Adaptive Precision Solvers for Sparse Linear Systems

THE UNIVERSITY of TENNESSEE KNOXVILLE

HARTWIG ANZT
JACK DONGARRA

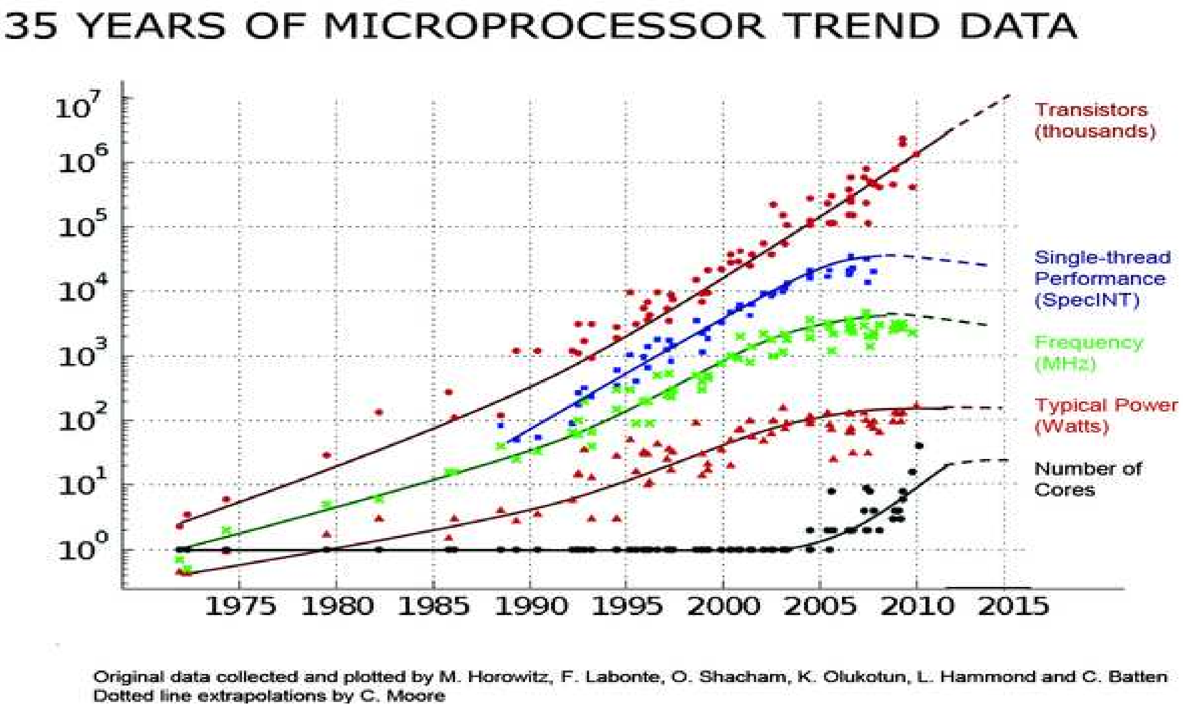
UNIVERSITAT JAUME I

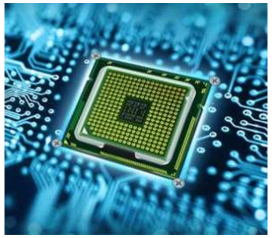
ENRIQUE S. QUINTANA-ORTÍ

SC15 Austin, TX | hpc transforms.

MOTIVATION

- Dennard's scaling vs Moore's Law





IBM Power8 (Q3'15)
- 22 nm
- 3.12 GHz
- TDP 190-200 W
- 12 cores/96 threads

Intel Xeon E5-4669 v3 (Q2'15)
- 22 nm
- 2.1 GHz
- TDP 135 W
- 18 cores/36 threads

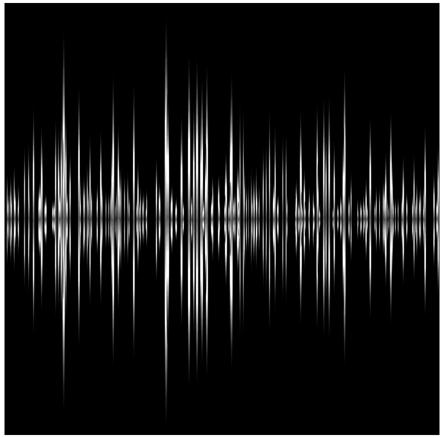
MOTIVATION

- 5 nm in about 7-9 years:
  - 2.4x faster
  - 10x more transistors
  - … but only 10% simultaneously active

Dark silicon (utilization wall) and specialization!

MOTIVATION

- *Approximate Computing*: energy vs accuracy (or reliability)

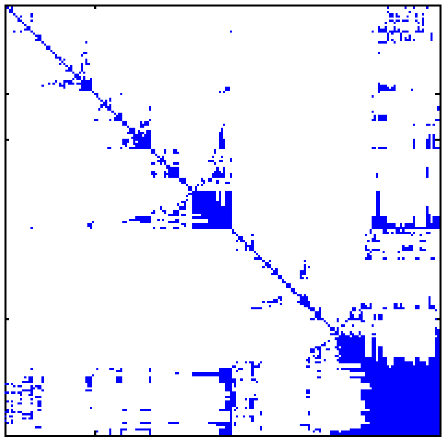


Signal & video processing

Probabilistic inference
Service profiling
Monte Carlo simulation
Machine learning

MOTIVATION

- *Approximate Computing*: energy vs accuracy (or reliability)



Numerical Linear Algebra?

- Tiny errors can rapidly aggregate
- Double precision is the standard

OUTLINE

- Jacobi solver for sparse linear systems
- Mantissa-adaptive Jacobi
- Experimental evaluation
- Concluding remarks

Stationary Methods: Jacobi

- Given $Ax = b$

$$
\begin{aligned}
x^{\{k\}} &:= D^{-1}\left(b-(A-D)x^{\{k-1\}}\right) \\
&= D^{-1}b+Mx^{\{k-1\}}, \qquad k=1,2,\ldots,
\end{aligned}
$$

with $D = diag(A)$, and $x^{\{0\}}$ a starting solution guess

- Linear convergence provided spectral radius of $M < 1$
- Components of $x^{\{k\}}$ can be computed in parallel
- Alternative to exact triangular solves in approximate ILU preconditioning

Mantissa-adaptive Jacobi

- Basic idea:
  - Operate in low (“cheap”) precision and gradually increase as needed
    - Currently {32,64}-bit precision
    - NVIDIA “Pascal” GPUs: {16,32,64}-bit precision?
    - FPGAs: custom

Mantissa-adaptive Jacobi

- Basic idea:
  - Operate in low ("cheap") precision and gradually increase as needed
  - Already explored for Jacobi

