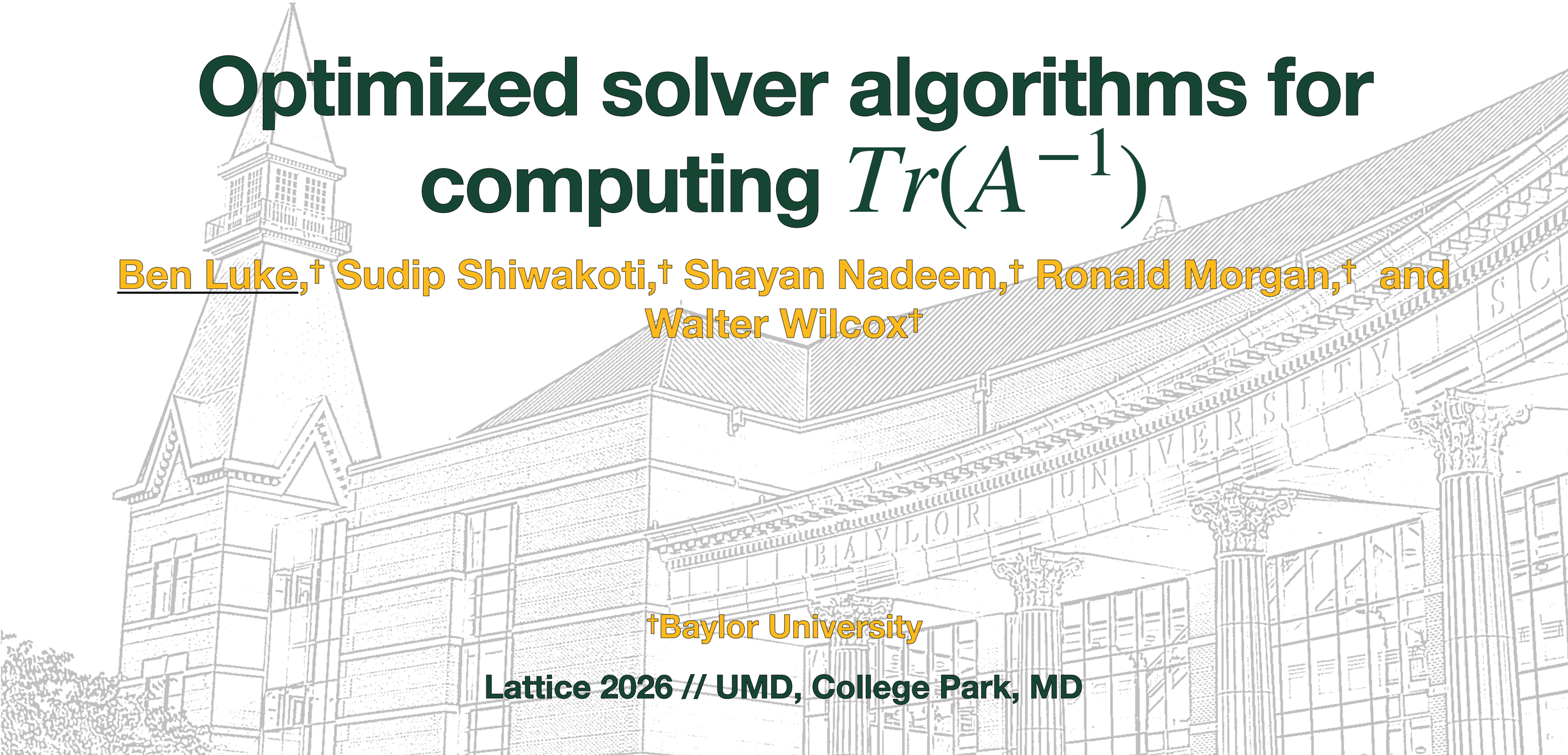
Baylor University

# Optimized solver algorithms for computing $Tr(A^{-1})$

Ben Luke,† Sudip Shiwakoti,† Shayan Nadeem,† Ronald Morgan,† and Walter Wilcox†

†Baylor University




Lattice 2026 // UMD, College Park, MD

# Introduction

- Estimating the trace of the inverse of a large quark matrix is a longstanding problem in lattice QCD.
- The standard estimator is **Hutchinson's method**, which requires solving systems of linear equations.
- In this talk I will explore:
  - **New solver algorithms** for Hutchinson's method
  - **Testing results**
  - **Singular value deflation** (SVD) and **polynomial approximation of** $A^{-1}$ for computing $Tr(A^{-1})$

# I. Hutchinson’s method & solving algorithms

# Hutchinson's method

Hutchinson's method for computing the trace of A-inverse is given by

$$Tr(A^{-1}) \approx \frac{1}{N} \sum_{i}^{N} b_i^{\dagger} A^{-1} b_i$$

where $b$ is a random noise vector (z-4 noise in this work). In practice, $N$ is not fixed but is dependent upon the desired trace tolerance.

It is prohibitively expensive to directly compute $A^{-1}$ directly, so we solve the system $Ax = b$. The trace then becomes

$$Tr(A^{-1}) \approx \frac{1}{N} \sum_{i}^{N} b_i^{\dagger} x_i$$

*Which linear equations solver do we use?*

# Solving algorithms at a glance

## BiCGStab

- 2 MVPs per iteration
- 2 dot products per iteration (extremely costly on a distributed system)
- Usually numerically stable and straightforward to implement
- Independent solve per noise vector

## BiCG-QCD

- BiCG algorithm specialized for QCD matrices
- Exploits the symmetry $\gamma_5 M = M^\dagger \gamma_5$ to eliminate a matrix vector product—1 MVP per iteration
- Premature breakdown is possible (zero divisors)
- Independent solve per noise vector

## Twin BiCG

- One base solve, apply parameters to all subsequent solves.
- 1 MPV per iteration after base solve
- Eliminates all dot products beyond the base solve
- Easily adaptable to batched solves/multiple RHS at a time

# Twin BiCG

Main Idea

*Twin* method applied to Bi-Conjugate Gradient (BiCG) recycles the same parameters from an initial base solve for all subsequent solves.

Theory

- Approximate solution is in the Krylov subspace,

$$\hat{x} \in \mathscr{K}^m(b, A) = \text{span}\{b, Ab, A^2b, \ldots, A^{m-1}b\} \implies \hat{x} = p(A)b$$

  Then

$$r = b - A\hat{x} = \pi(A)b$$

- To solve $Ax^{(i)} = b^{(i)}$, the polynomial $\pi(A) = I - Ap(A)$ must be small along the whole spectrum of $A$. This is true for each system solved.
- Reuse the polynomial on future solves!

