Final project:

Low-rank correction for accelerating preconditioned iterative methods

Context

Objective

- Compute solution to linear system $Ax = b$
- $A \in \mathbb{R}^{n \times n}$ is ill conditioned

Preconditioned iterative method

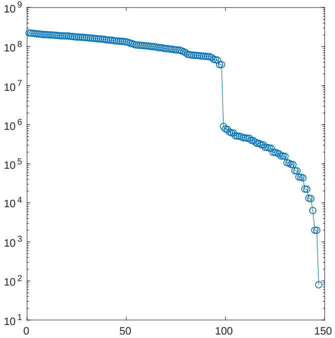
1. Compute preconditioner M^{-1} such that $M^{-1} \approx A^{-1}$, e.g.,
   - Low precision LU factorization
   - Incomplete LU factorization
   - Block Low-Rank LU factorization
2. Solve $Ax = b$ via some iterative method (e.g., GMRES) preconditioned by M^{-1}, e.g., with leeft-preconditioning, $M^{-1}Ax = M^{-1}b$

- Convergence to solution may be slow or fail

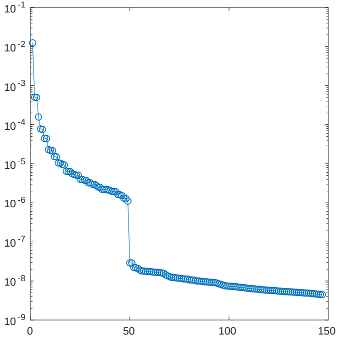
$\Rightarrow$ **Objective: accelerate convergence**

Key observation

Matrix lund_a ($n = 147$, $\kappa(A) = 2.8e{+}06$)



SVD of A



SVD of A^{-1}

- Often, A is ill conditioned due to a small number of small singular values
- Then, A^{-1} is numerically low-rank

Key idea

Factorization error might be low-rank?

Assume $M = A + \Delta A$ and consider the error

$$E = M^{-1}A - I = M^{-1}(M + \Delta A) - I$$
$$= M^{-1}\Delta A \approx A^{-1}\Delta A$$

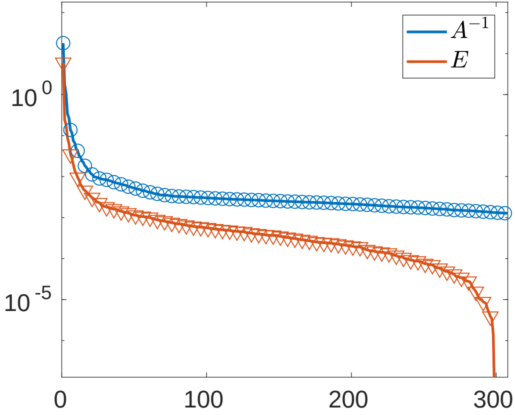
Does E retain the low-rank property of A^{-1}?

A novel preconditioner

Consider the preconditioner $M_k = M(I + E_k)$ with E_k a rank-k approximation to E.

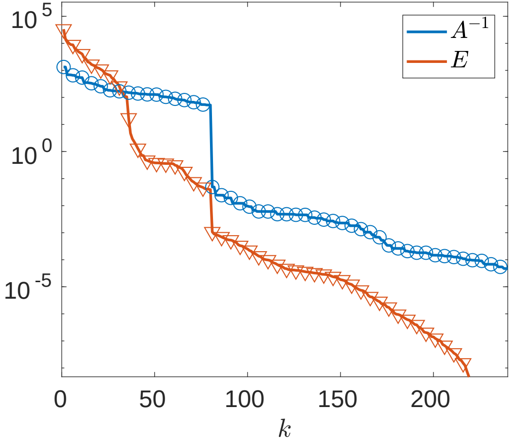
- If $E = E_k$, $M_k = A$
- If $E \approx E_k$ for some small k, M_k^{-1} can be computed cheaply via Sherman-Morrison-Woodbury formula

Typical SV distributions of A^{-1} and E



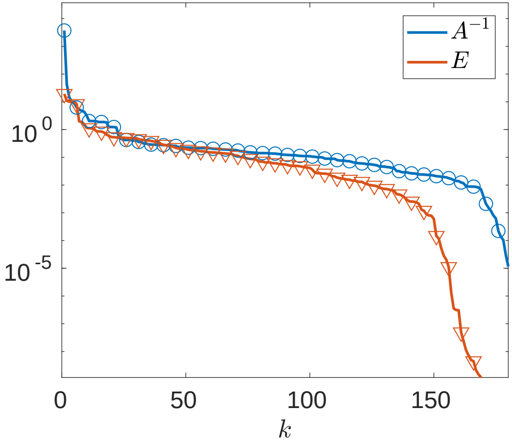
Matrix cz308

Typical SV distributions of A^{-1} and E



Matrix steam1

Typical SV distributions of A^{-1} and E



Matrix rajat14

Suggestions

- Gather some test matrices for which A^{-1} is numerically low-rank (you can generate them randomly, or take a look at Suitesparse collection for real-life problems)
- Prepare a reference solver (suggestion: use MATLAB's `gmres`) and some reference preconditioners M (e.g., MATLAB's `ilu`, or low precision `lu`)[1] (Lecture 9)
- If you use sparse matrices, remember Lecture 6 and look up MATLAB's reordering tools (e.g., `dissect`)
- How to compute a rank-k approximation of E ? Explicitly forming E is not a good idea! You should rather use a method that only requires matrix–vector multiplies...
- Perform some numerical experiments and test the role of k (or ε), etc.
- Should one build a fixed-rank (k) or fixed-accuracy (ε) LRA of E?
- Should one use left or right preconditioning? (note that M_k is defined differently in either case)
- Can refer to Higham and M. (2019) for some guidance

[1] Either using MATLAB's `single` or simulating low precision by computing `lu`$(A + \Delta A)$ for a random perturbation ΔA