Lecture 24
Preconditioning

Songting Luo

Department of Mathematics
Iowa State University

MATH 562 Numerical Analysis II

Outline

1. Preconditioning

Outline

1. Preconditioning

Preconditioning

- Motivation: Convergence of iterative methods heavily depends on eigenvalues or singular values of equation
- Main idea of preconditioning is to introduce a nonsingular matrix M such that $\mathbf{M}^{-1}\mathbf{A}$ has better properties than $\mathbf{A}$. Thereafter, solve

$$\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$$

  , which has the same solution as $\mathbf{A}\mathbf{x} = \mathbf{b}$
- Criteria of $\mathbf{M}$
  - "Good" approximation of $\mathbf{A}$, depending on iterative solvers
  - Ease of inversion
- Typically, a preconditioner $\mathbf{M}$ is good if $\mathbf{M}^{-1}\mathbf{A}$ is not too far from normal and its eigenvalues are clustered

Left, Right, and Hermitian Preconditioners

- Left preconditioner: Left multiply $\mathbf{M}^{-1}$ and solve $\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$
- Right preconditioner: Right multiply $\mathbf{M}^{-1}$ and solve $\mathbf{A}\mathbf{M}^{-1}\mathbf{y} = \mathbf{b}$ with $\mathbf{x} = \mathbf{M}^{-1}\mathbf{y}$.
- However, if $\mathbf{A}$ is Hermitian, $\mathbf{M}^{-1}\mathbf{A}$ or $\mathbf{A}\mathbf{M}^{-1}$ breaks symmetry
- How to resolve this problem?

Left, Right, and Hermitian Preconditioners

- Left preconditioner: Left multiply $\mathbf{M}^{-1}$ and solve $\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$
- Right preconditioner: Right multiply $\mathbf{M}^{-1}$ and solve $\mathbf{A}\mathbf{M}^{-1}\mathbf{y} = \mathbf{b}$ with $\mathbf{x} = \mathbf{M}^{-1}\mathbf{y}$.
- However, if $\mathbf{A}$ is Hermitian, $\mathbf{M}^{-1}\mathbf{A}$ or $\mathbf{A}\mathbf{M}^{-1}$ breaks symmetry
- How to resolve this problem?
- Suppose $\mathbf{M}$ is Hermitian positive definite, with $\mathbf{M} = \mathbf{C}\mathbf{C}^*$ for some $\mathbf{C}$, then $\mathbf{A}\mathbf{x} = \mathbf{b}$ is equivalent to

$$[\mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-*}]\mathbf{C}^*\mathbf{x} = \mathbf{C}^{-1}\mathbf{b},$$

  where $\mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-*}$ is Hermitian positive definite, and it is similar to $\mathbf{C}^{-*}\mathbf{C}^{-1}\mathbf{A} = \mathbf{M}^{-1}\mathbf{A}$ and has the same eigenvalues as $\mathbf{M}^{-1}\mathbf{A}$
- Example of $M = \mathbf{C}\mathbf{C}^*$ is Cholesky factorization $\mathbf{M} = \mathbf{R}\mathbf{R}^*$, where $\mathbf{R}$ is upper triangular.

Preconditioned Conjugate Gradient

- When preconditioning a symmetric matrix, use SPD matrix $\mathbf{M}$, and $\mathbf{M} = \mathbf{R}\mathbf{R}^T$
- In practice, algorithm can be organized so that only $\mathbf{M}^{-1}$ (instead of $\mathbf{R}^{-1}$) appears

Algorithm: Preconditioned Conjugate Gradient Method

$\mathbf{x}_0 = \mathbf{0},\ \mathbf{r}_0 = \mathbf{b},\ \mathbf{p}_0 = \mathbf{M}^{-1}\mathbf{r}_0,\ \mathbf{z}_0 = \mathbf{p}_0$

for $n = 1$ to $1, 2, 3, \ldots$

$\alpha_n = (\mathbf{r}_{n-1}^T \mathbf{z}_{n-1}) / (\mathbf{p}_{n-1}^T \mathbf{A}\mathbf{p}_{n-1})$	step length
$\mathbf{x}_n = \mathbf{x}_{n-1} + \alpha_n \mathbf{p}_{n-1}$	approximate solution
$\mathbf{r}_n = \mathbf{r}_{n-1} - \alpha_n \mathbf{A}\mathbf{p}_{n-1}$	residual
$\mathbf{z}_n = \mathbf{M}^{-1}\mathbf{r}_n$	preconditioning
$\beta_n = (\mathbf{r}_n^T \mathbf{z}_n) / (\mathbf{r}_{n-1}^T \mathbf{z}_{n-1})$	improvement this step
$\mathbf{p}_n = \mathbf{z}_n + \beta_n \mathbf{p}_{n-1}$	search direction

Commonly Used Preconditioners

- Jacobi preconditioning: $\mathbf{M} = diag(\mathbf{A})$. Very simple and cheap, might improve certain problems but usually insufficient
- Block-Jacobi preconditioning: Let $\mathbf{M}$ be composed of block-diagonal instead of diagonal.
- Classical iterative methods: Precondition by applying one step of Jacobi, Gauss-Seidel, SOR, or SSOR
- Incomplete factorizations: Perform Gaussian elimination or Cholesky factorization but ignore fill
- Multigrid (coarse-grid approximations): For a PDE discretized on a grid, a preconditioner can be formed by transferring the solution to a coarser grid, solving a smaller problem, then transferring back. This is sometimes the most efficient approach if applicable