# Benefits of Twin

- MVPs are effectively cut in half
- Dot products are eliminated beyond the base solve
  - Important for large scale problems on distributed memory where the relative cost of a DP is very high
- Each solve requires the same number of iterations
  - Easily adapted to a batched solve context (i.e. solve multiple RHS at once until convergence)—further improves efficiency
- Only one good solve is needed!
  - If the problem is very difficult, independent solves may yield several non-convergences. Twin circumvents this by picking a successful system and reusing the parameters.

Summary

$$Tr(A^{-1}) \approx \frac{1}{N}\sum_{i}^{N} b_i^{\dagger} A^{-1} b_i$$

1. $n_{MVP} \xrightarrow{Twin} \frac{1}{2} n_{MVP}$

   (after base solve)

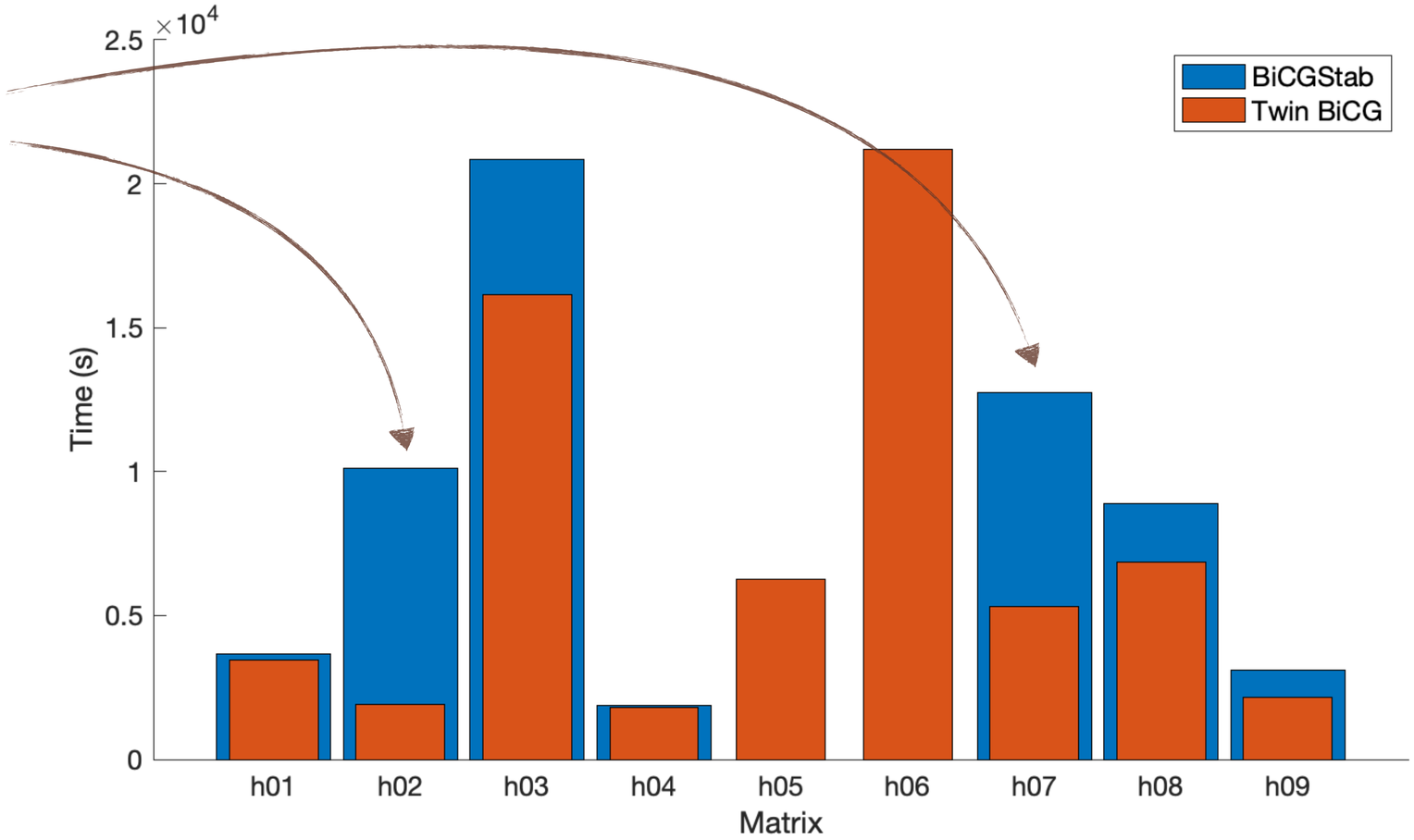
2. $\sum_i n_{DP,\,i} = n_{DP,\,1}$

3. $n_{iter,\,i>1} = n_{iter,\,i=1}$

# Twin BiCG vs BiCGStab

- Single thread, single CPU tests
- Directly compare algorithm performance of Twin BiCG against BiCGStab
- 9 matrices, $12^3 \times 16$ lattice, quenched, & $\kappa_{crit}$

