Preconditioning finite difference matrices from density functional theory

Shikhar Shah Edmond Chow

School of Computational Science and Engineering
Georgia Institute of Technology

International Conference on Preconditioning Techniques
for Scientific and Industrial Applications
10 June 2024

Overview

1. A family of block linear systems
2. A method of efficient preconditioning
3. A pair of welcome surprises

Overview

1. A family of block linear systems
2. A method of efficient preconditioning
3. A pair of welcome surprises

Computing the functional trace of a linear operator

- We want to approximate

$$\int_0^\infty \mathrm{Tr}[f(\mathcal{A}(\omega))]\mathrm{d}\omega$$

  by computing the trace at quadrature points ω_k
- Any method we use to approximate the trace will require performing $\mathcal{A}(\omega_k)V$ for a block of vectors V (e.g., subspace iteration, Arnoldi quadrature, Hutchinson trace estimator, ...)
- $\mathcal{A}(\omega_k)$ cannot be constructed explicitly or applied in some matrix-free fashion; instead, applying $\mathcal{A}(\omega_k)$ requires an expensive and complicated sequence of operations

Sternheimer equations: a parameterized family

The multiplication $\mathcal{A}(\omega_k)V$ is performed in three stages:

1. $$(H - \lambda_j I - i\omega_k I)Y_j = -V \odot \Psi_j, \quad j = 1, 2, \ldots, n_s$$

2. $$W = 4\,\mathrm{Re}\left(\sum_{j=1}^{n_s} \Psi_j^* \odot Y_j\right)$$

3. $$\nabla^2(\mathcal{A}(\omega_k)V) = -4\pi W$$

Stage 1 is the most computationally intensive and requires solving n_s block linear systems.

Hamiltonian matrix (from real-space DFT)

$$\underbrace{(H - \lambda_j I - i\omega_k I)}_{A} Y_j = -V \odot \Psi_j, \quad j = 1, 2, \ldots, n_s$$

- $H = -\frac{1}{2}\nabla^2 + \chi\chi^H + \text{diag}\,(P_{\text{eff}})$
- H is real symmetric and indefinite; its lowest n_s eigenpairs (λ_j, Ψ_j) are known *a priori*
- A is not Hermitian but satisfies $A = A^T$ (i.e., it is *complex symmetric*)
- A becomes nearly singular as $\omega_k \to 0$
- We use the convention

$$\lambda_1 < \lambda_2 < \ldots < \lambda_{n_s}$$

$$0 < \omega_8 < \omega_2 < \ldots < \omega_1$$

A complex symmetric solver: preconditioned COCG

Algorithm 1 Block preconditioned COCG[a] for solving $AX = B$ satisfying $A = A^T$ and using a preconditioner M^{-1}

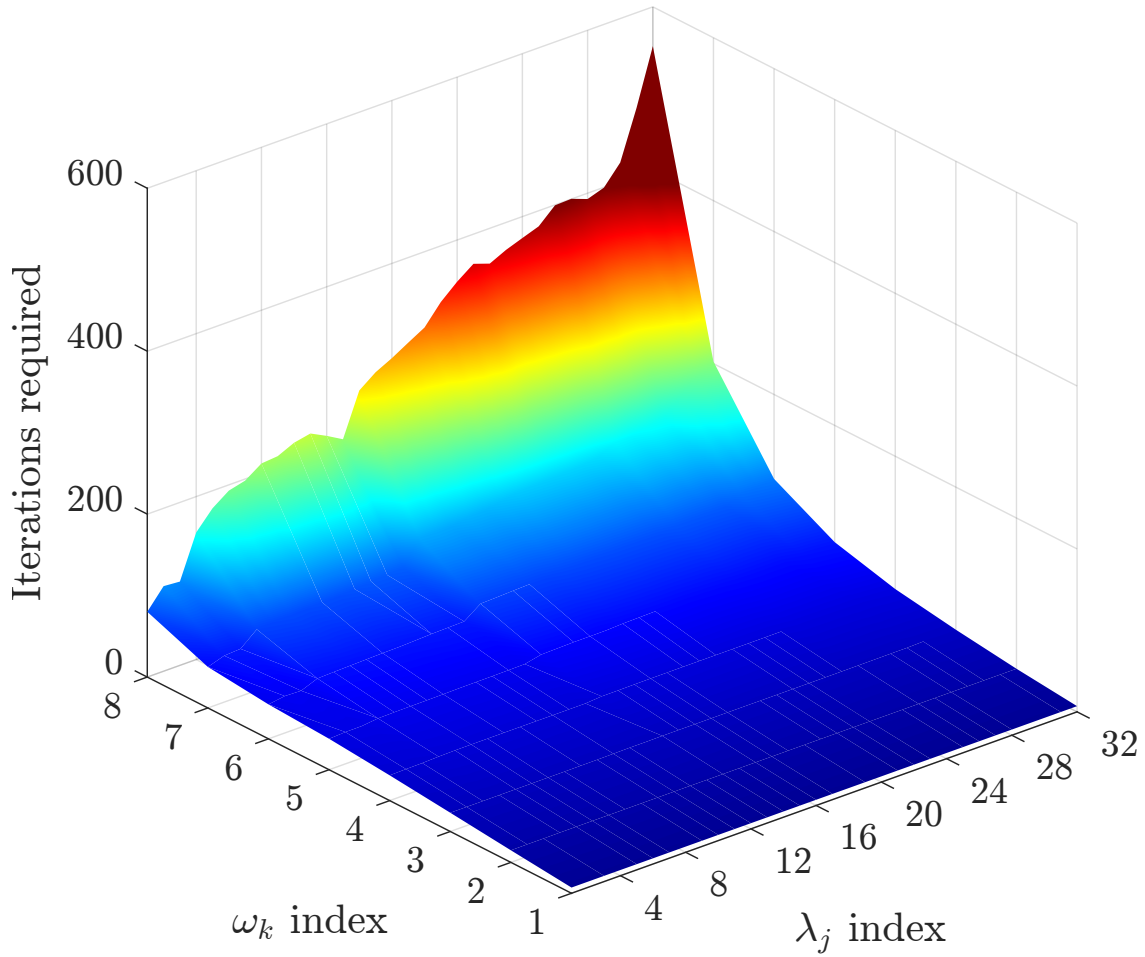
1: Initialize X_0 and $V_0 \leftarrow B - AX_0$
2: $W_0 \leftarrow M^{-1} V_0$
3: $\rho_0 \leftarrow V_0^T W_0$
4: $P_{-1} \leftarrow 0$, $\beta_{-1} \leftarrow 0$
5: **for** j $= 0, 1, 2, \ldots$ **do**
6: $\quad P_j \leftarrow V_j + P_{j-1}\beta_{j-1}$
7: $\quad U_j \leftarrow AP_j$
8: $\quad \mu_j \leftarrow U_j^T P_j$
9: $\quad \alpha_j \leftarrow \mu_j^{-1} \rho_j$
10: $\quad X_{j+1} \leftarrow X_j + P\alpha_j$
11: $\quad V_{j+1} \leftarrow V_j - U_j \alpha_j$
12: $\quad W_{j+1} \leftarrow M^{-1} V_{j+1}$
13: $\quad \rho_{j+1} \leftarrow V_{j+1}^T W_{j+1}$
14: $\quad \beta_j \leftarrow \rho_j^{-1} \rho_{j+1}$
15: **end for**

[a] van der Vorst and Melissen 1990.

