Exploiting the potential of the PRIMME eigensolver
A general overview

Andreas Stathopoulos

Computer Science Department
College of William and Mary

PRIMME developers: James McCombs, Eloy Romero, A.S., Lingfei Wu

Partial support from: NSF SI2, NSF CCF, DOE SciDAC

The eigenvalue and singular value problems

Given A Hermitian, find *nev* eigenpairs:

$$\boxed{Ax_i = \lambda_i x_i, \quad i = 1, \ldots, nev}$$

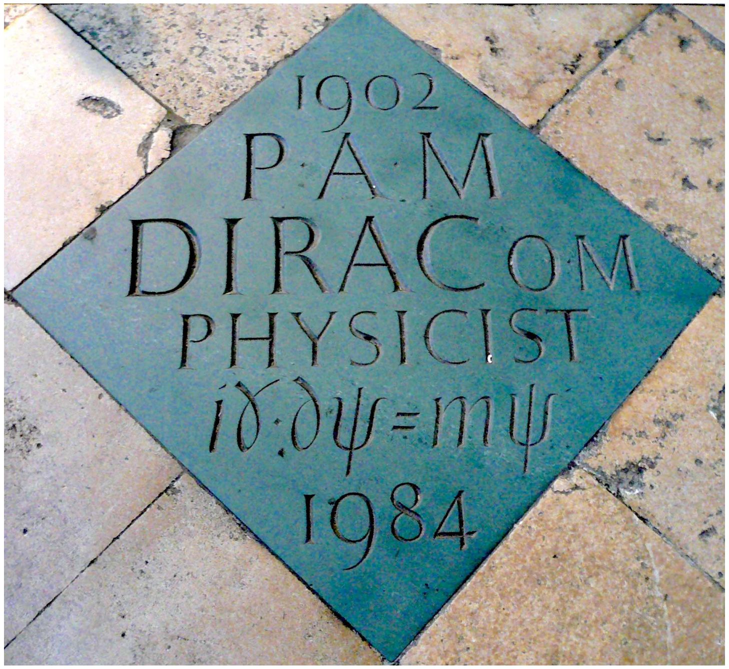
Given A any square or rectangular matrix, find *nsv* singular triplets:

$$\boxed{Av_i = \sigma_i u_i, \quad i = 1, \ldots, nsv}$$

- One of the dimensions of A can be $O(10^6 - 10^9)$
- A is sparse or provided through a matvec function

Why this problem?

- Quantum mechanics (Scrhöndinger equation) a successful model of the world!



Lattice QCD
Nuclear physics
Atomic physics
Materials science

- Also macroscopic phenomena that involve vibrations/frequencies

Structural engineering
Fluids

Eigenvalues and singular values important tool

- in computational sciences

  Stability analysis (norm/condition number estimation)

  Low rank approximations (model reduction)

  Variance reduction in Monte Carlo methods

  Deflation preconditioning

- in graph analysis

  Graph partitioning, coloring

  Network analysis

- in data sciences

  Principal Component Analysis, Latent Semantic Indexing, Page-rank

  Combining with sparse approximations (sparse+low rank)

A walk through the state-of-the-art eigenvalue iterative methods

Without preconditioning, unrestarted Lanczos or Arnoldi are the optimal methods in terms of matvecs

Add preconditioning to the basic iteration $\Longrightarrow$ Generalized Davidson (GD)
Like FGMRES. More expensive per iteration but very flexible

Work on a block of vectors per iteration $\Longrightarrow$ block Lanczos, block GD.
More robust for multiplicities but slower convergence

A walk through the state-of-the-art eigenvalue iterative methods

Without preconditioning, unrestarted Lanczos or Arnoldi are the optimal methods in terms of matvecs

Add preconditioning to the basic iteration ⟹ Generalized Davidson (GD)
Like FGMRES. More expensive per iteration but very flexible

Work on a block of vectors per iteration ⟹ block Lanczos, block GD.
More robust for multiplicities but slower convergence