> "On the potential of significance-driven execution for energy-aware HPC"
> P. Gschwandtner, C. Chalios, D. S. Nikolopoulos, H. Vandierendonck, T. Fahringer
> Computer Science – Research and Development, 2015

  - …but do it with a fine granularity (component-wise)
  - …and apply a cheap criterion to detect when to increase precision

Mantissa-adaptive Jacobi

- How?
  - Component-wise contraction property:

$$\exists\, 0 < \theta_i < 1 \ :$$
$$\left| x_i^{\{k\}} - x_i^{\{k-1\}} \right| \leq \theta_i \left| x_i^{\{k-1\}} - x_i^{\{k-2\}} \right| \leq \theta_i^2 \left| x_i^{\{k-2\}} - x_i^{\{k-3\}} \right| \ldots$$

Mantissa-adaptive Jacobi

- How?
  - Component-wise contraction rate is constant:

$$c_i^{\{k\}} := \frac{z_i^{\{k-1\}}}{z_i^{\{k\}}} = \frac{\left|x_i^{\{k-1\}} - x_i^{\{k-2\}}\right|}{\left|x_i^{\{k\}} - x_i^{\{k-1\}}\right|}, \quad k \geq 2, \qquad c_i^{\{2\}} = c_i^{\{3\}} = c_4^{\{i\}} = \ldots = c_i$$

Exploding ratio $z^{\{k-1\}}/z^{\{k\}}$ due to small $z^{\{k\}}$ indicates convergence of the component in the current precision

Mantissa-adaptive Jacobi

- Practicalities:
  - Reduce the cost/periodicity of the test: $z^{\{k-\Phi\}}/z^{\{k\}}$
  - Take into account rounding errors:

$$\left| \frac{z_i^{\{k-\phi\}}}{z_i^{\{k\}}} - c_i^{\phi} \right| > \tilde{\delta}$$

determines that an extension is necessary

  - Avoid stagnation by setting

$$\tilde{\delta} := \delta \cdot (c_i^{\phi} - 1)$$

for some user-defined $0 < \delta < 1$

Mantissa-adaptive Jacobi

- Practicalities:
  - δ governs how quickly the mantissa is extended:
    - Faster/slower as $\delta \rightarrow 0/1$
  - Control the magnitude (gradient) of the increase: γ bits
  - First three iterations in full precision, to estimate component-wise contraction rate

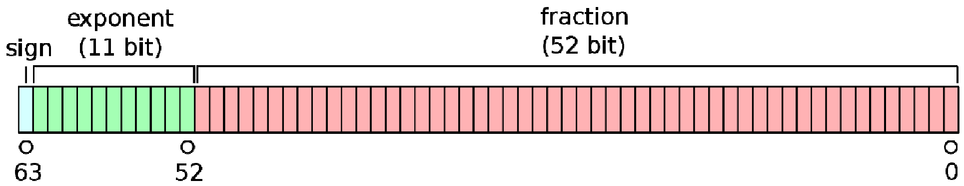
Mantissa-adaptive Jacobi

- Connection between fault tolerance and AC
  - Fault tolerance: obtain “exact” solution in presence of errors
  - AC: operating with low precision can be viewed as errors in the part of the mantissa that is chopped
  - Deviation from the contraction property: errors (fault tolerance) or convergence in current precisión (AC)

“Tuning iterative solvers for fault resilience”
H. Anzt, J. Dongarra, E. S. Quintana-Ortí
ScalA’2015 (Tomorrow!)

Experiments

- Matlab (R2014a) and IEEE 754 double precision (64 bits):



- Solvers:
  - Full precision (52-bit mantissa)
  - 8-bit precision (8-bit mantissa)
  - Adaptive precision, starting with 8 bits

- Tune δ (tolerance threshold), γ (bit extension gradient), Φ (periodicity)

Experiments

- 27-pt stencil discretization of 3D Laplace using *16x16x16* mesh:
  - Symmetric matrix of order 4,096 with 97K nonzeros, well conditioned
  - Starting guess $x^{\{0\}} = 0$, $b = [1,1,\ldots,1]$
  - Convergence stopping criterion

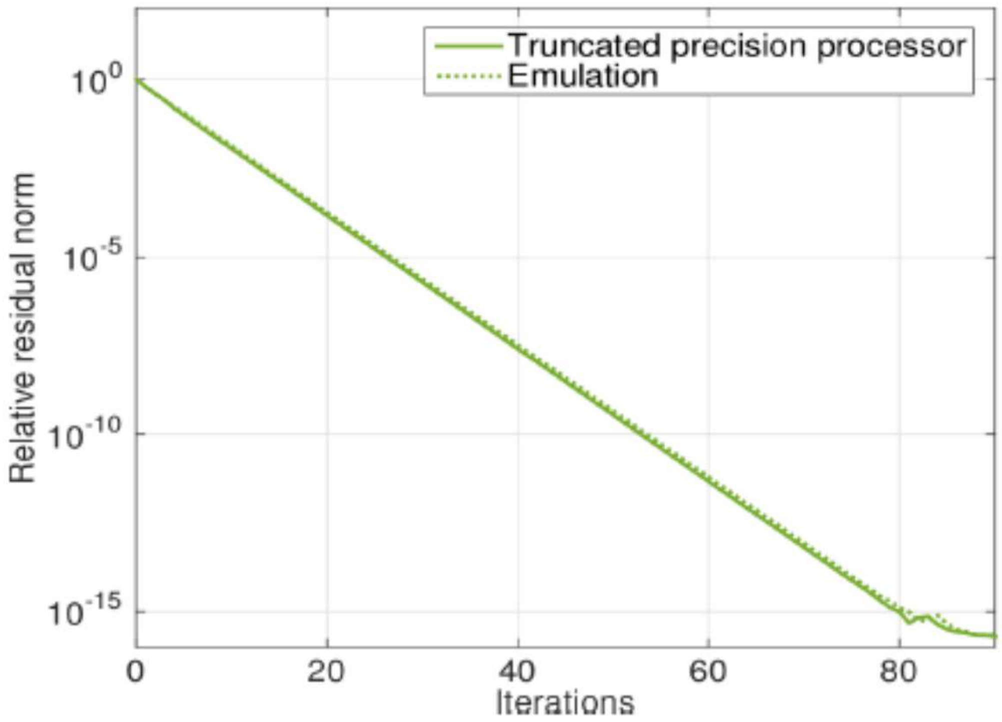
$$\mathcal{R}\left(x^{\{k\}}\right) = \frac{\|b - Ax^{\{k\}}\|_2}{\|b - Ax^{\{0\}}\|_2}$$

