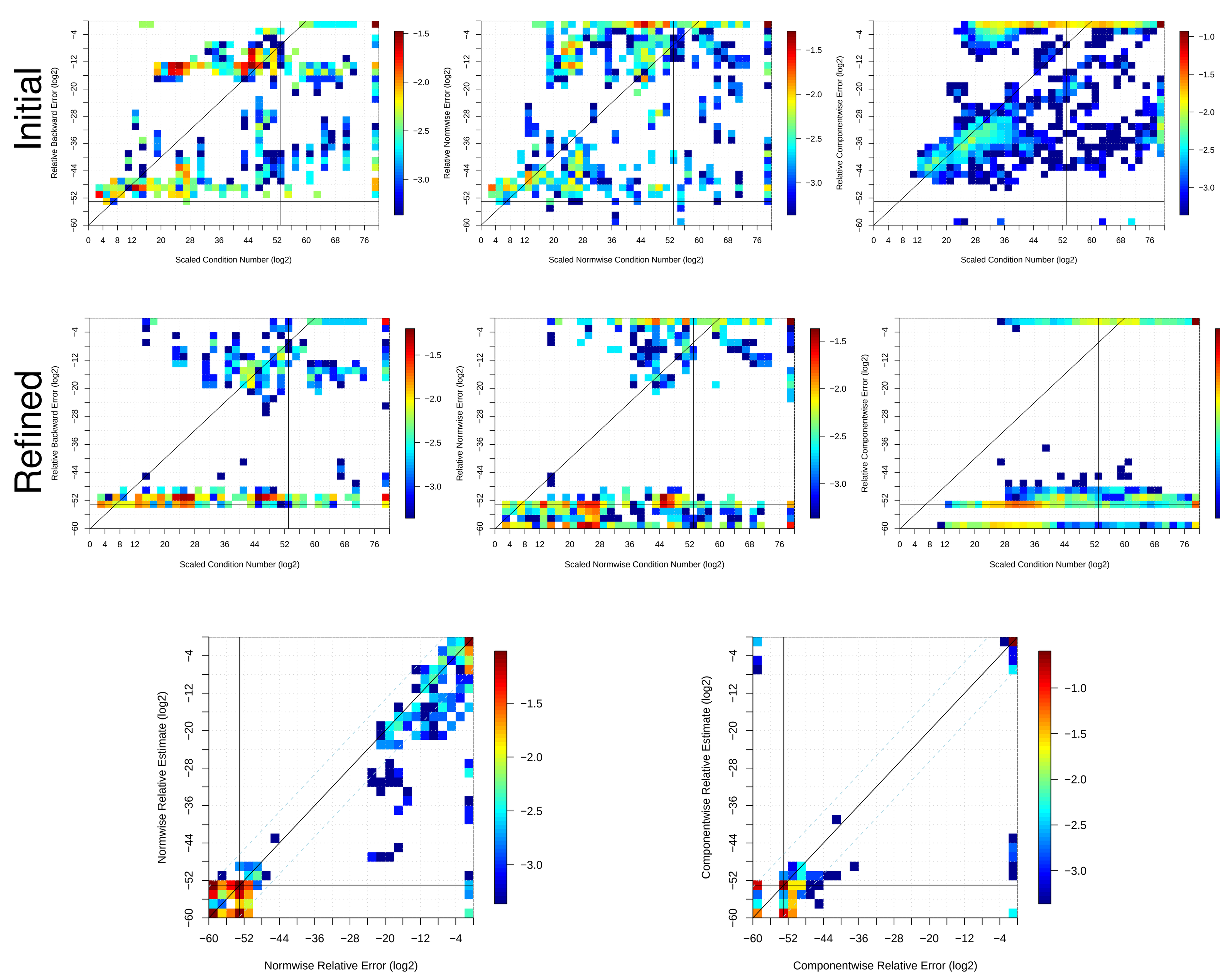
Norm-Relative Perturbations, $u = 2^{-17}$

$\text{berr} = \max \frac{|b-Ax|}{|b|+|A|\cdot|x|}$ $\quad \text{err}_{\text{nrm}} = \frac{\|x-x_{\text{true}}\|_\infty}{\|x_{\text{true}}\|_\infty}$ $\quad \text{err}_{\text{cmp}} = \max \frac{|x-x_{\text{true}}|}{|x_{\text{true}}|}$



Diagonal-Relative Perturbations, $u = 2^{-12}$

$\text{berr} = \max \frac{|b-Ax|}{|b|+|A|\cdot|x|}$ $\quad \text{err}_{\text{nrm}} = \frac{\|x-x_{\text{true}}\|_\infty}{\|x_{\text{true}}\|_\infty}$ $\quad \text{err}_{\text{cmp}} = \max \frac{|x-x_{\text{true}}|}{|x_{\text{true}}|}$



Colorbars: $\log_{10}$ of relative population.

Bay Area Scientific Computing Day 2006, 4 March 2006, Livermore, CA