Yet,

Increasing memory and iteration costs
⇓
Restarting
⇓
slow convergence, misconvergence

A walk through the state-of-the-art eigenvalue iterative methods

Advanced restarting solutions make significant difference

Thick restart = Implicit restart

Keep nearby spectrum at restart

+k restart

Keep approximations from previous iteration acts like a 3-term CG iteration

Must use both to obtain near optimal restarting

Efficient restarting is one of PRIMME's backbones

A walk through the state-of-the-art eigenvalue iterative methods

Approach as a non-linear problem:

Newton ⟹ inner-outer methods, TraceMin or Jacobi Davidson
Stop inner when no further benefits ⟹ near optimal JDQMR

also one of PRIMME’s backbones

A walk through the state-of-the-art eigenvalue iterative methods

Approach as a non-linear problem:

Newton $\Longrightarrow$ inner-outer methods, TraceMin or Jacobi Davidson
Stop inner when no further benefits $\Longrightarrow$ near optimal JDQMR

$\boxed{\text{also one of PRIMME's backbones}}$

Non-linear CG $\longrightarrow$ LOPCG (locally optimal restarting)
Equivalent to +k restarting
For large *nev* needs large block (LOBPCG), subspace acceleration, or both:

$\boxed{\text{PRIMME's GD+k a more flexible, near optimal method}}$

A walk through the state-of-the-art eigenvalue iterative methods

For large *nev* and for interior eigenproblems also much interest in:

Polynomial filtering where $p(A)$ is the operator in Lanczos or GD
– Reduces iteration/orthogonalization costs and parallel syncs
– However, filter tuning is an art, and matvecs increase

A walk through the state-of-the-art eigenvalue iterative methods

For large *nev* and for interior eigenproblems also much interest in:

Polynomial filtering where $p(A)$ is the operator in Lanczos or GD
– Reduces iteration/orthogonalization costs and parallel syncs
– However, filter tuning is an art, and matvecs increase

Contour integration, $\oint_\Gamma (A - zI)^{-1} dz$, Γ encloses the desired spectrum
– An approximate spectral projector used with subspace iteration
– Linear systems must be solved with complex shifts for each vector in subspace
– Could be very efficient with direct linear solvers
– Various levels of parallelism
– Without direct solvers its performance worse than filtering

State-of-the-art software for Eigenproblems

- Without preconditioning:

  ARPACK (Implicitly Restarted Arnoldi)
  TRLAN (Implicitly Restarted Lanczos)
  FILTLAN (Polynomially filtered unrestarted Lanczos)

- With preconditioning, general purpose is more challenging:

  Anasazi (block GD, LOBPCG, IRTR, in Trilinos)
  BLOPEX (LOBPCG)
  SLEPc (JD and most major methods, extends PETSc)
  FEAST (Contour integration, available in MKL)
  PRIMME (block GD+k/JDQMR, most major methods)