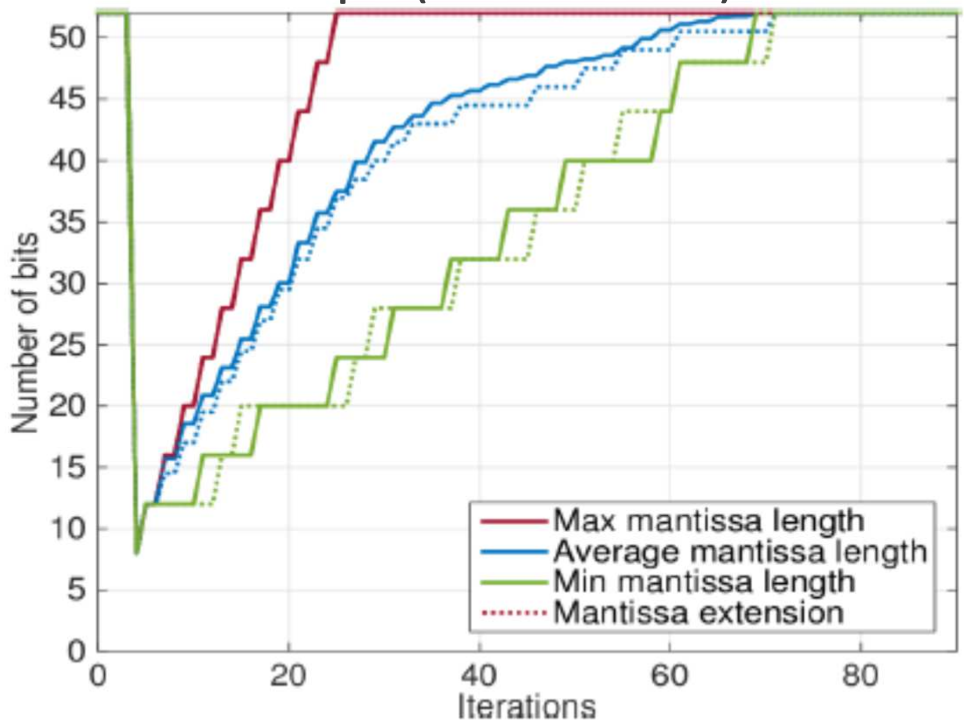
Experiments

- Intel Xeon E5-2670 CPU. Emulate adaptive precision in hardware. At each iteration:
  - Truncate input data to desired precision
  - Compute (in 64-bit arithmetic) $y_i := A(i,:) \cdot x^{\{k-1\}}$ and truncate (to desired precision)
  - Compute $p_i := (b_i - y_i) / A(i,i)$ and truncate
  - Compute next iterate $x^{\{k\}} := x^{\{k-1\}} + p_i$ and truncate

Experiments: validate emulation mode

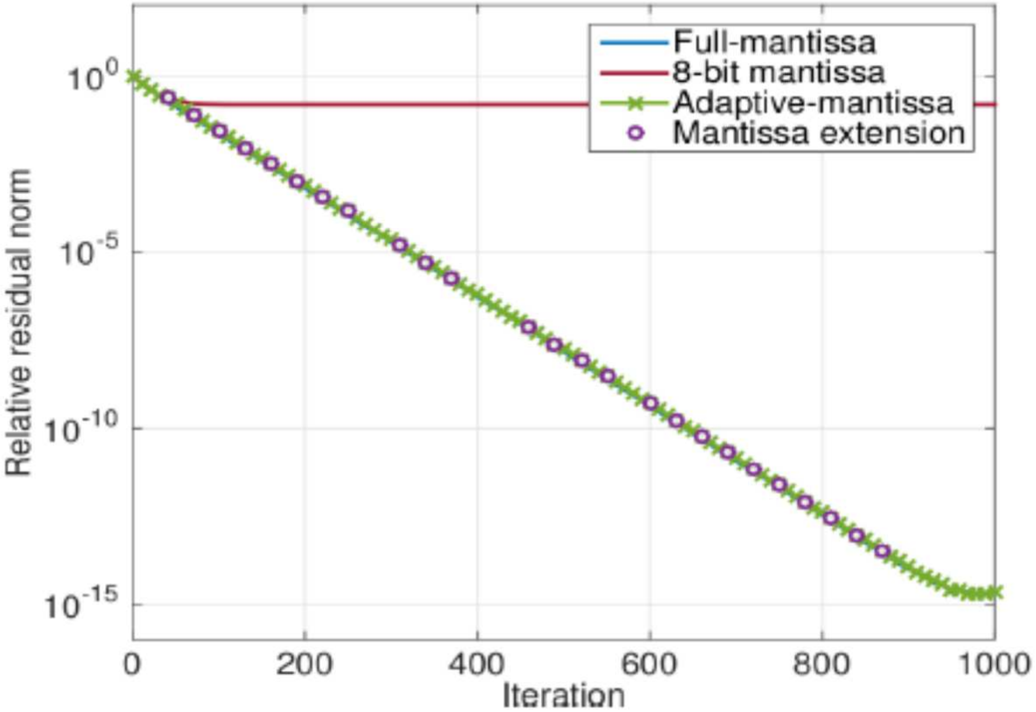
- Truncate result of dot-product vs truncate all flops (*4x4x4* case)