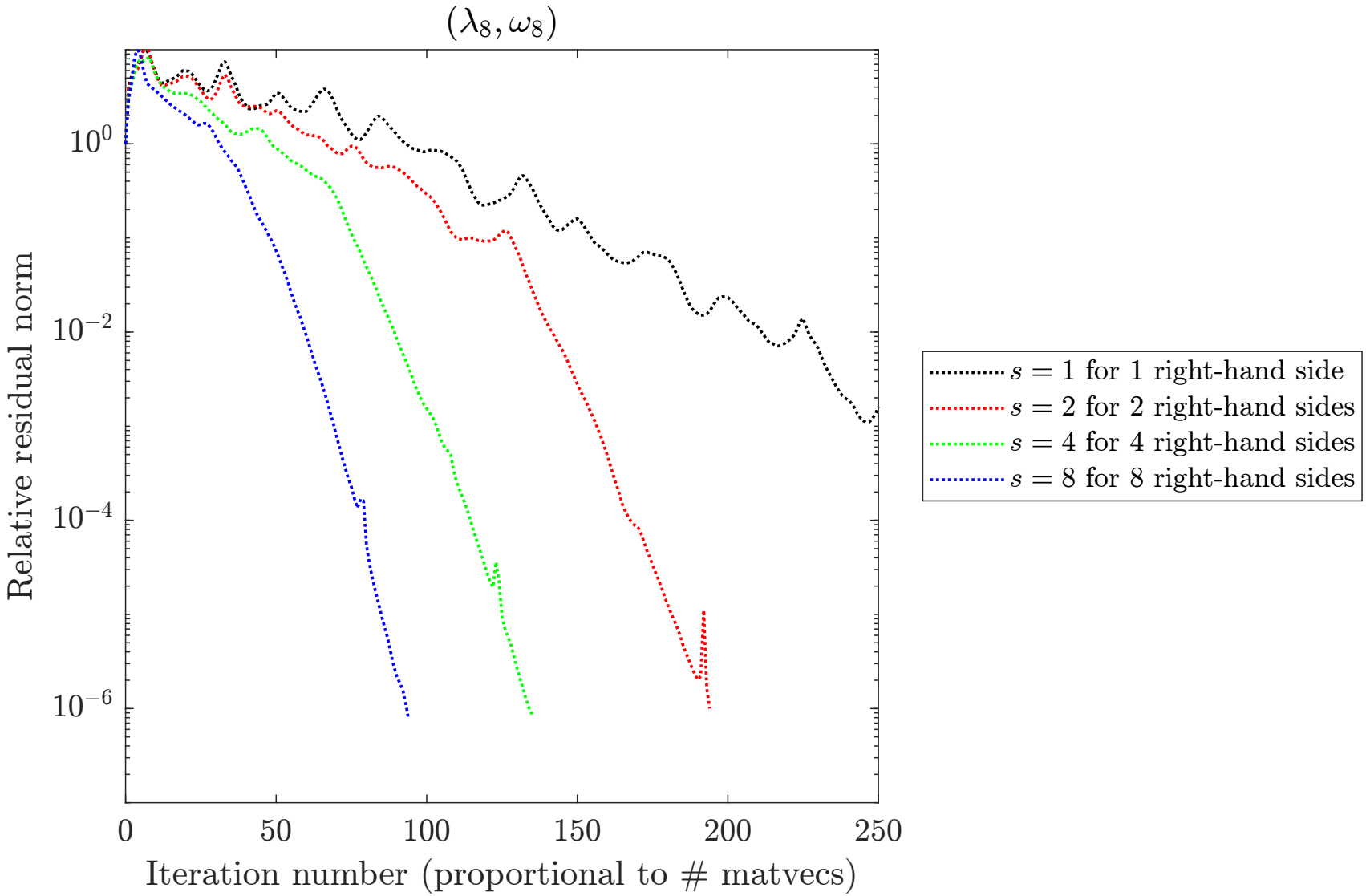
- COCG is not optimal in the residual norm but achieves similar convergence to GMRES in practice
- There is potential for breakdown in the iteration

The linear systems vary widely in difficulty

$$(H - \lambda_j I - i\omega_k I) Y_j = -V \odot \Psi_j$$



Should we use a block size greater than 1?