Many skipped noises (non-converged) for BiCGStab

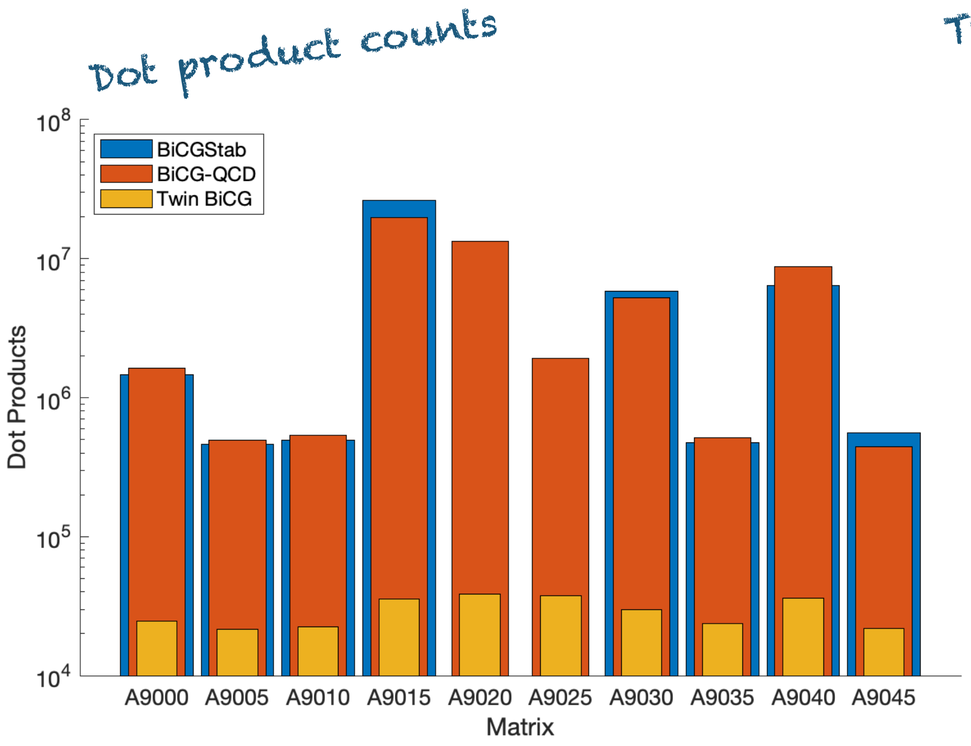


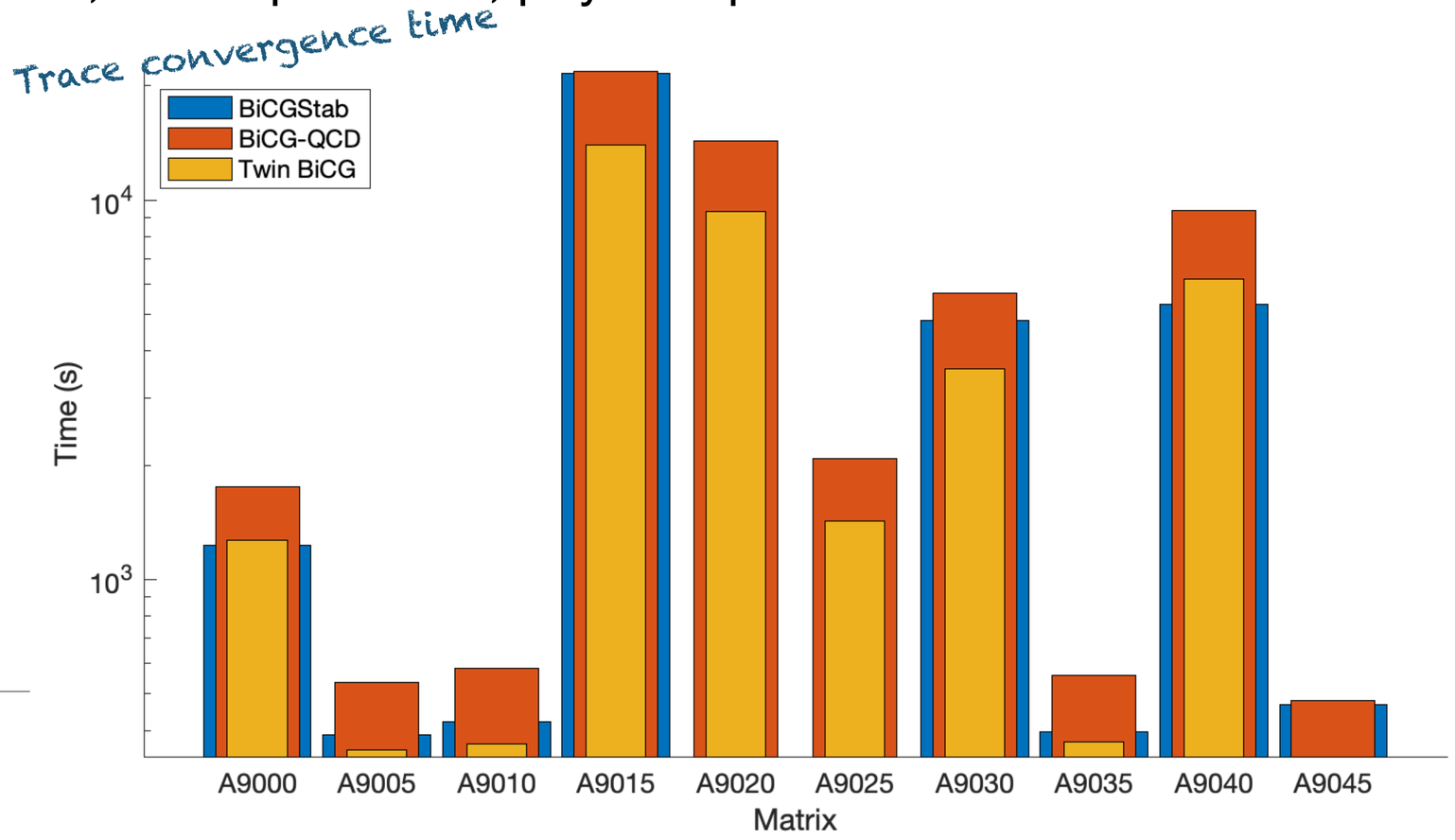

Takeaway

Twin BiCG outperforms BiCGStab on the level of algorithm structure.

# Twin BiCG, BiCGStab, BiCG-QCD

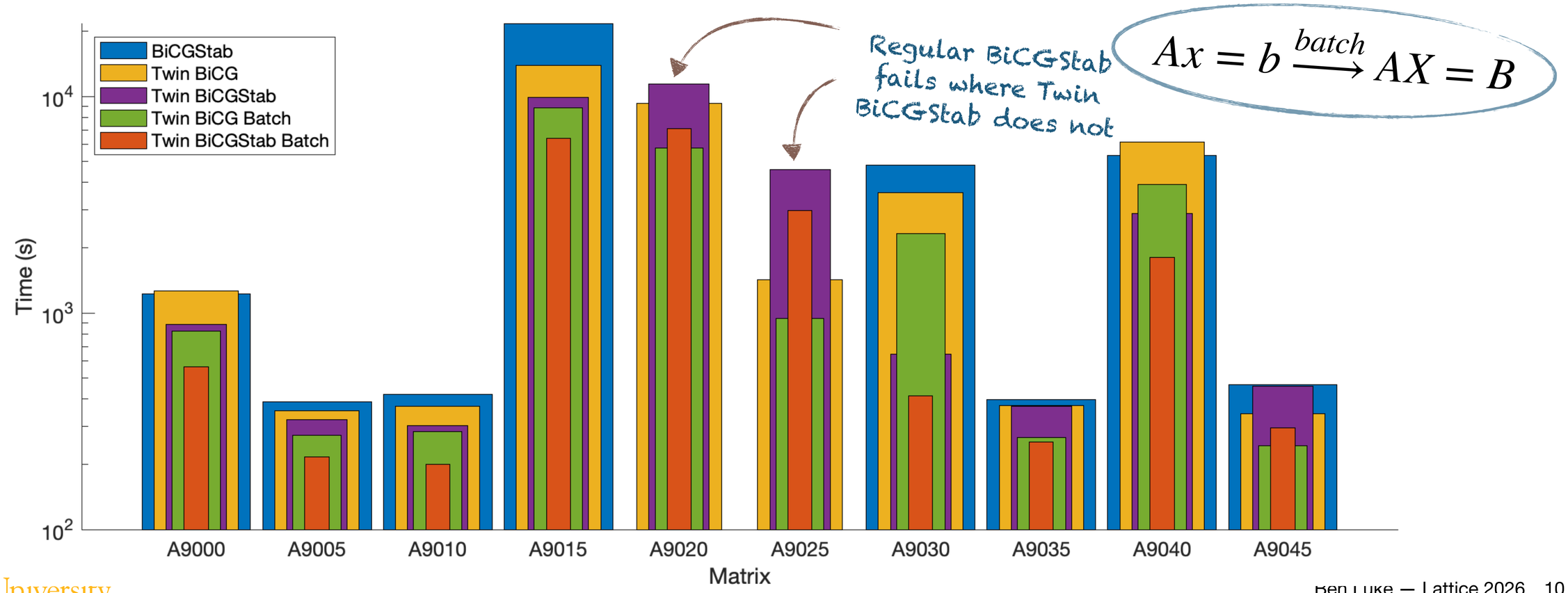
- Compare the three algorithms on NVIDIA A100 GPUs (and all tests hereafter)
- 10 matrices, $16^3 \times 48$ MILC lattice, semi-quenched, physical pion mass