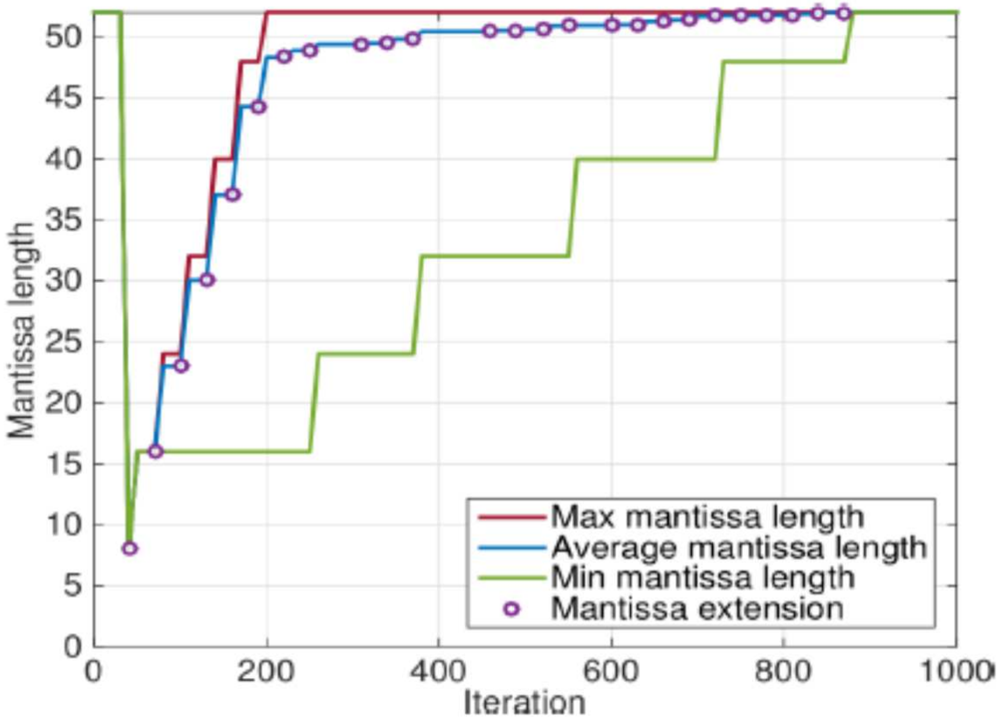




Experiments: select δ

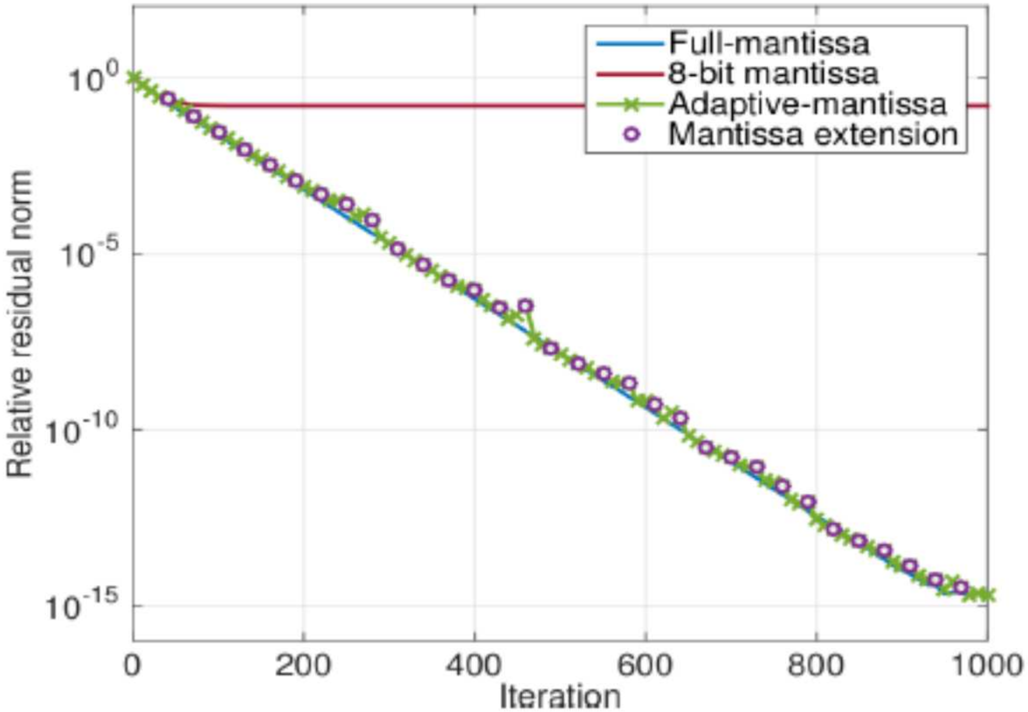
- $\delta = 0.1$, $\gamma = 8$, $\Phi = 10$