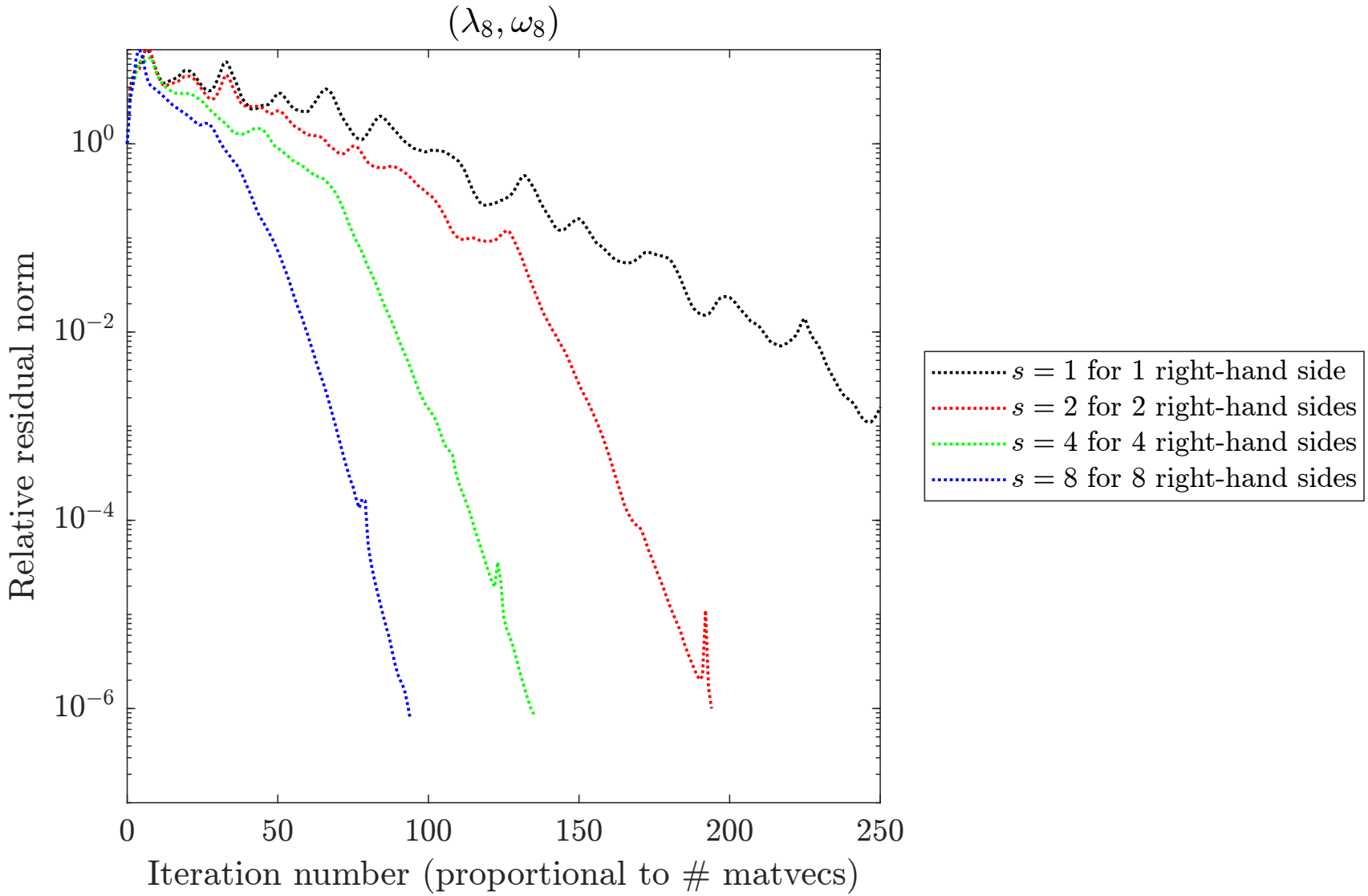


Constructing a good initial guess

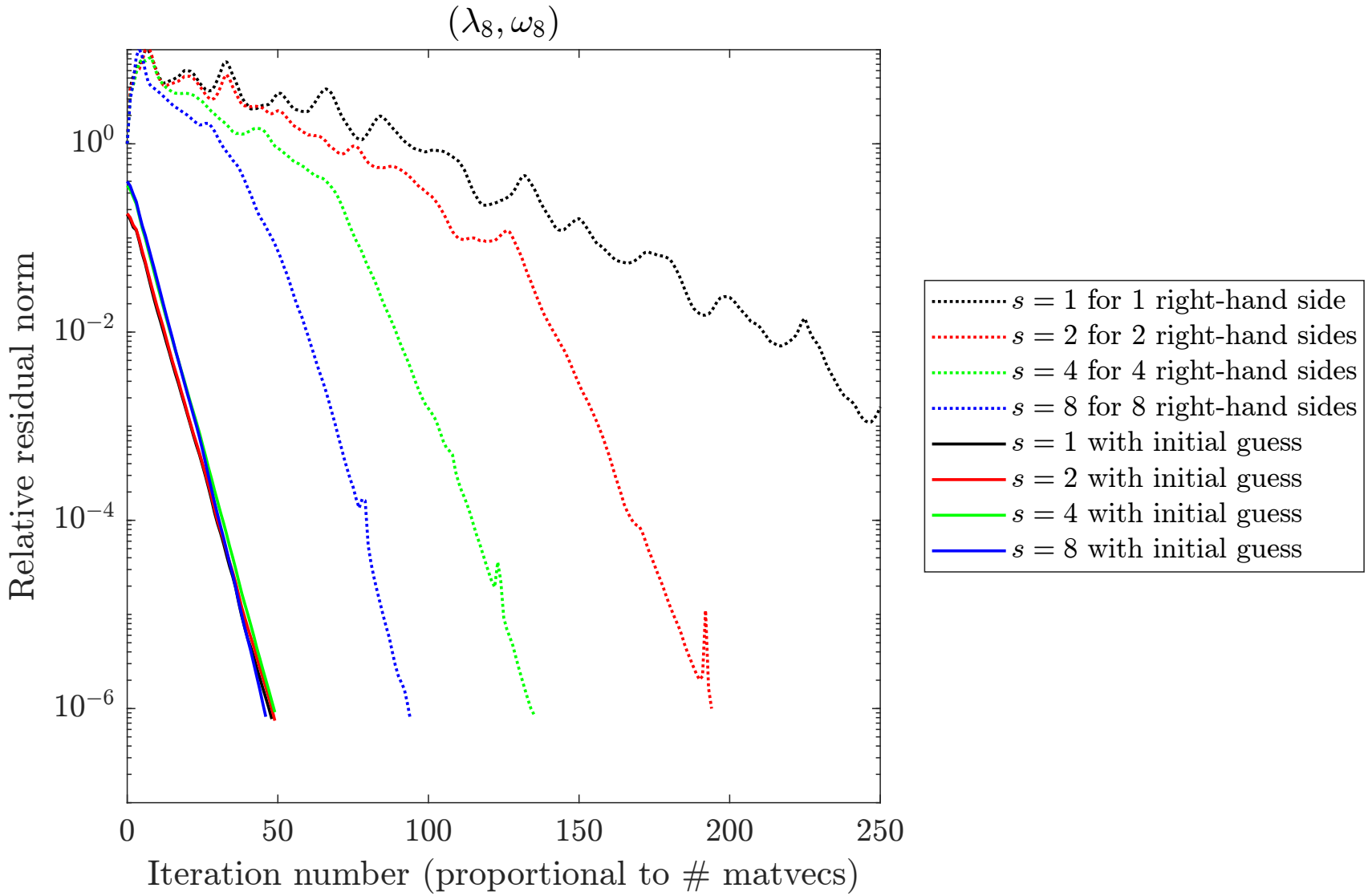
We know the lowest n_s eigenpairs $(\lambda_\ell, \Psi_\ell)$ of H; since H and A share eigenvectors, we can deflate an invariant subspace from the initial residual:

$$
\begin{aligned}
X_0 &= \boxed{\Psi(\Lambda - \lambda_j I - i\omega_k I)^{-1}\Psi^H B} \\
R_0 &= B - AX_0 \\
&= B - (H - \lambda_j I - i\omega_k I)\Psi(\Lambda - \lambda_j I - i\omega_k I)^{-1}\Psi^H B \\
&= B - \Psi\Psi^H B \\
\Psi^H R_0 &= \Psi^H B - \Psi^H \Psi\Psi^H B \\
&= 0
\end{aligned}
$$