PReconditioned Iterative MultiMethod Eigensolver PRIMME

A no-shortcuts eigensolver since 2006 www.github.com/primme

- Robustness
- Flexibility
- Efficiency

PReconditioned Iterative MultiMethod Eigensolver PRIMME

A no-shortcuts eigensolver since 2006 www.github.com/primme

- Robustness

  achieve full accuracy allowed by machine precision

  effectively resolve multiple eigenvalues

  converge to interior eigenvalues

  avoid misconvergence problems

  avoid stagnation problems

  attention to all implementation details

PReconditioned Iterative MultiMethod Eigensolver PRIMME

A no-shortcuts eigensolver since 2006 www.github.com/primme

- Flexibility

  user-provided matvec, preconditioner, convergence test

  a parametrized solver allows 12 methods and their block versions

  expert defaults and easy interface for end-users

  fully customizable for various levels of expertise

  accepts multiple initial guesses

  finds few or many eigenpairs in various spectrum locations

  float, double, complex, double complex

  BSD 3-clause license

  interfaces in Fortran, MATLAB, Octave, Python, R

PReconditioned Iterative MultiMethod Eigensolver **PRIMME**

A no-shortcuts eigensolver since 2006 `www.github.com/primme`

- Efficiency

  small memory requirements

  near optimal convergence for small *nev* even with limited memory

  dynamically chooses the best method

  both algorithmic and HPC efficiency

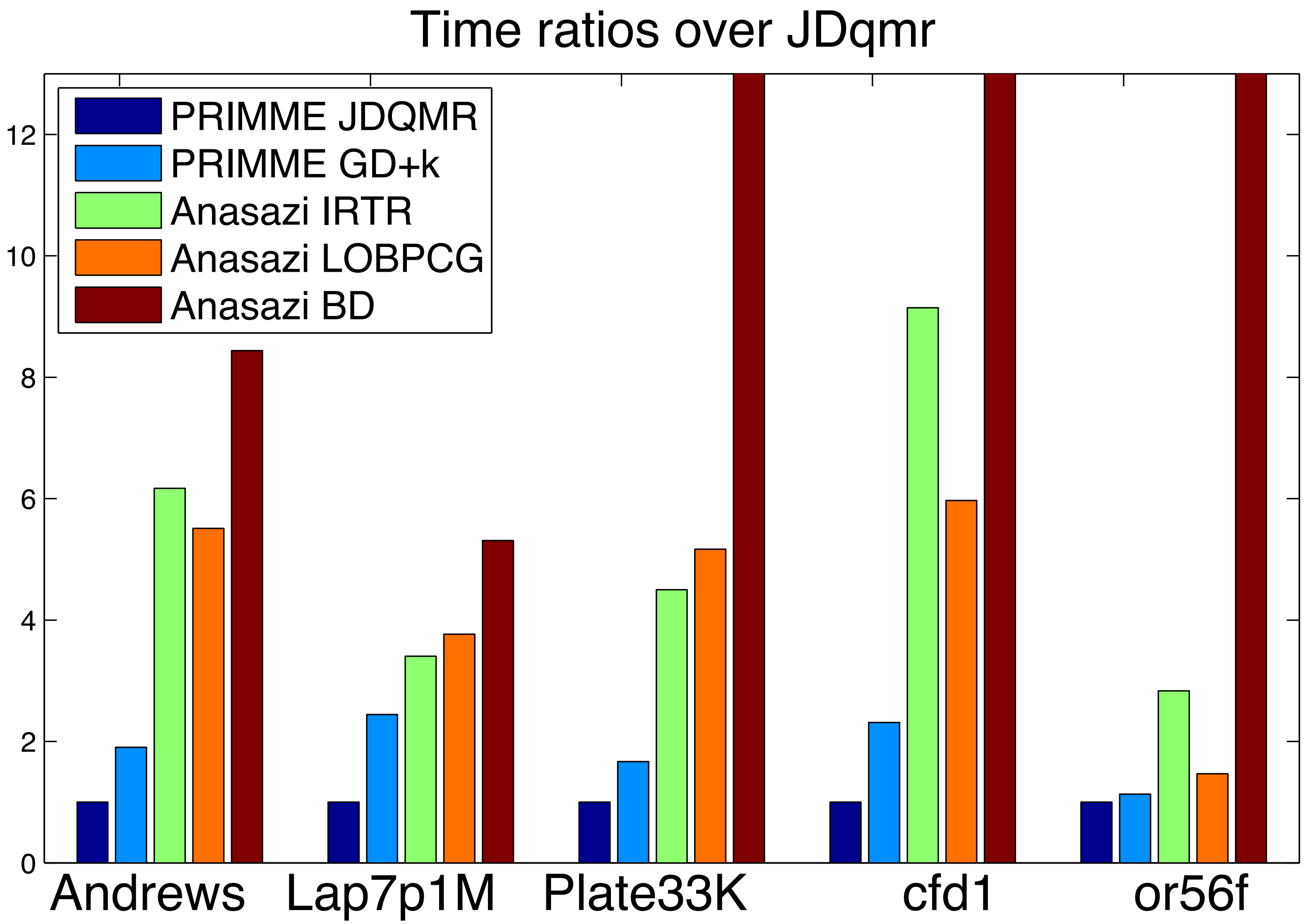
  blocking and locking techniques

  almost all computation on top of BLAS/LAPACK interface

  SPMD parallelism (with user provided globalSum)