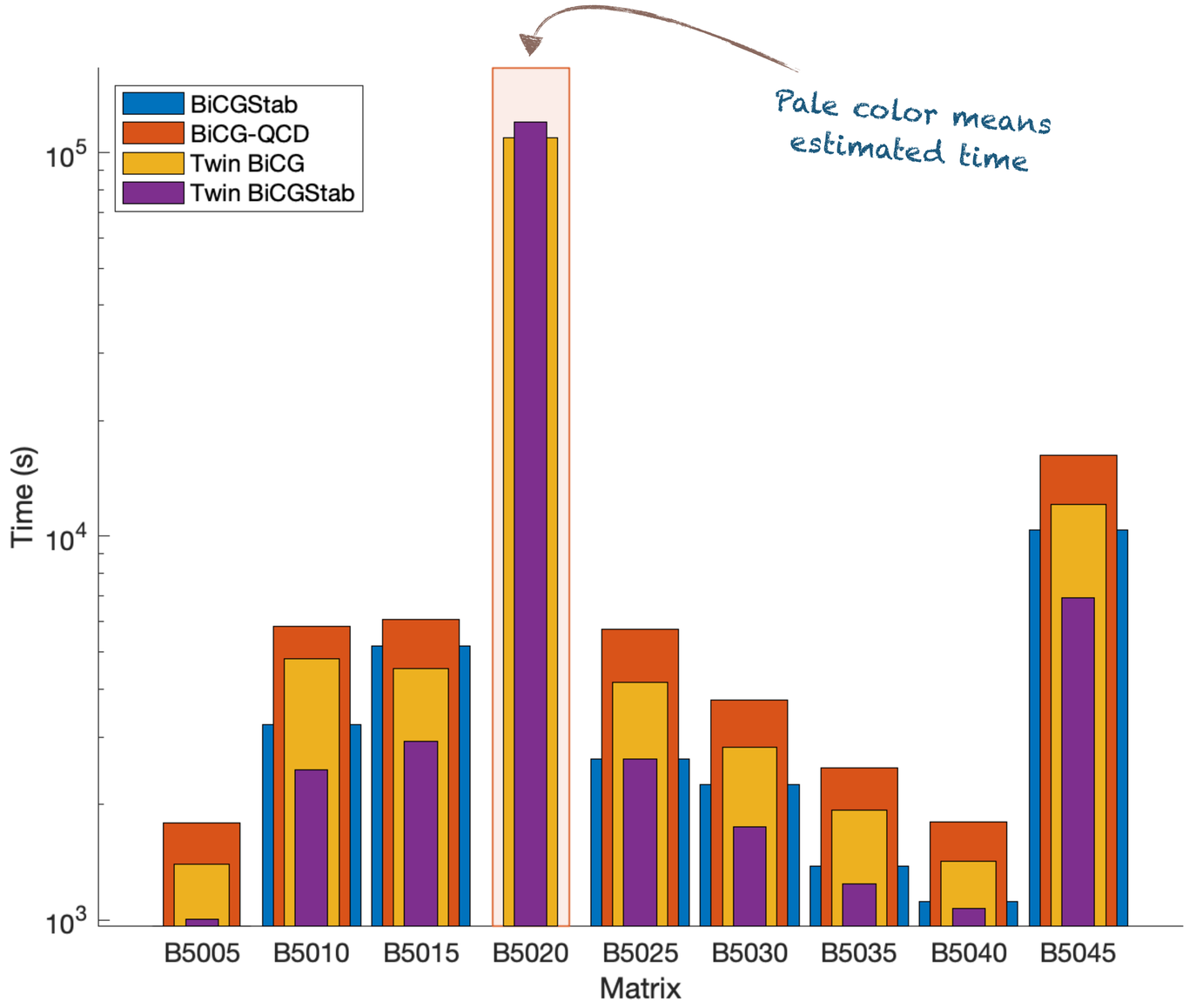
# Include batching

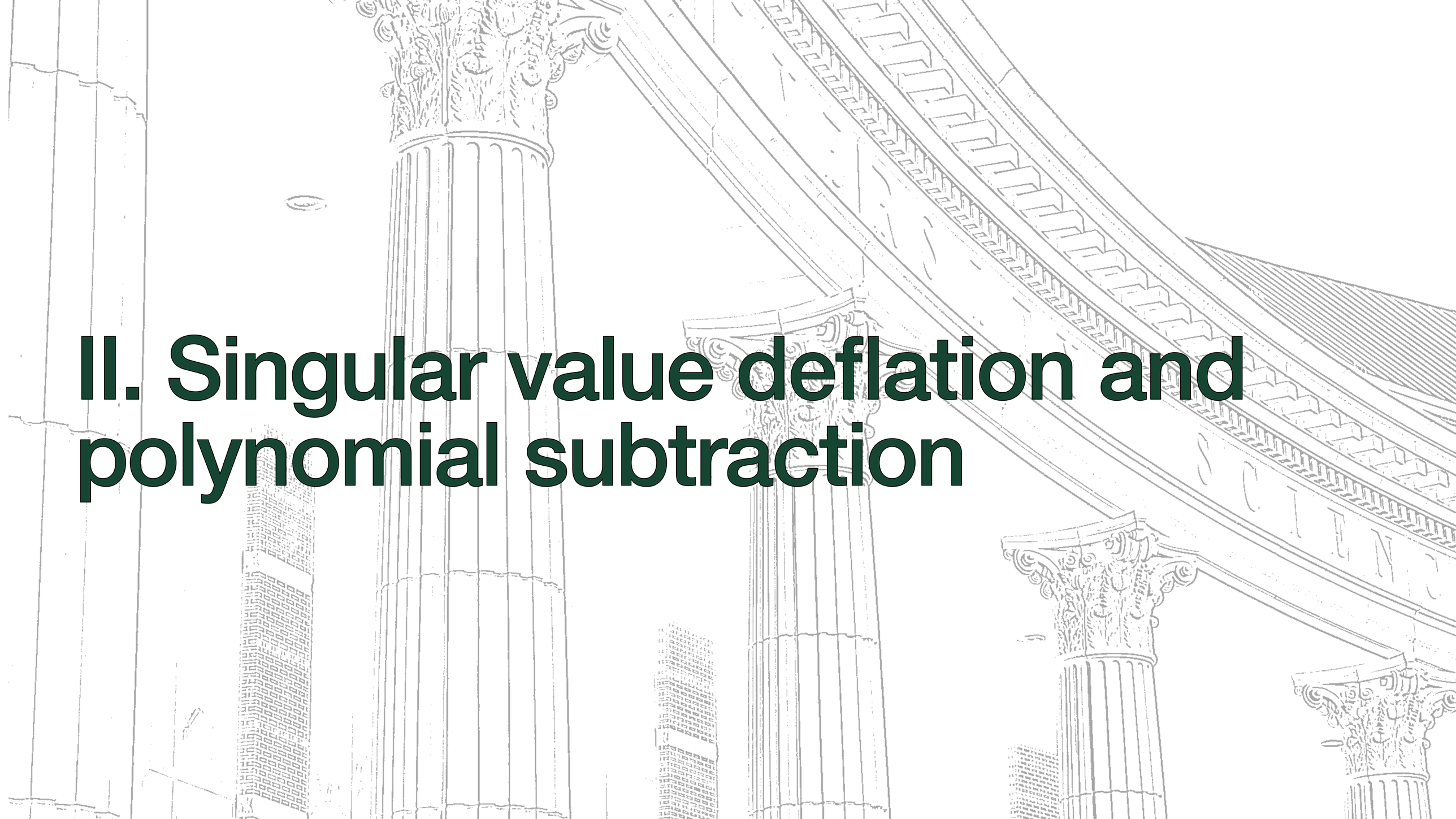
- Same matrices: $16^3 \times 48$ MILC lattice size, semi-quenched, physical pion mass. Trace tol. $t_{tr} = 0.0005 \times 16^3 \times 48$, solve tol. $t_{solve} = 10^{-3}$
- ***Batched*** solves for Twin methods (BiCGStab included). (“Batching” solves multiple RHS at once.)



# Larger matrix

- $24^3 \times 64$ MILC lattice, physical pion mass
- Trace tolerance
  $t_{tr} = 0.0005 \times 24^3 \times 64$
- Solve tolerance $t_{solve} = 10^{-3}$
- In instances of non-convergence, a good-faith time estimate is given, when possible.





# II. Singular value deflation and polynomial subtraction

# Polynomials and singular values

- The solver algorithms discussed stand in service of a more sophisticated evaluation of the trace,

$$Tr(A^{-1}) \approx \underbrace{Tr(A^{-1} - p(A))}_{(1)} + \underbrace{Tr\left( p(A) - \sum_i \frac{1}{\sigma_i} v_i u_i^\dagger \right)}_{(2)} + \underbrace{Tr\left( \sum_i \frac{1}{\sigma_i} v_i u_i^\dagger \right)}_{(3)}$$

- Polynomial $p(A) \approx A^{-1}$, vectors $\{v_i\}$ and $\{u_i\}$ are right and left singular vectors, respectively, $\{1/\sigma_i\}$ are the largest singular values of $A^{-1}$.