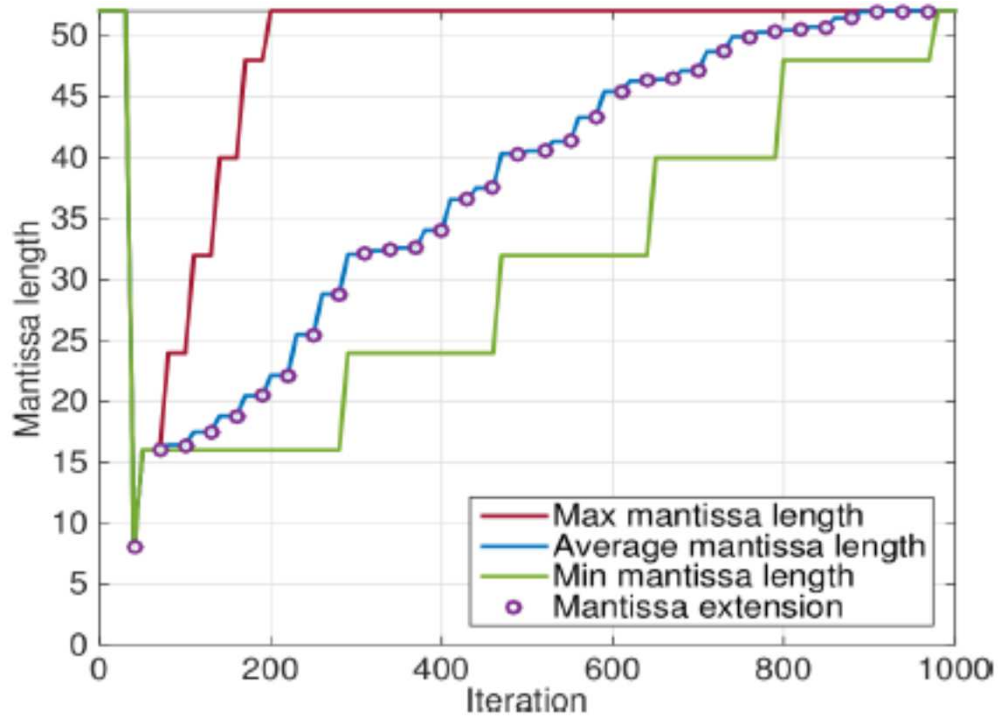




Experiments: select δ

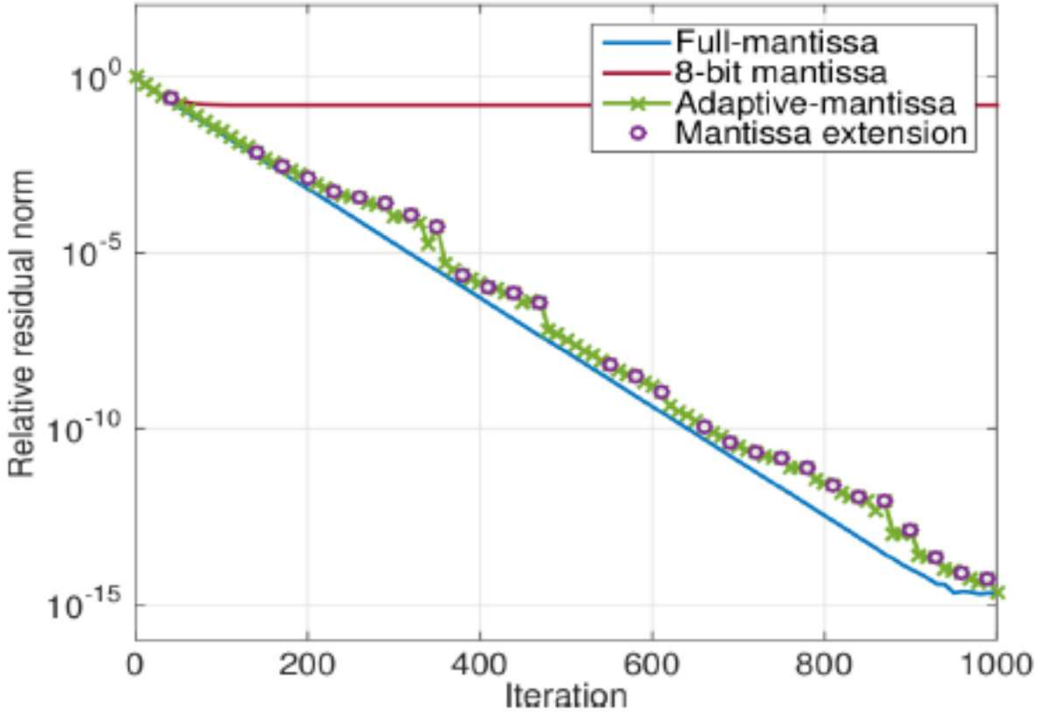
- $\delta = 0.5$, $\gamma = 8$, $\Phi = 10$