  multithreaded if threaded operators and BLAS/LAPACK

PRIMME vs Anasazi for five lowest eigenpairs



The SVD problem

An eigenvalue problem either on A^TA or on $\begin{pmatrix} 0 & A^T \\ A & 0 \end{pmatrix}$

– Solving A^TA faster but with relative error $\frac{\|A\|^2}{\sigma_i^2}\varepsilon_{mach} = \begin{cases} \varepsilon_{mach}, & \sigma_1 = \|A\| \\ \kappa(A)^2\varepsilon_{mach}, & \sigma_N \end{cases}$

– Augmented difficult interior problem

– Lanczos Bidiagonalization (LBD) $\Leftrightarrow$ Lanczos on A^TA but more accurate
With restarting, eigenmethods on A^TA still faster

– JDSVD is a preconditioned inner-outer method on the augmented.

The SVD problem: PRIMME's two stage approach

1. Use best eigensolver on $A^T A$ up to accuracy limit
2. If further accurcy needed, continue on the augmented

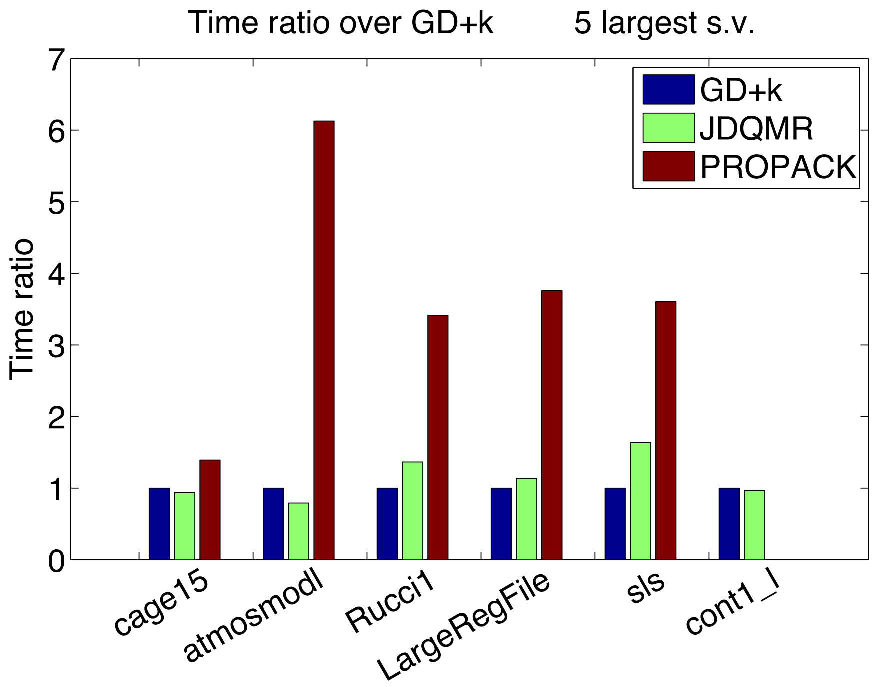
   - Fine-tunes methods and their transition
   - Allows for preconditioning
   - Carries all PRIMME functionality and interfaces