- When $p(A)$ is a strong approximation of $A^{-1}$, very few noises are needed for trace (1).
- Trace (2) deflates out the largest singular values, and trace (3) is computed exactly.
- $p(A)$ and SVD can be expensive to generate, but payoff is anticipated for difficult problems.

# Polynomial approximation of $A^{-1}$

- Recall that the solution is extracted from the Krylov subspace

$$\hat{x} \in \mathscr{K}^m(b, A) = \text{span}\{b, Ab, A^2b, \ldots, A^{m-1}b\} \Longrightarrow \hat{x} = p(A)b$$

high-degree

- The polynomial $p(A)$ must approximate $A^{-1}$ well
- Run non-symmetric Lanczos algorithm (specialized for QCD) on $A$
- In testing we focus on $\underbrace{Tr(A^{-1} - p(A))}_{(1)} + \underbrace{Tr(p(A))}_{(2)}$

| Matrix | h01 | h02 | h03 | h04 | h05 | h08 | h09 | h10 |
|---|---|---|---|---|---|---|---|---|
| noises (1) | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| noises (2) | 323 | 114 | 585 | 235 | 74 | 145 | 128 | 59 |

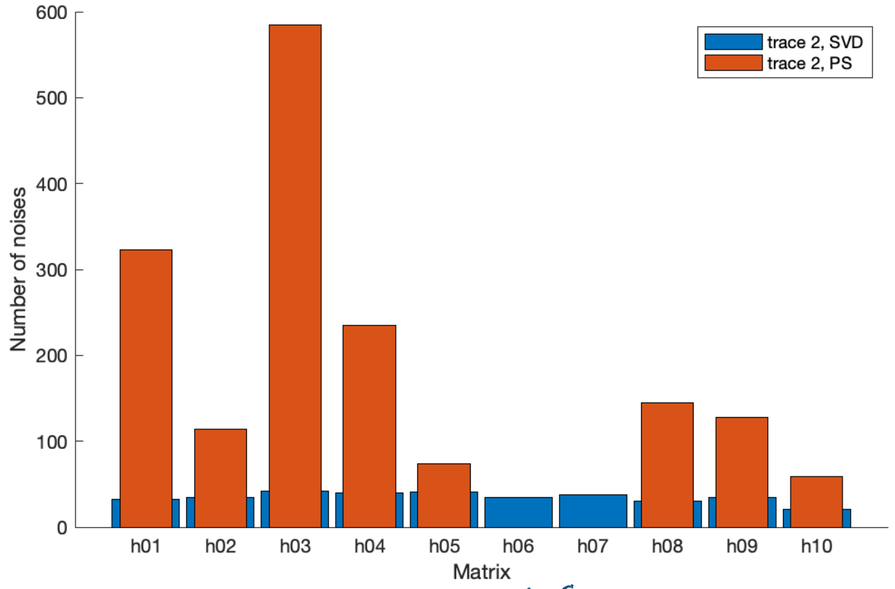
Noises required for each trace on $24^4$ quenched lattice at $\kappa_{crit}$. Trace tolerance = $0.0005 \times 24^4$.