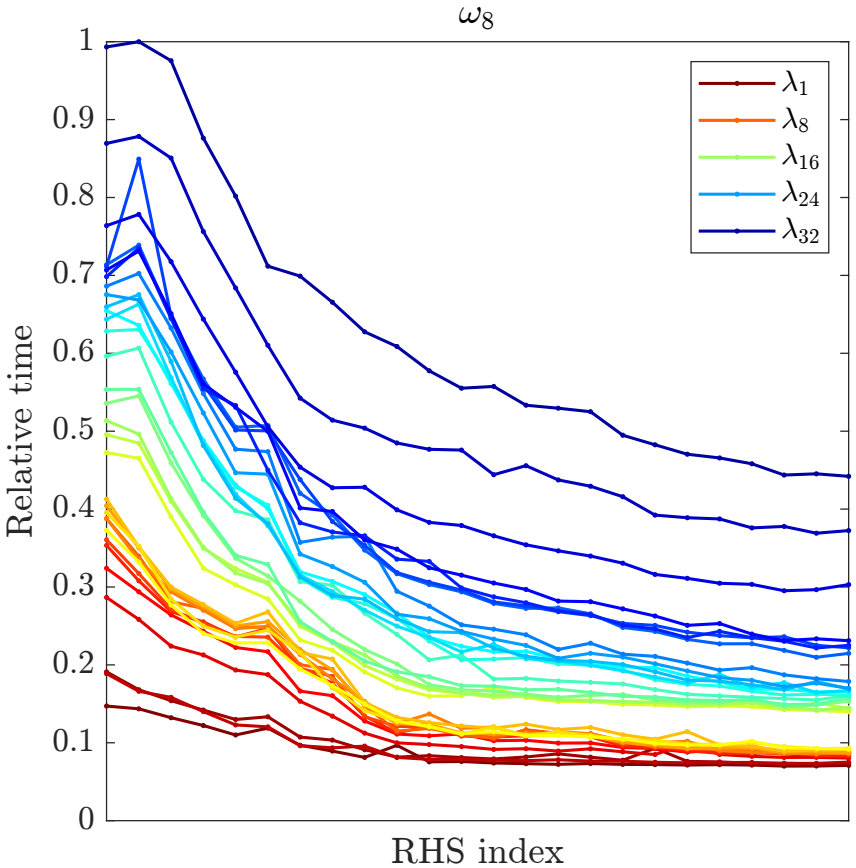
Should we use a block size greater than 1?

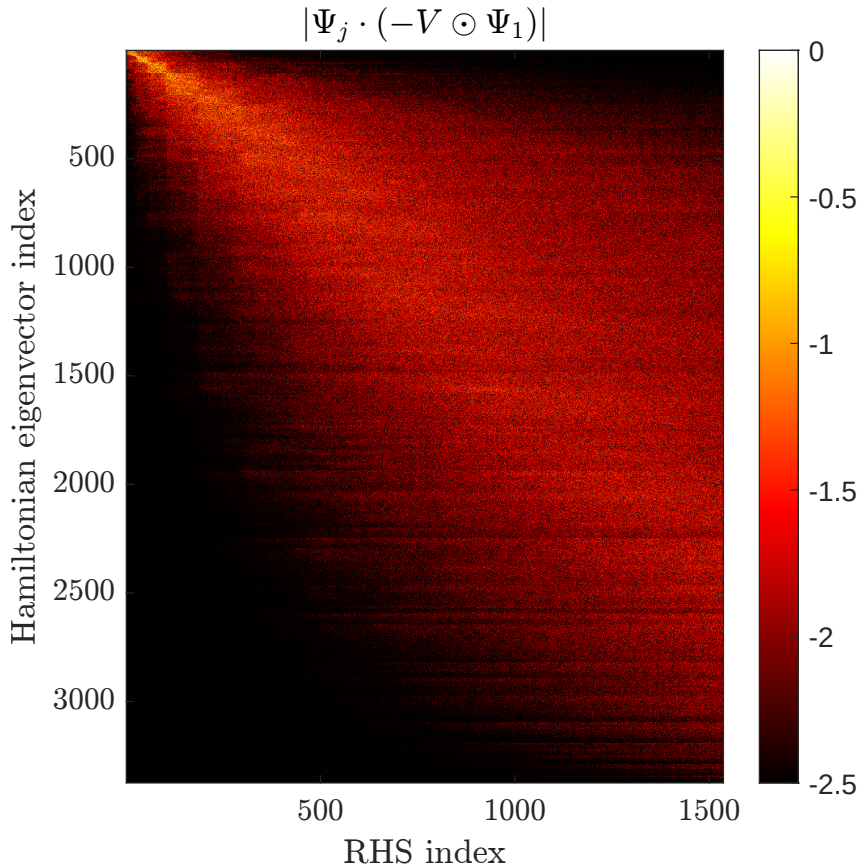


Should we use a block size greater than 1?



Iterations required is *also* RHS-dependent





In search of a preconditioner

We want a preconditioner that reduces the total time required to solve all of the Sternheimer equations. This preconditioner must either

be extremely cheap to apply

or

drastically accelerate convergence

or

ideally both!

Overview

1. A family of block linear systems
2. A method of efficient preconditioning
3. A pair of welcome surprises

Complex symmetric preconditioners

Assume $M = M^T$. It admits an LDU decomposition:

$$M = LDU \text{ and } M^T = U^T D^T L^T = U^T DL$$

By induction, we can show $U = L^T$ and therefore:

$$M = LDL^T = LD^{1/2}D^{1/2}L^T = (LD^{1/2})(LD^{1/2})^T = KK^T$$

The preconditioned matrix in COCG is $A_{\text{precond}} = K^{-1}AK^{-T}$, thus:

$$A_{\text{precond}}^T = K^{-1}A^T K^{-T} = A_{\text{precond}}$$

Choosing a good preconditioner

$$\left(-\frac{1}{2}\nabla^2 + \chi\chi^H + \text{diag}\left(P_{\text{eff}}\right) - \lambda_j I - i\omega_k I\right) Y_j = -V \odot \Psi_j$$

The natural choice of preconditioner is

$$M^{-1} = \left(-\frac{1}{2}\nabla^2 + \overline{P_{\text{eff}}} I - \lambda_j I - i\omega_k I\right)^{-1}$$

If we know the eigendecomposition $\nabla^2 = Q\Lambda Q^H$, M^{-1} simply becomes

$$M^{-1} = Q\left(-\frac{1}{2}\Lambda + (\overline{P_{\text{eff}}} - \lambda_j - i\omega_k)I\right)^{-1} Q^H$$

Kronecker application of $(\nabla^2)^{-1}$

$$\nabla^2 = (\mathcal{I}_z \otimes \mathcal{I}_y \otimes \mathcal{D}_x) + (\mathcal{I}_z \otimes \mathcal{D}_y \otimes \mathcal{I}_x) + (\mathcal{D}_z \otimes \mathcal{I}_y \otimes \mathcal{I}_x)$$

If $\mathcal{D}_x$, $\mathcal{D}_y$, and $\mathcal{D}_z$ have eigenvectors Q_x, Q_y, and Q_z, respectively, then:

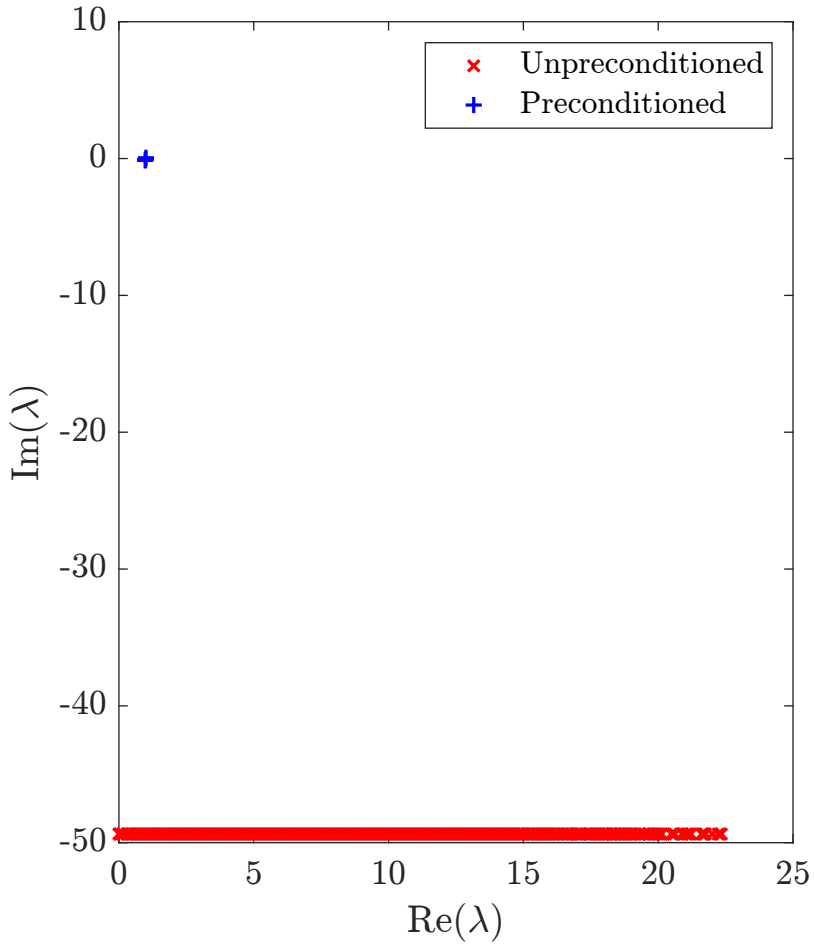
$$Q = Q_z \otimes Q_y \otimes Q_x \text{ and } Q^H = Q_z^H \otimes Q_y^H \otimes Q_x^H$$

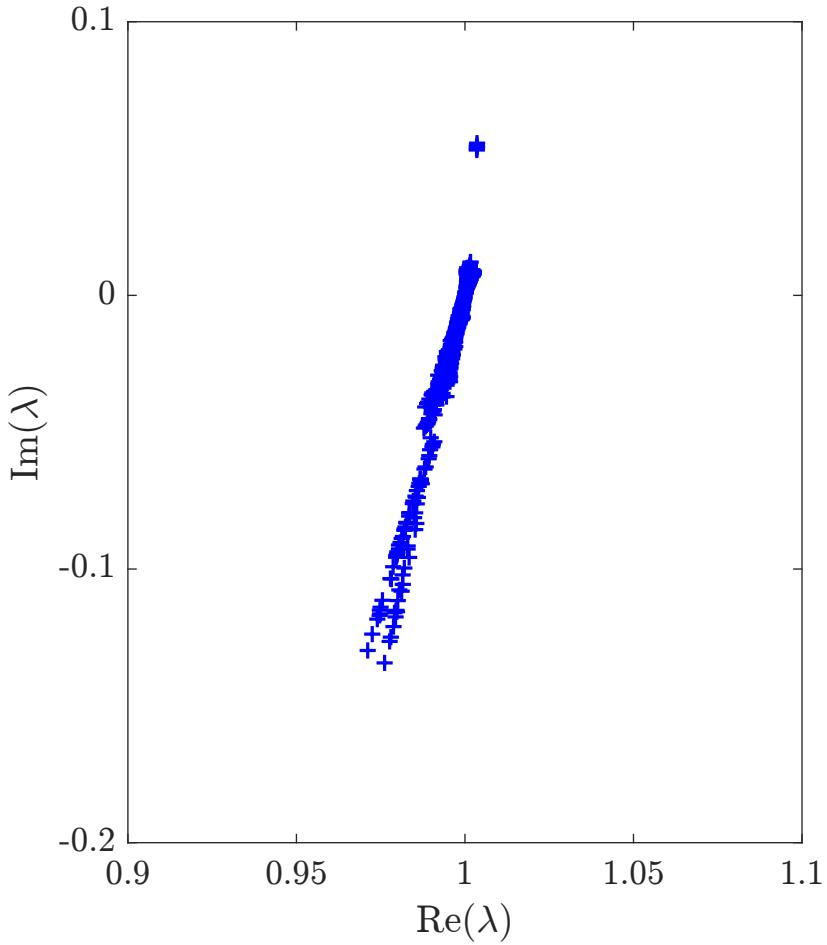
$$M^{-1}u = Q\Lambda_{M^{-1}}Q^H u = (Q_z \otimes Q_y \otimes Q_x)\Lambda_{M^{-1}}(Q_z^H \otimes Q_y^H \otimes Q_x^H)u$$

The product $(A \otimes B \otimes C)u$ can be computed efficiently *even when the finite difference directions are non-orthogonal*[1].

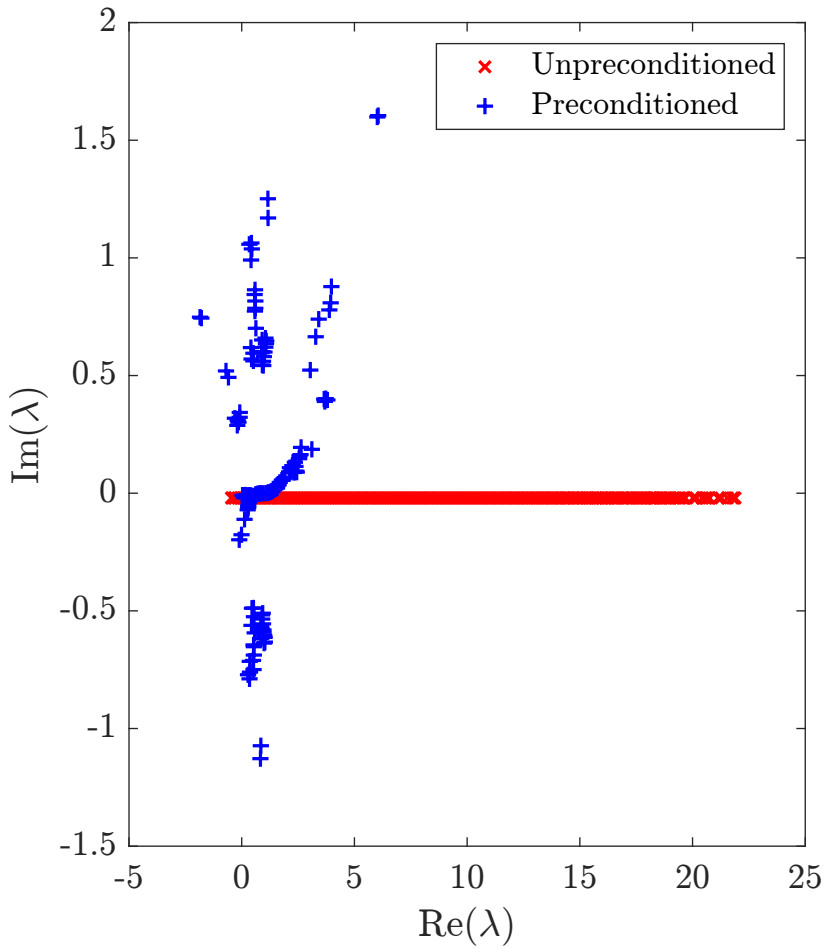
[1] Sharma and Suryanarayana 2018.