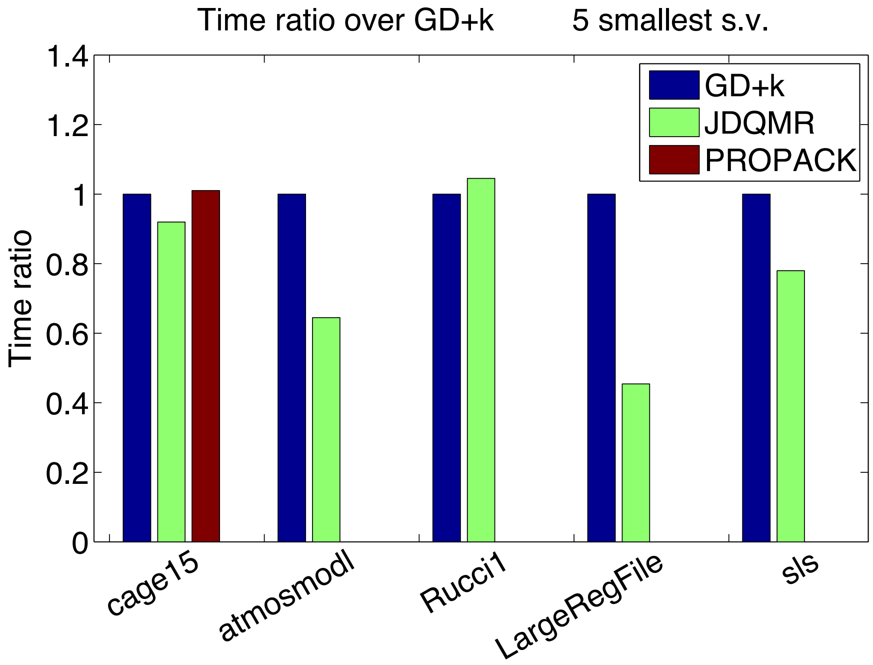
Other SVD-specific software:

PROPACK (implicitly restarted LBD in F77)
SLEPc (thick restarted LBD in C)
IRLBA (thick restarted block LBD in R)
JDSVD (only in MATLAB)

Finding a few singular values

Comparing against the industry's standard PROPACK:





PRIMME more efficient and significantly more robust

Challenges in all current methods

What is the best way to compute:

1. MANY eigen/singular values $O(10^2 - 10^4)$
2. INTERIOR eigenvalues inside the spectrum
3. MANY INTERIOR eigenvalues

Increasingly needed in:

- Quantum chemistry (many occupied states or excited states)
- Low rank approximation (model reduction, variance reduction, embeddings)
- Computation of trace, determinant, density of states, etc
- Smallest singular values (stability analysis, condition number estimation)

Many eigenvalues challenges

Orthogonalization

- Cost grows as $O(nev^2N)$

Extraction method = projection and solution of projected problem

- Small basis size ⇒ slow convergence
- Large basis size cost grows as $O(basisSize^3 + basisSize^2N)$

Many eigenvalues challenges

Orthogonalization

- Cost grows as $O(nev^2N)$

Extraction method = projection and solution of projected problem

- Small basis size ⇒ slow convergence
- Large basis size cost grows as $O(basisSize^3 + basisSize^2N)$

Inner-Outer

- Reduces extraction and ortho costs
- More inner iterations increases total “matvecs”
- Filters or inner-outer methods?

Many eigenvalues challenges

Orthogonalization

- Cost grows as $O(nev^2N)$

Extraction method = projection and solution of projected problem

- Small basis size ⇒ slow convergence
- Large basis size cost grows as $O(basisSize^3 + basisSize^2N)$

Inner-Outer

Block methods

- Induce BLAS 3 ops in extraction and restarting
- Optimal block size (convergence vs robustness vs GFLOPS)?
- Allows ortho to use BLAS 3 (to do in PRIMME)
- Sparse MV is memory bound so special block MV needed

Many eigenvalues challenges

Orthogonalization

- Cost grows as $O(nev^2 N)$

Extraction method = projection and solution of projected problem

- Small basis size ⇒ slow convergence
- Large basis size cost grows as $O(basisSize^3 + basisSize^2 N)$

Inner-Outer

Block methods

Communication avoiding

- Based on s-step iterative methods
- s typically cannot be very large
- Also block ortho concerns vs convergence
- Not necessary less memory accesses

Interior eigenproblem challenges

Extraction method

- Rayleigh-Ritz method fastest when it works
- Refined slower but robust
- Harmonic not easy to black box

Robust implementations in PRIMME

Restarting method

- Thick restarting necessary
- Effective +k restarting is involved (but implemented!)