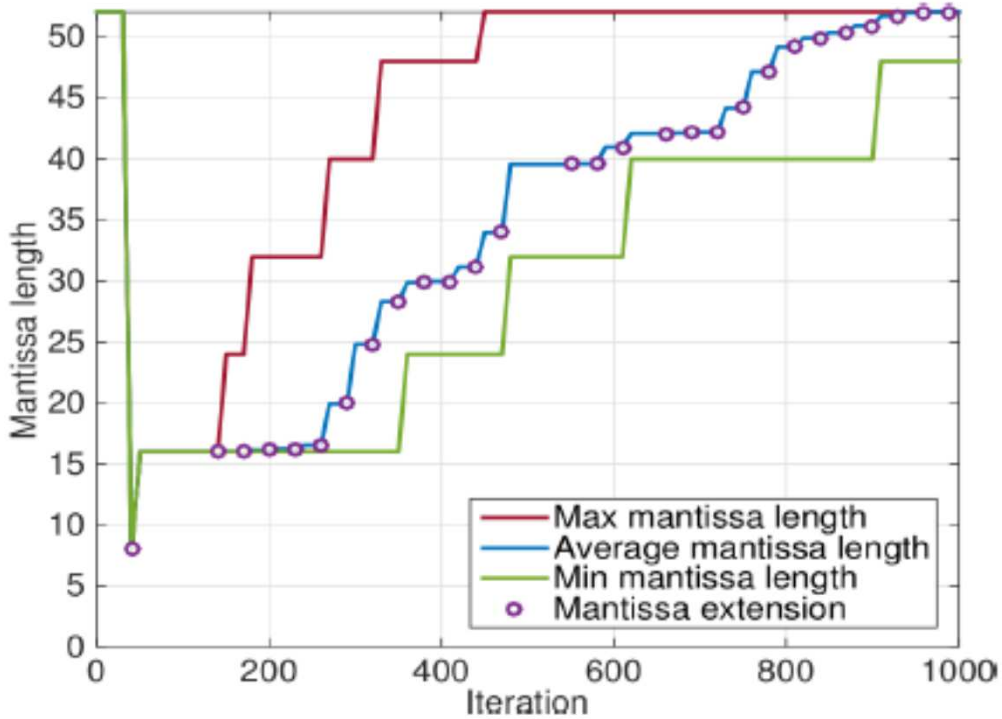




Experiments: select δ

- $\delta = 0.9$, $\gamma = 8$, $\Phi = 10$





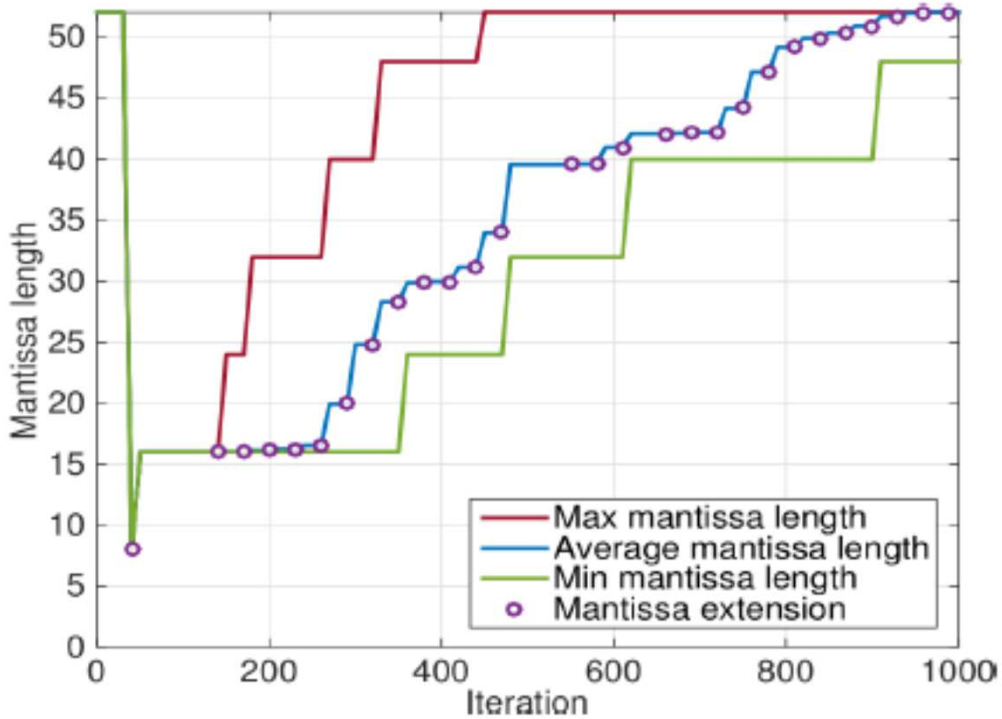
Experiments: select δ

- Quantify computational cost/savings
  - For iteration k

$$\mathscr{C}^{\{k\}} := \sum_{i=1}^{n} nnz_i(A) \cdot \mathscr{L}_i^{\{k\}}$$

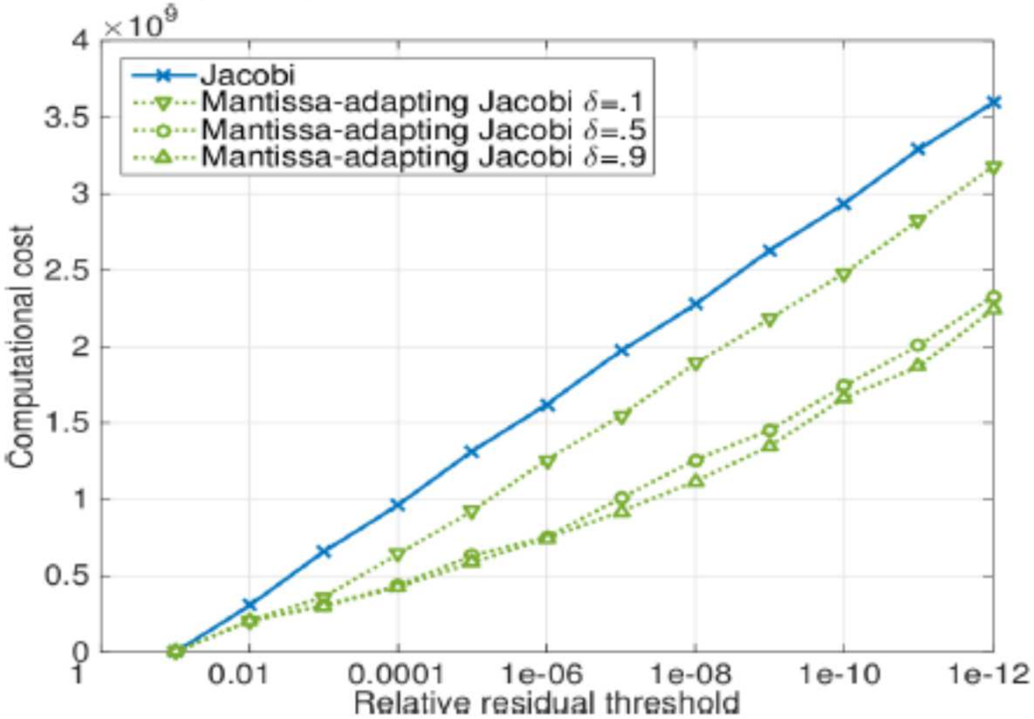
  - For the full solve

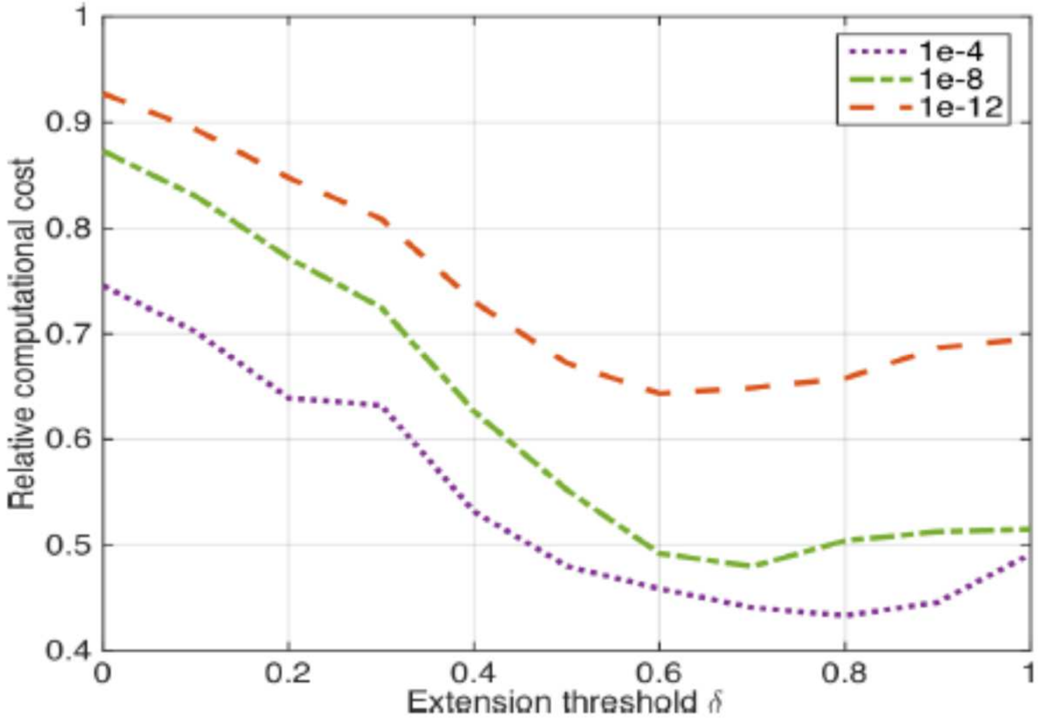
$$\mathscr{C}^{[1,\tilde{k}]} := \sum_{k=1}^{\tilde{k}} \mathscr{C}^{\{k\}} = \sum_{k=1}^{\tilde{k}} \left(\sum_{i=1}^{n} nnz_i \cdot \mathscr{L}_i^{\{k\}} \right)$$