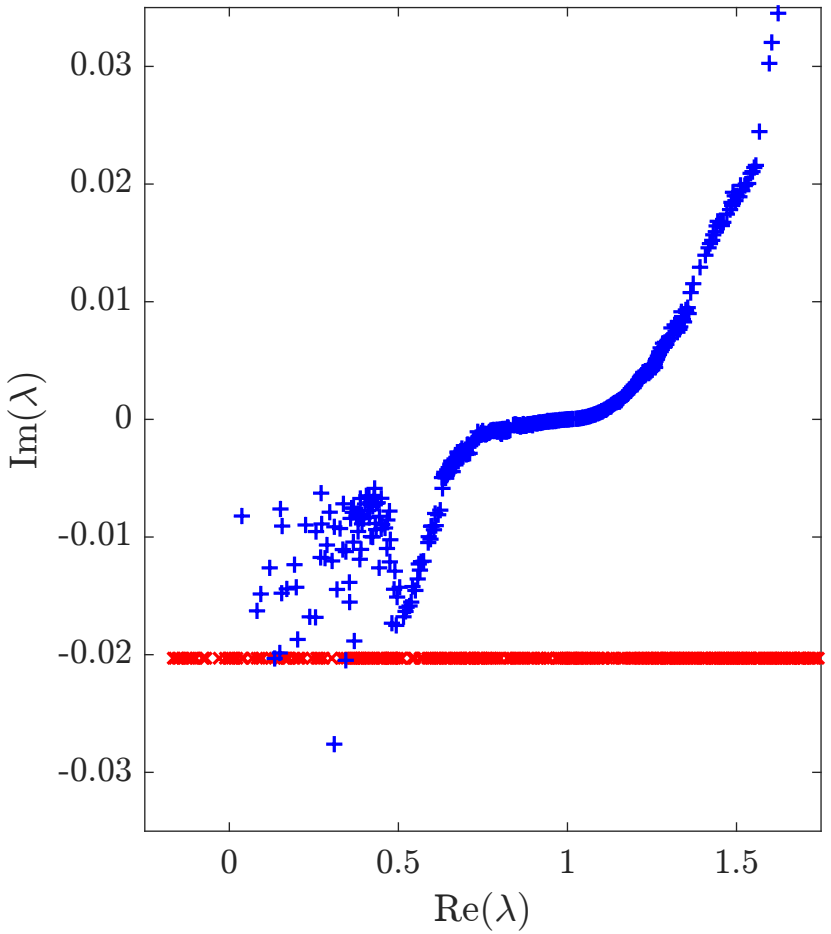
Spectral behavior: easy Sternheimer system (λ_1, ω_1)





Spectral behavior: hardest Sternheimer system (λ_{32}, ω_8)