Block methods

- May improve robustness at the cost of convergence speed

Preconditioning

- Use when available, but what is a good “indefinite” preconditioner?
- Use filters or inner-outer methods?

Many Interior challenges

Filtered, Inner-Outer

- What is optimal inner degree? How about deg=2?
- How do we pick the range to filter?
- How do we know the #evals?

Contour integration

- If direct solvers possible, good approach
- With iterative linear methods, not so competitive

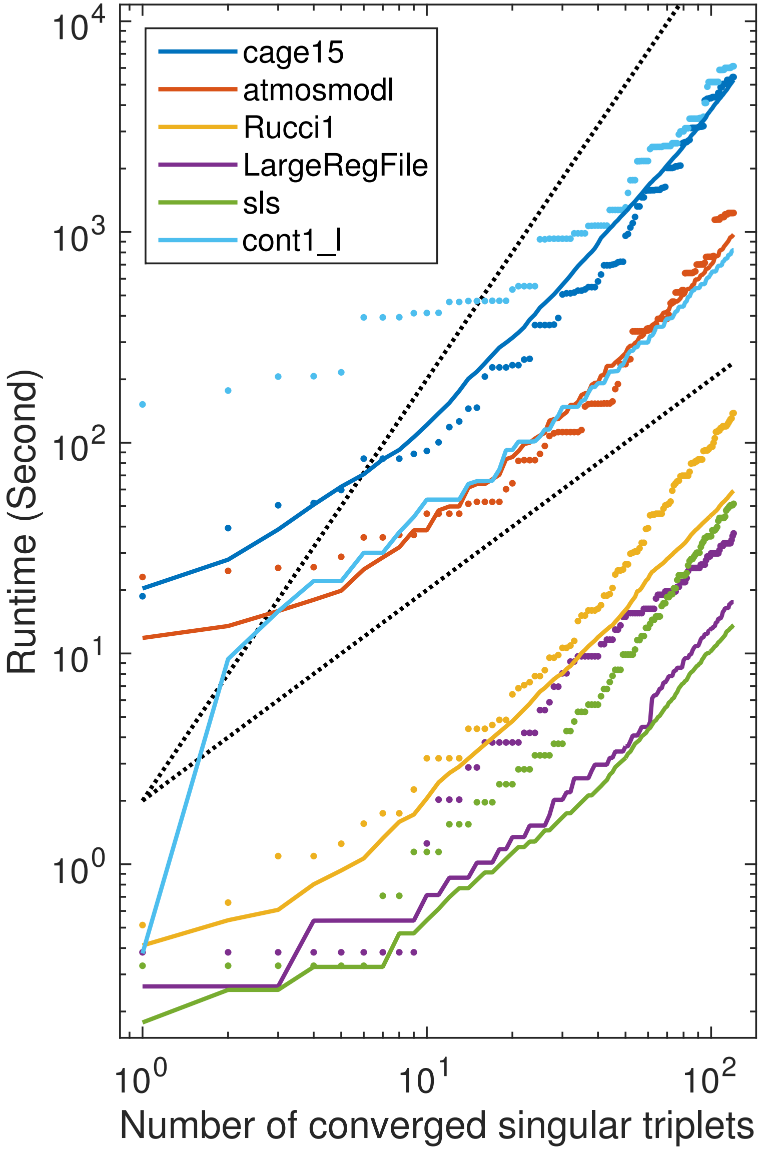
Spectrum slicing

- Iteration and ortho costs only for the slice
- Multiple levels of parallelism
- Load balancing of the slices (#evals and convergence rate)?
- General technique that can be used with many methods