# Singular value deflation (SVD)

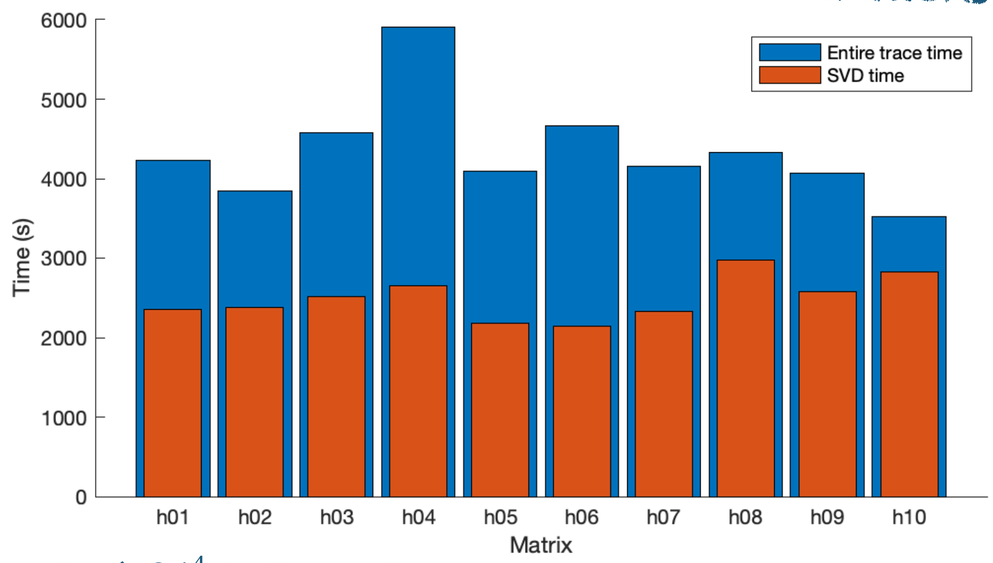
- Run the symmetric Lanczos algorithm on $\gamma_5 A$ (special for QCD) to generate singular values and vectors
- We test the full polynomial-subtracted, SVD trace,

$$Tr(A^{-1}) \approx \underbrace{Tr(A^{-1} - p(A))}_{(1)} + \underbrace{Tr\left(p(A) - \sum_i \frac{1}{\sigma_i} v_i u_i^\dagger\right)}_{(2)} + Tr\left(\sum_i \frac{1}{\sigma_i} v_i u_i^\dagger\right)$$

| Matrix | nsvals |
|---|---|
| h01 | 62 |
| h02 | 71 |
| h03 | 69 |
| h04 | 70 |
| h05 | 73 |
| h06 | 69 |
| h07 | 61 |
| h08 | 73 |
| h09 | 56 |
| h10 | 53 |




SVD time contributions

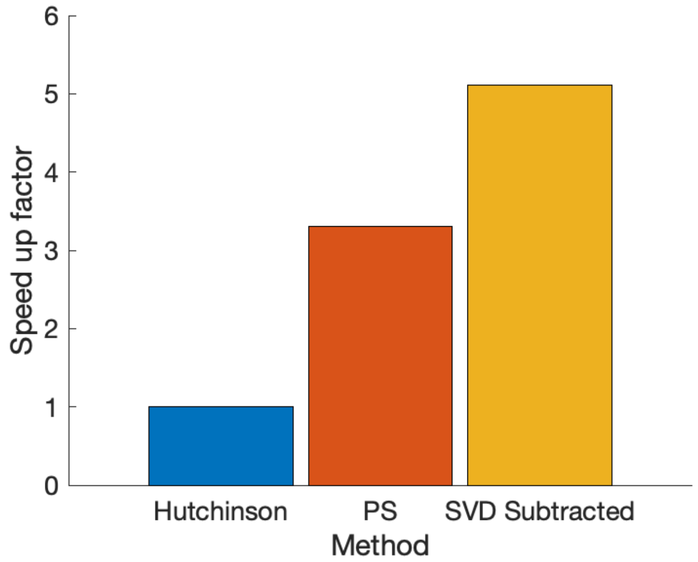



Noises required for each trace on quenched $24^4$ lattice at $\kappa_{crit}$.
Trace tolerance = $0.0005 \times 24^4$.