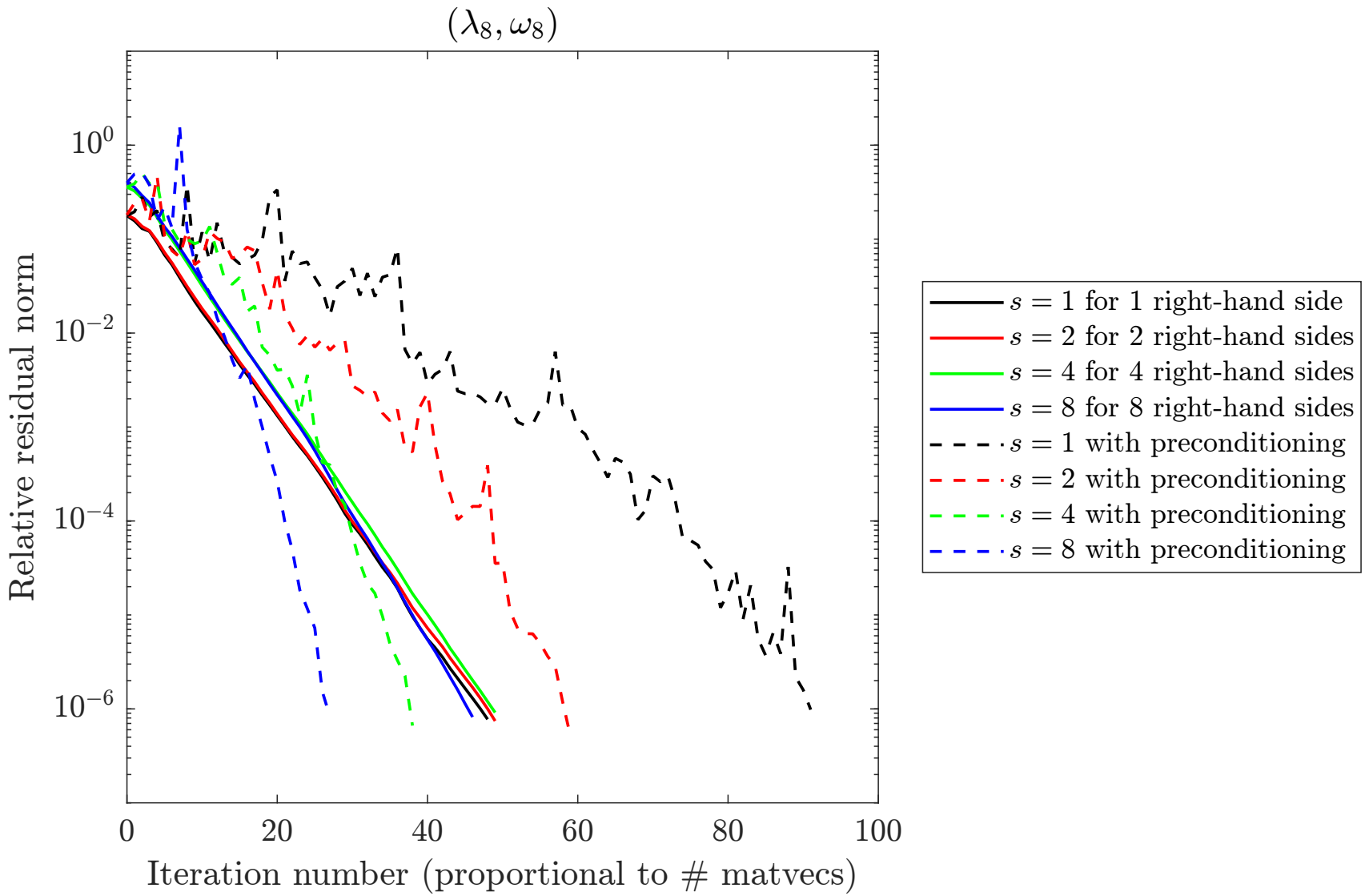




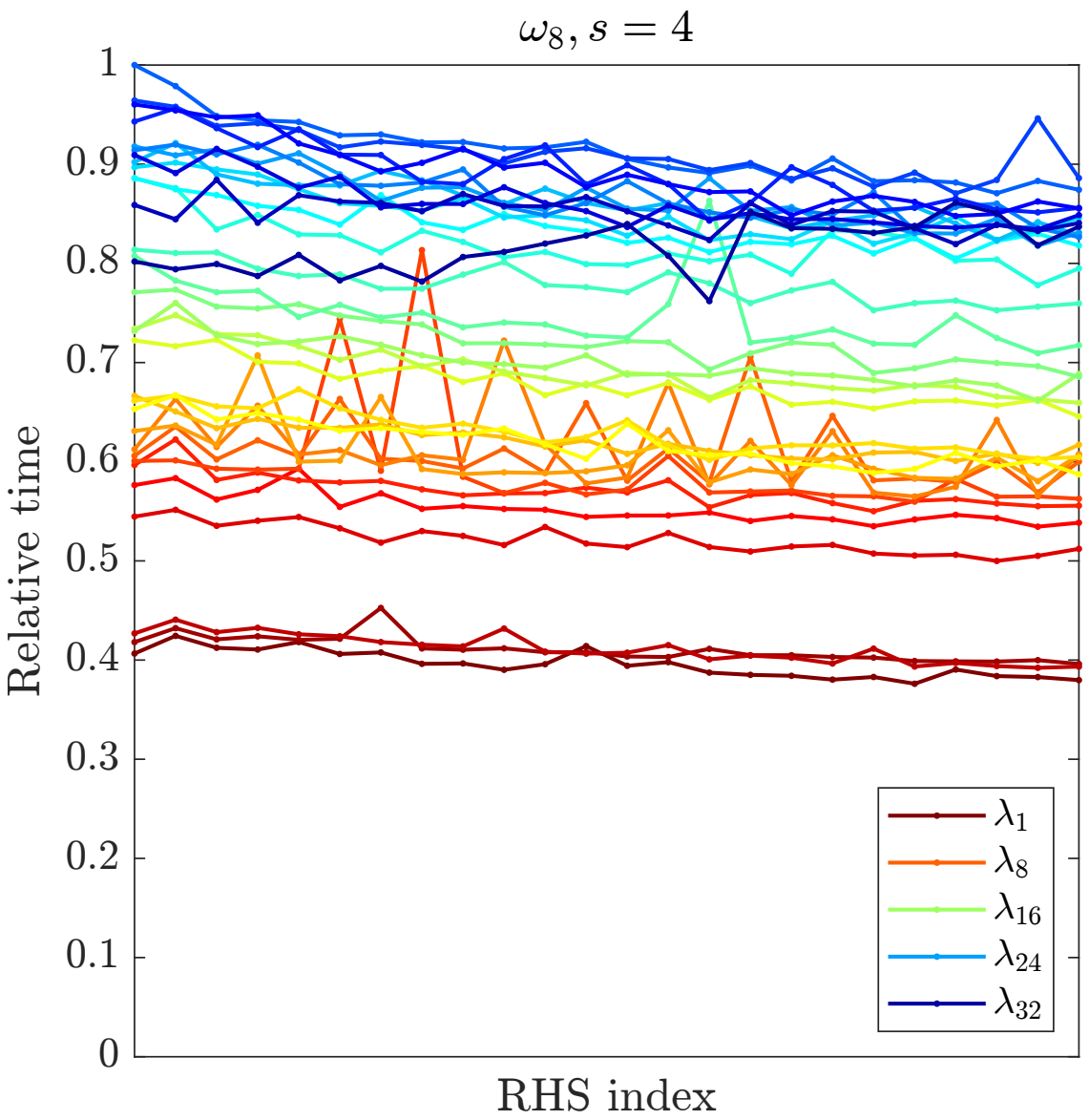
Overview

1. A family of block linear systems
2. A method of efficient preconditioning
3. A pair of welcome surprises

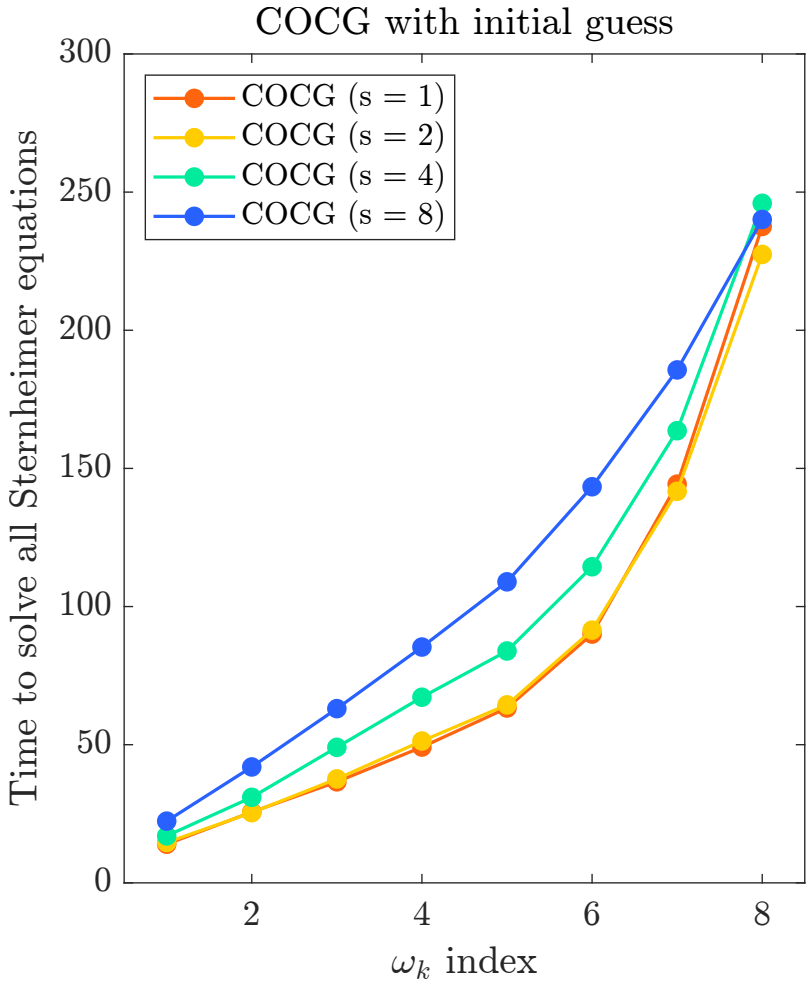
Block size dependence returns with a twist