Experiments: select δ

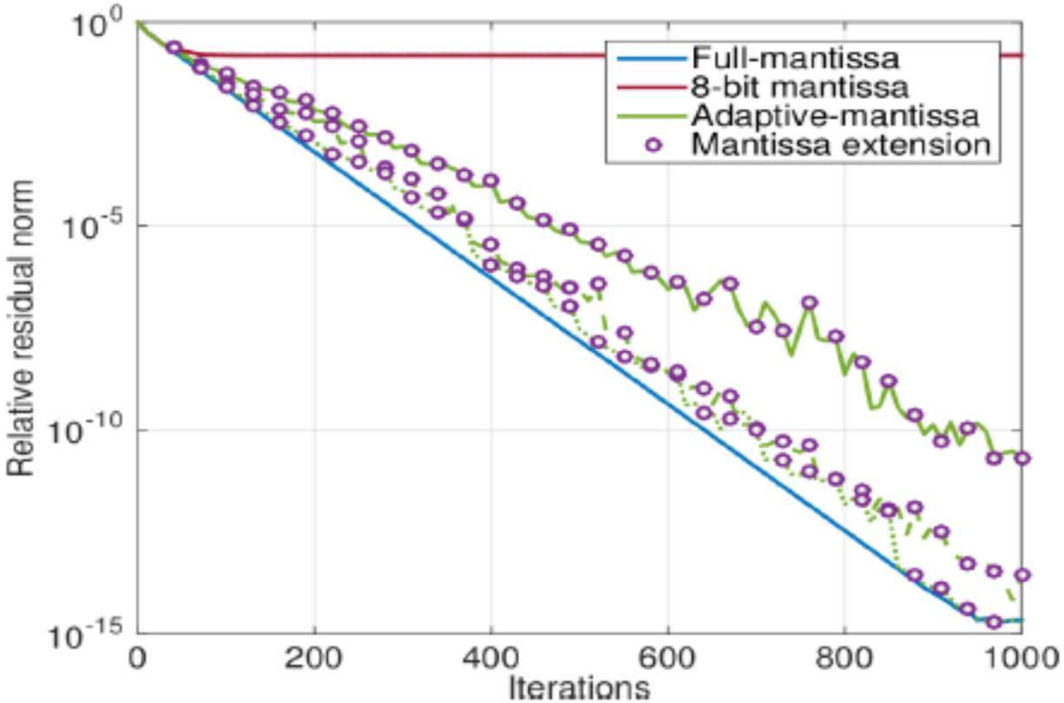
- $\gamma = 8$, $\Phi = 10$

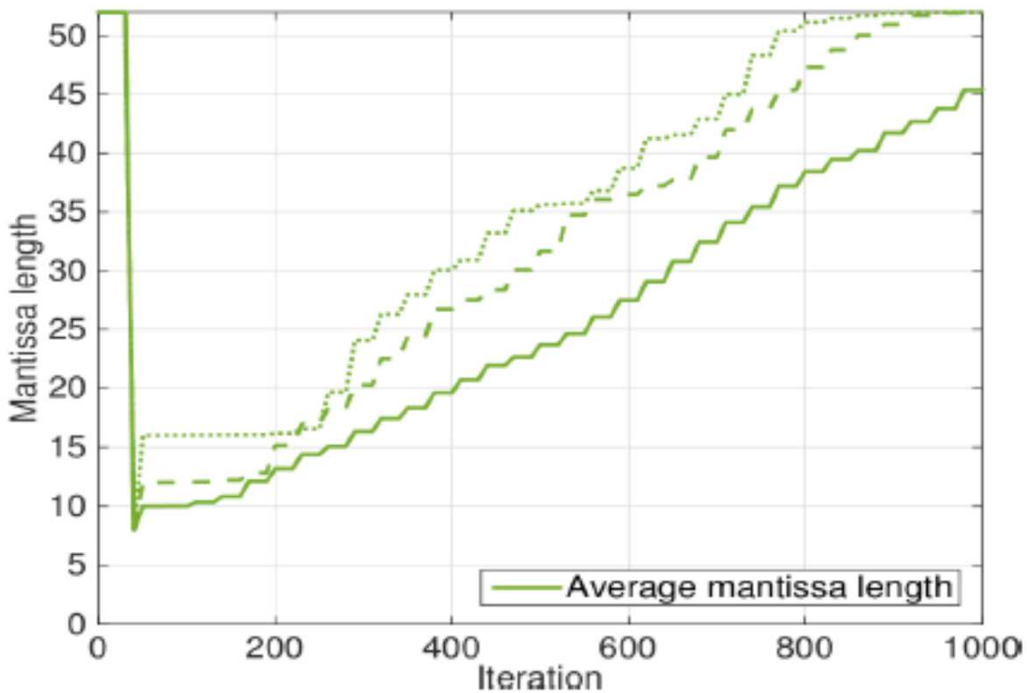




Experiments: select γ

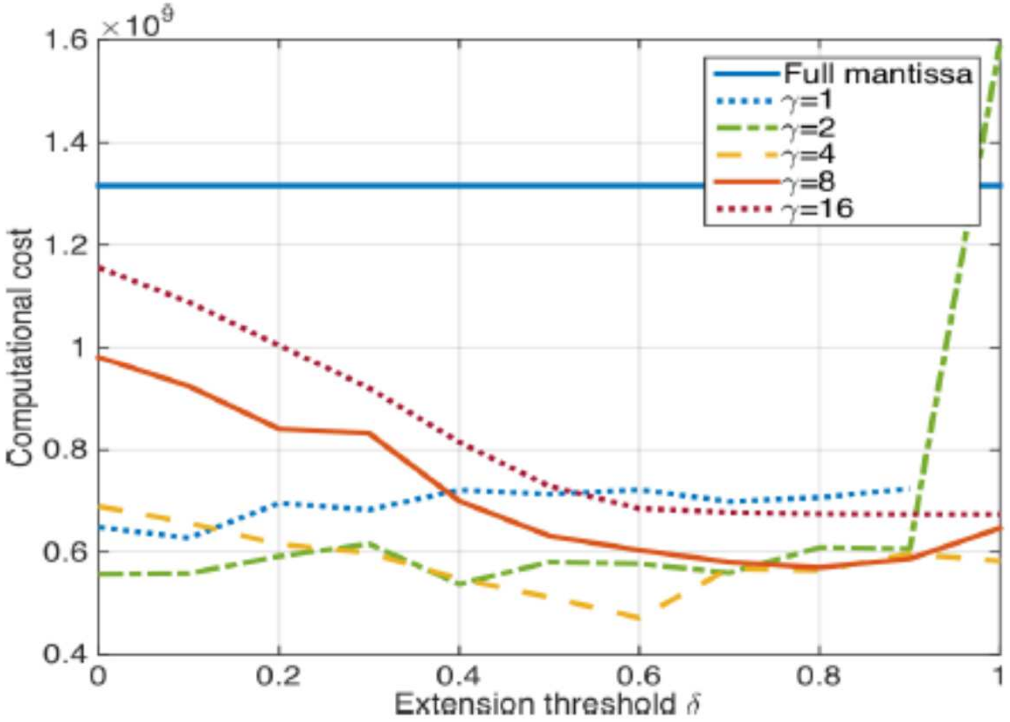
- $\gamma = 2, 4$ or 8, $\Phi = 10$, $\delta = 0.8$