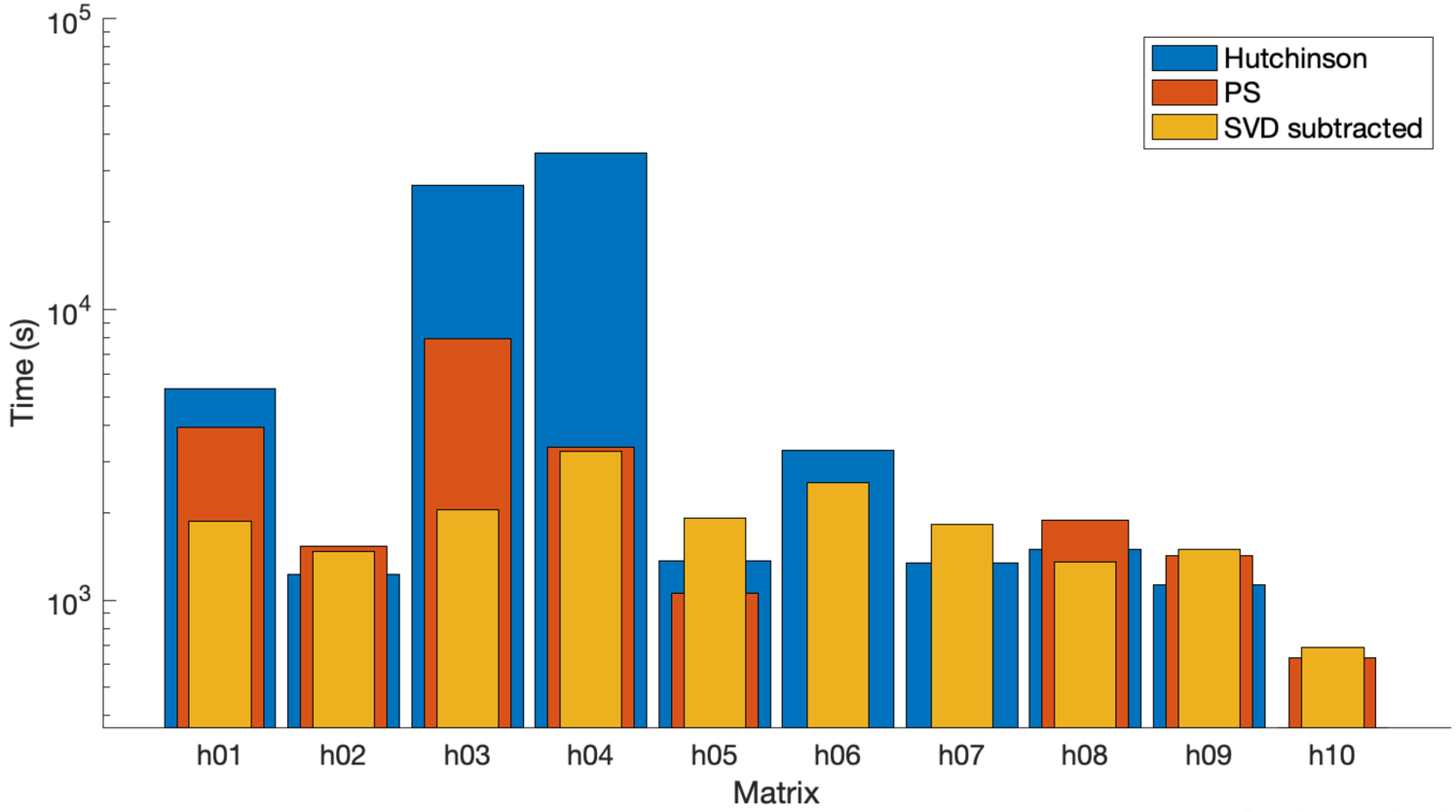
# Trace times

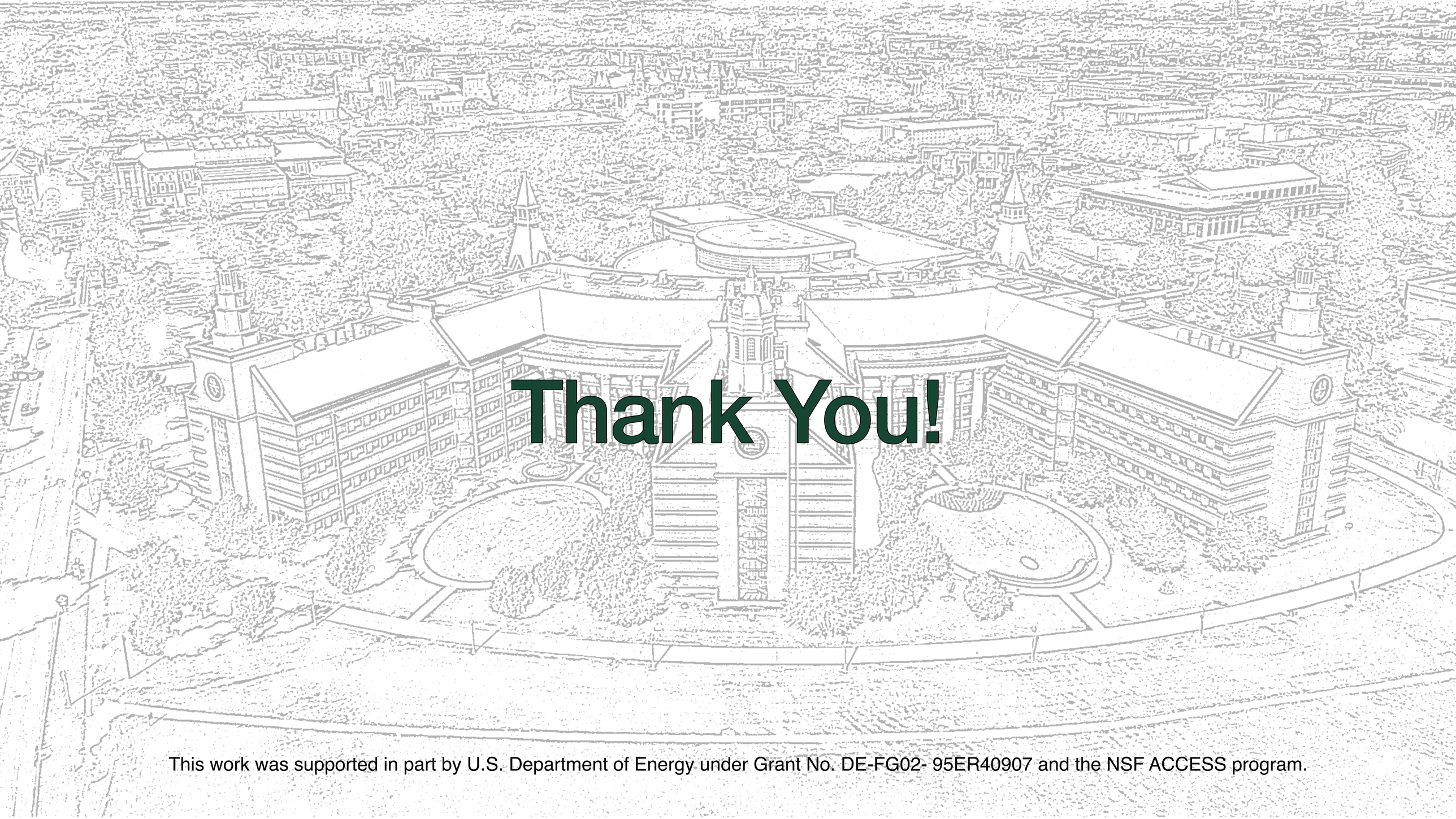
## Polynomial subtraction and SVD

- Compare the total times for regular Hutchinson's method, a polynomial subtraction trace, and a full polynomial subtraction and SVD subtracted trace.