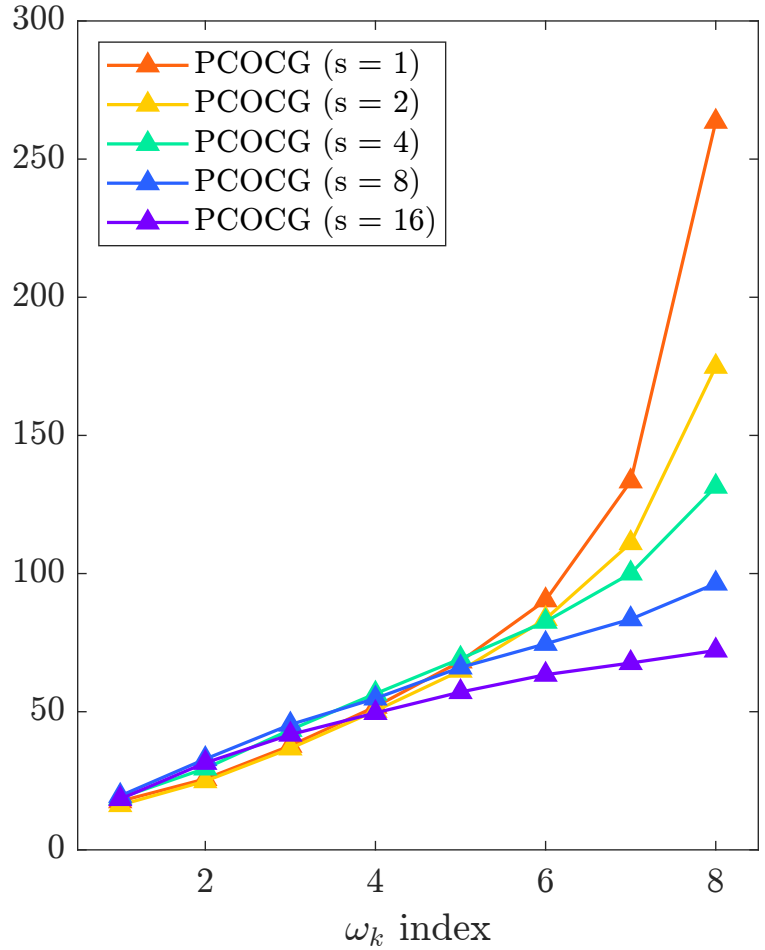


Uniform difficulty across the preconditioned systems

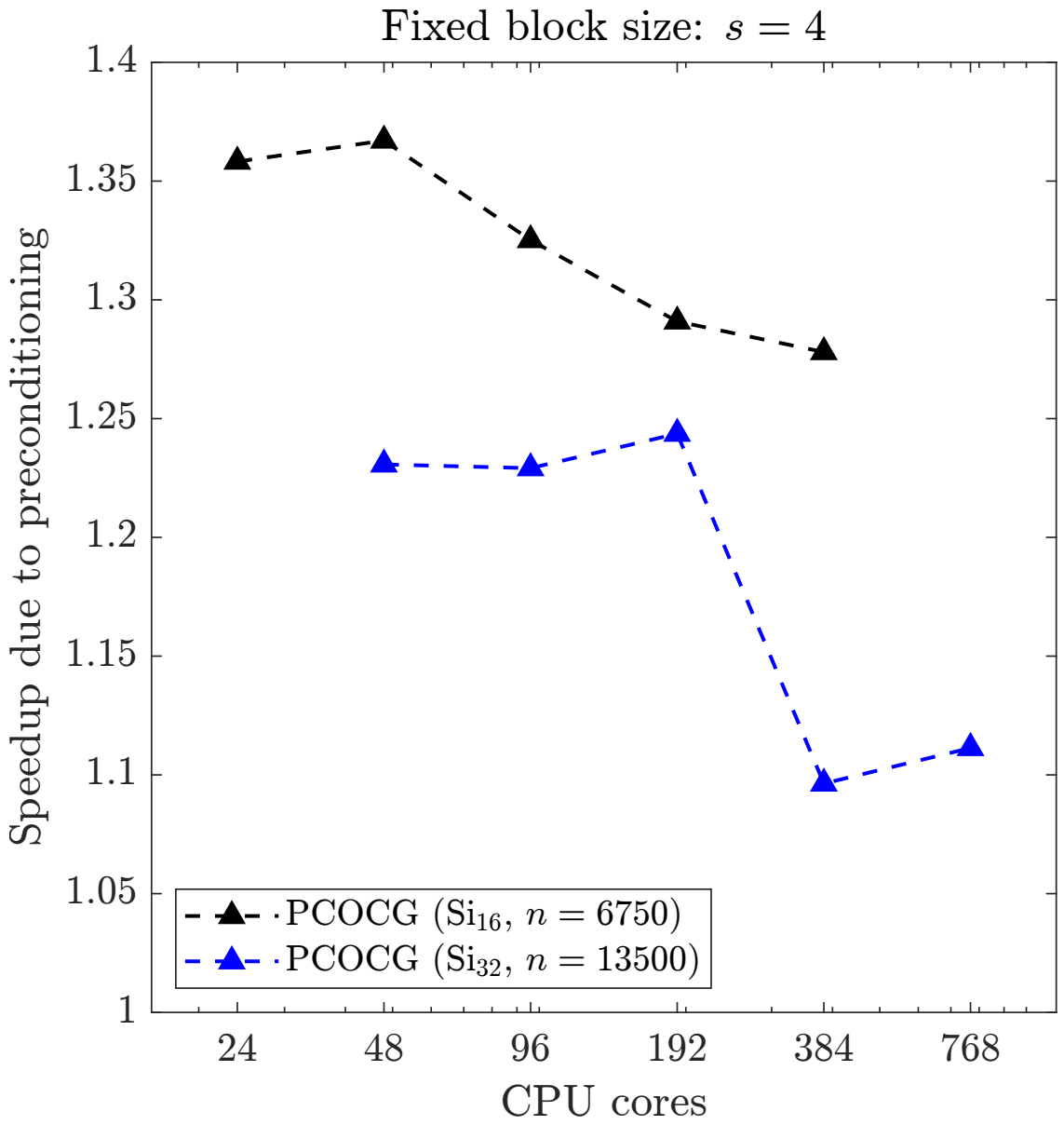


When should we use this preconditioner?





Modest speedup even before optimizing



Preconditioning and beyond

1. Without preconditioning, the initial guess is remarkably effective; however, it seems to make convergence independent of block size

Preconditioning and beyond

1. Without preconditioning, the initial guess is remarkably effective; however, it seems to make convergence independent of block size
2. The initial guess is mostly incompatible with preconditioning, although it helps a small amount in practice

Preconditioning and beyond

1. Without preconditioning, the initial guess is remarkably effective; however, it seems to make convergence independent of block size
2. The initial guess is mostly incompatible with preconditioning, although it helps a small amount in practice
3. Preconditioning reintroduces the block size dependence and is faster when large block sizes are used on the hardest linear systems

Preconditioning and beyond

1. Without preconditioning, the initial guess is remarkably effective; however, it seems to make convergence independent of block size
2. The initial guess is mostly incompatible with preconditioning, although it helps a small amount in practice
3. Preconditioning reintroduces the block size dependence and is faster when large block sizes are used on the hardest linear systems
4. Shockingly, preconditioning fixes the issue of load imbalance that cropped up due to our specific right-hand side vectors

Full comparison for (λ_8, ω_8)