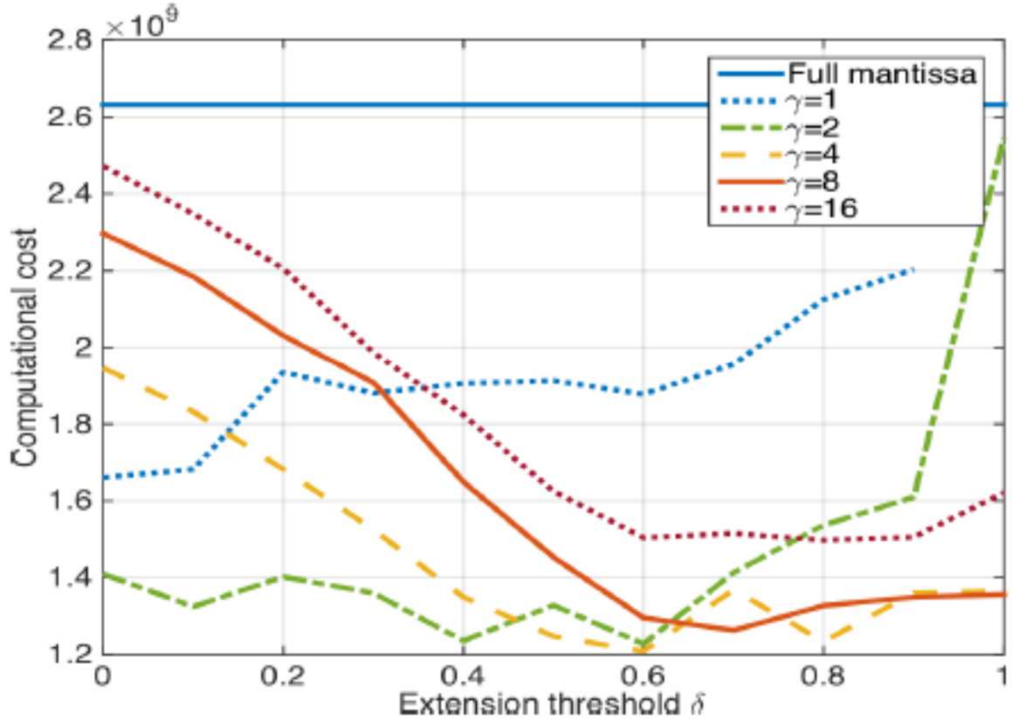




Experiments: select γ

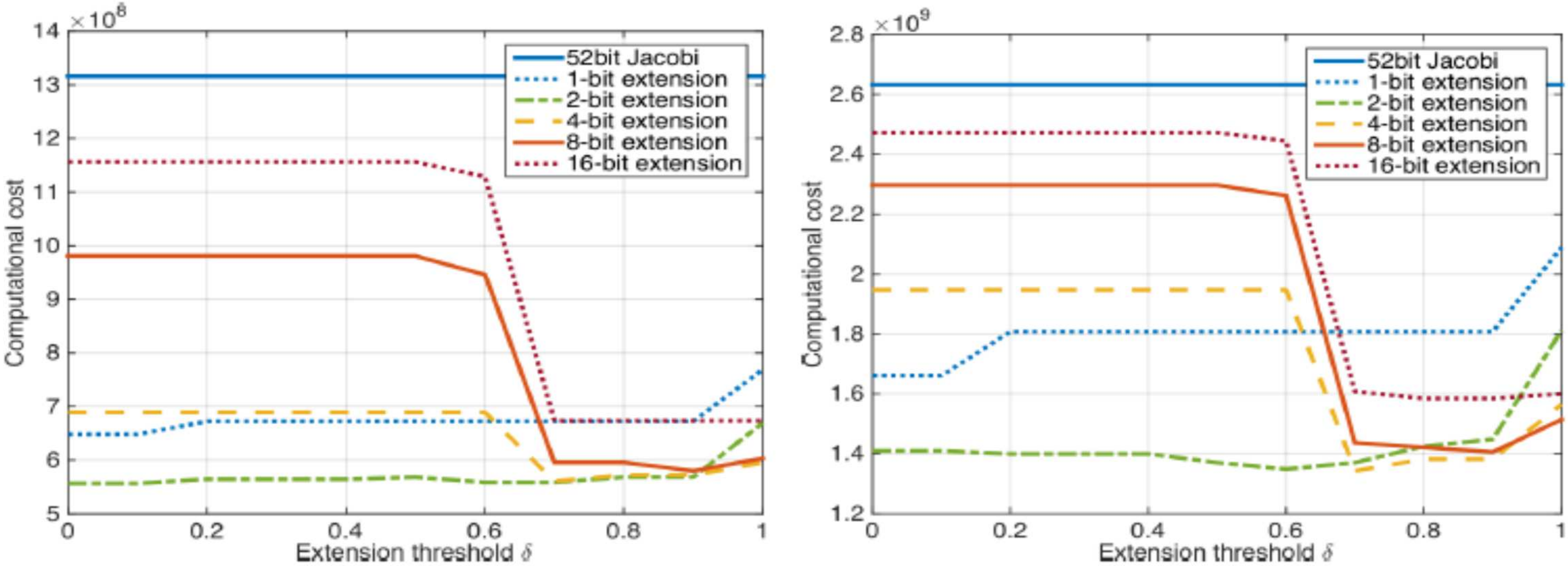
- $\gamma = 1, 2, 4, 8$ or 16, $\Phi = 10$





Experiments: vector-wise adaptive

- $\gamma = 1, 2, 4, 8$ or $16, \Phi = 10$



Experiments: Approx. triangular solves in ILU preconditioner

		Optimal configuration			Number of iterations Adaptive-mantissa Jacobi/full-mantissa Jacobi					Computational cost of adaptive-mantissa w.r.t. full-mantissa Jacobi				
		ϕ	γ	δ	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}
CHP	L	1	16	0.5	13/11	23/20	31/29	40/38	48/46	0.69	0.82	0.82	0.85	0.88
	U	1	16	0.4	12/11	22/20	29/28	37/37	45/46	0.60	0.76	0.79	0.81	0.82
DC	L	1	32	0.8	12/8	13/10	15/11	17/13	19/13	1.31	1.13	1.21	1.17	1.34
	U	1	32	0.8	14/8	15/10	16/11	18/12	19/13	1.59	1.35	1.31	1.37	1.34
LAP	L	1	32	0.5	11/11	21/20	30/29	39/38	48/48	0.59	0.74	0.77	0.79	0.79
	U	1	8	0.3	11/11	20/20	28/29	37/38	47/48	0.43	0.40	0.43	0.51	0.59
STO	L	1	16	0.7	10/9	21/20	31/28	38/34	46/44	0.74	0.87	0.98	1.01	0.96
	U	1	8	0.1	8/9	20/20	28/28	35/36	45/45	0.51	0.65	0.75	0.78	0.85
VEN	L	1	16	0.9	25/24	43/42	60/59	74/73	86/85	0.82	0.89	0.92	0.94	0.95
	U	1	8	0.6	17/17	36/35	53/52	68/68	82/82	0.54	0.77	0.85	0.87	0.89

Conclusions

- Careful exploitation of the component-wise contraction property of Jacobi iteration:
  - Monitor deviations from the expected convergence rate
  - Account for rounding error, but avoid stagnation
  - Cheap and periodic test for extension
- Potential savings of up to 60% for 3D Laplace benchmark
- Link with fault tolerance

"Tuning iterative solvers for fault resilience"
H. Anzt, J. Dongarra, E. S. Quintana-Ortí
ScalA'2015 (Tomorrow!)

Adaptive Precision Solvers for Sparse Linear Systems

THE UNIVERSITY of TENNESSEE KNOXVILLE

HARTWIG ANZT
JACK DONGARRA

UNIVERSITAT JAUME·I

ENRIQUE S. QUINTANA-ORTÍ