(Comparison excludes matrices h06 and h07 which are incomplete)

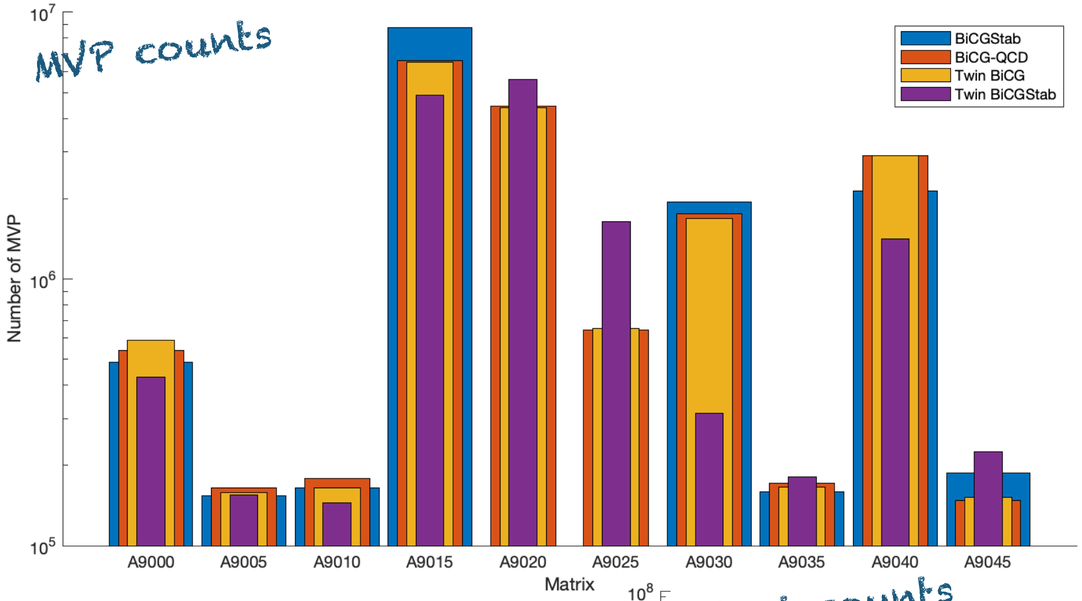


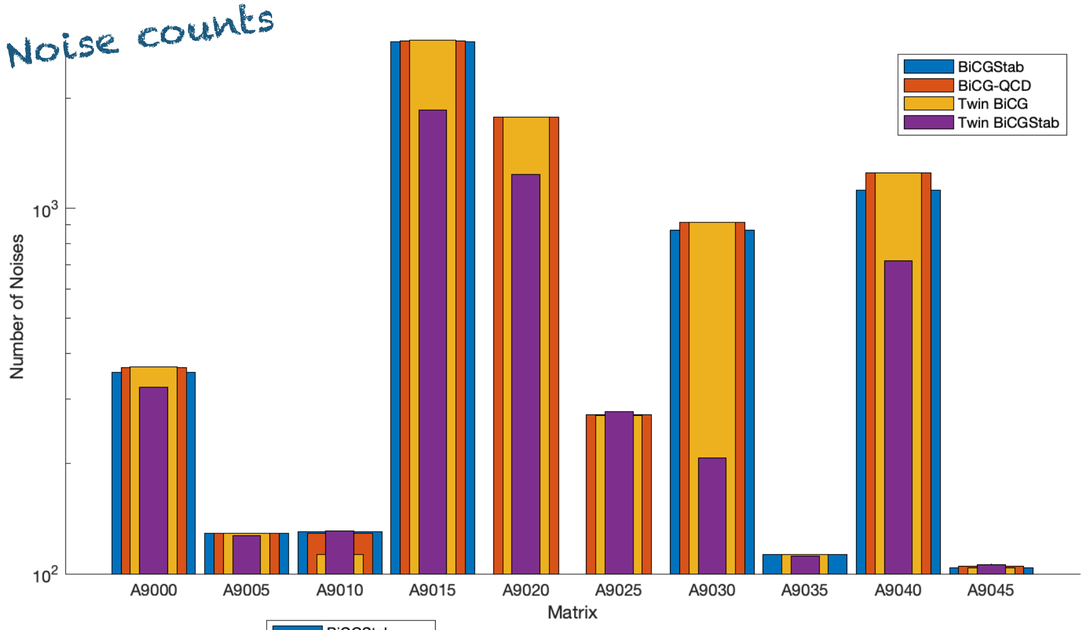


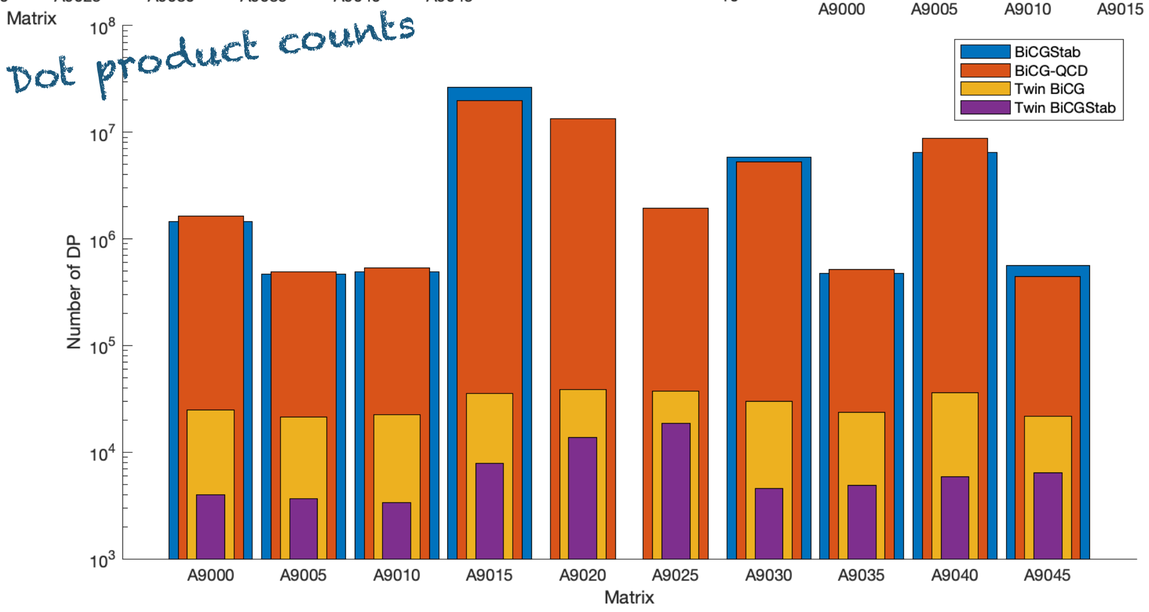
# Thank You!

This work was supported in part by U.S. Department of Energy under Grant No. DE-FG02- 95ER40907 and the NSF ACCESS program.

# A matrices, physical pion mass



# B matrices, physical pion mass