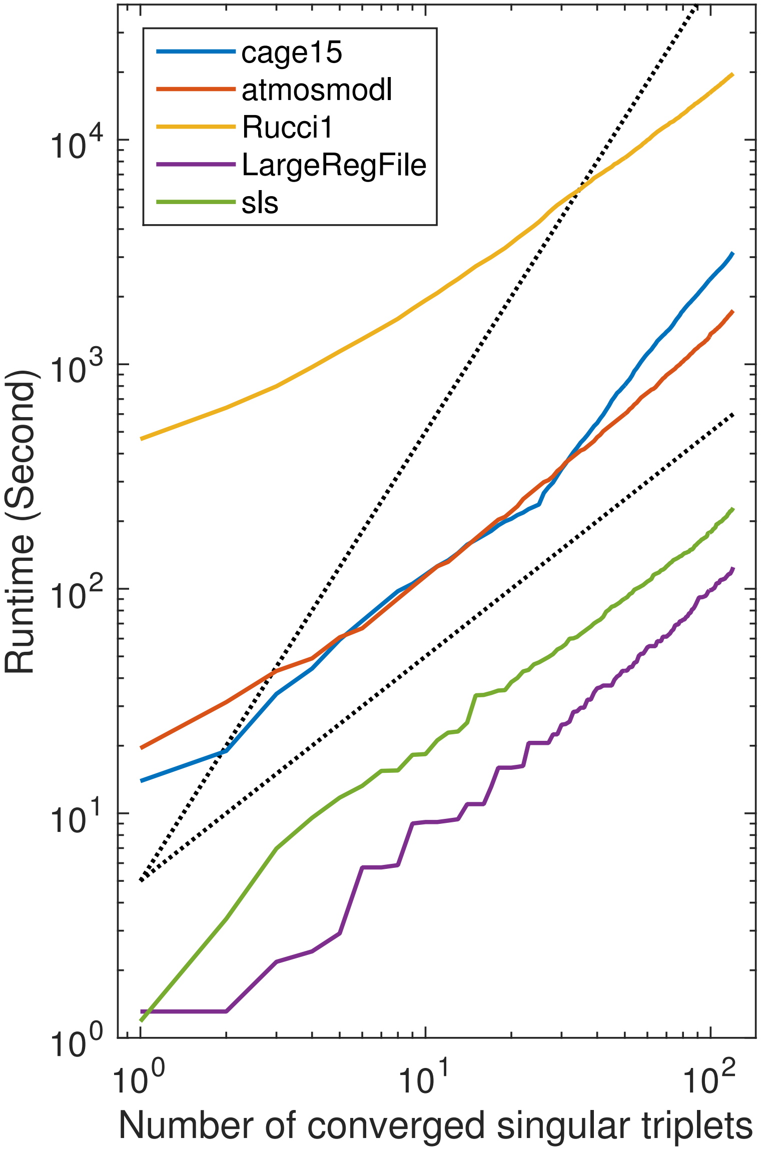
Results and future projections

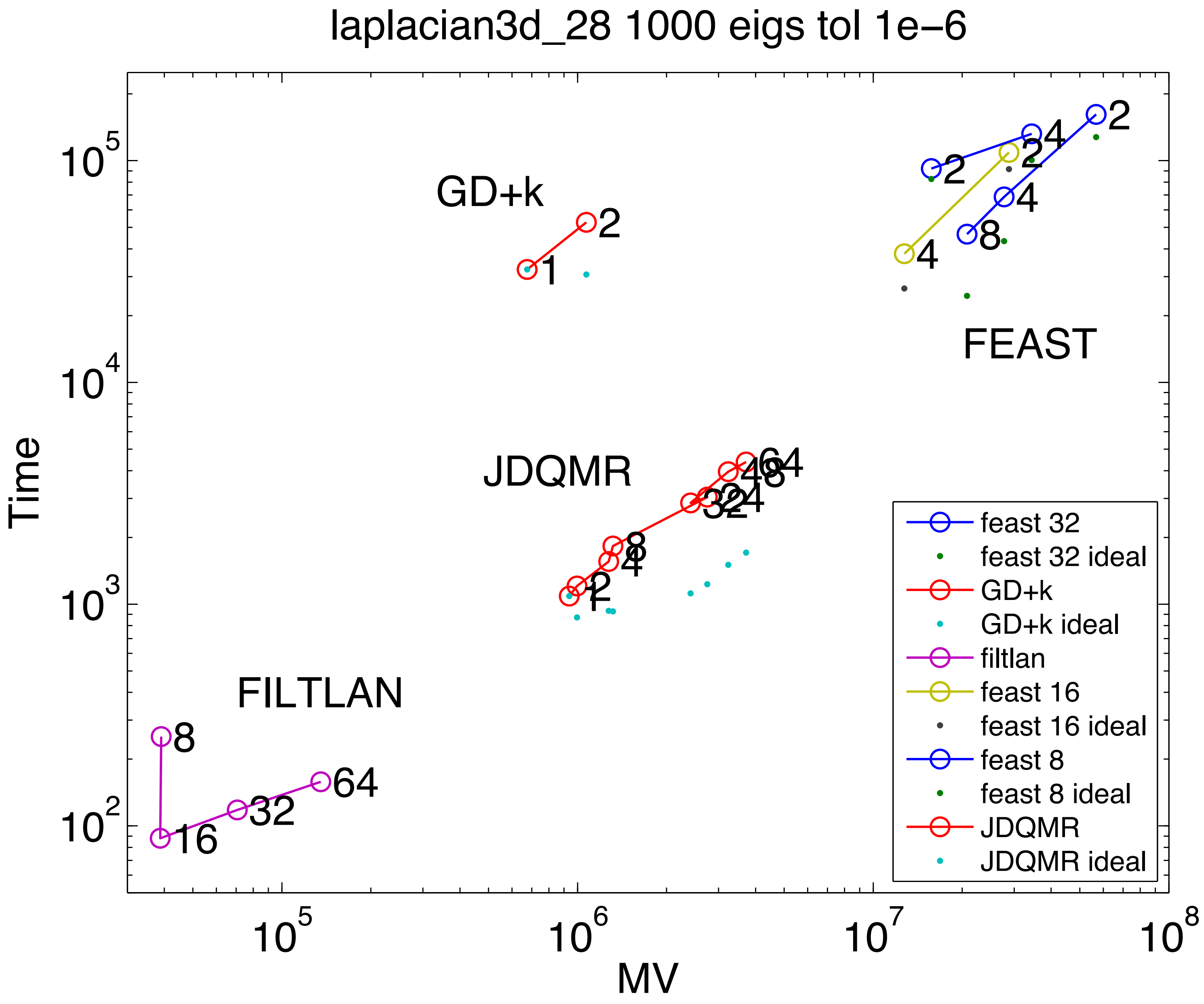
vs SLEPc (dots)

Seeking the largest 120 singular triplets without preconditioning



Seeking the smallest 120 singular triplets with preconditioning




laplacian3d_28 1000 eigs tol 1e−6
Time
MV
10^2
10^3
10^4
10^5
10^5
10^6
10^7
10^8
GD+k
1
2
FEAST
2
4
2
4
8
2
4
JDQMR
1
2
4
8
32
24
48
64
FILTLAN
8
16
32
64
feast 32
feast 32 ideal
GD+k
GD+k ideal
filtlan
feast 16
feast 16 ideal
feast 8
feast 8 ideal
JDQMR
JDQMR ideal

Discussion

Which method?

– DYNAMIC almost identical to best near optimal method. Use defaults!

Highly clustered spectrum or multiplicities

– PRIMME with block size 1 or 2 sufficient

Very low accuracy

– DM/ML need low accuracy (1e-2) $\Longrightarrow$ use PRIMME with large block size

Very high accuracy

– Rare, but PRIMME can obtain $\|A\|\varepsilon_{mach}$

Large number of eigenpairs needed

– For very large *nev* switch to unrestarted or large block size

PRIMME as randomized method

– Provide *k* random initial guesses to PRIMME

– PRIMME has the advantage of a sophisticated acceleration