Welcome to Ma 221! Lecture 28, Fall 24

Preconditioning to accelerate iterative Methods for $Ax=b$

How to change $Ax=b$ cheaply to converge faster:

simplest: $M^{-1}Ax = M^{-1}b$ where $M \approx A$

(1) M^{-1} should be cheap to multiply by

(2) $M^{-1}A$ better conditioned than A

Idea straightforward for GMRES, which works for any general A

Apply GMRES to $M^{-1}Ax = M^{-1}b$

Trickier for CG: M s.p.d. and A s.p.d.

then $M^{-1}A$ generally not

M s.p.d $\Rightarrow M = Q\Lambda Q^T \Rightarrow M^{1/2} = Q\Lambda^{1/2}Q^T$

Imagine applying CG to

$$(*) \qquad \underbrace{(M^{-1/2} A M^{-1/2})}_{\text{s.p.d}}(M^{1/2}x) = M^{-1/2}b$$

$M^{-1}A$ and $M^{-1/2}AM^{-1/2}$ are similar

$M^{\frac{1}{2}}(M^{-1}A)M^{-\frac{1}{2}} = M^{-\frac{1}{2}}AM^{-\frac{1}{2}}$

$\Rightarrow$ same evals and

$\text{cond}(M^{-1/2}AM^{1/2}) \leq \text{cond}(M^{-1}A)$

Preconditioned CG (equivalent to CG on (*))

$k=0$, $x(0)$, $r(0)=b$, $p(1)=M^{-1}b$, $y(0)=M^{-1}r(0)$

repeat

$k=k+1$

$z=Ap(k)$

$\nu(k)=(y(k-1)^T r(k-1))/(p(k)^T z)$

$x(k)=x(k-1)+\nu(k)p(k)$

$r(k)=r(k-1)-\nu(k)\cdot z$

$y(k)=M^{-1}r(k)$

$\mu(k+1)=(y(k)^T r(k))/(y(k-1)^T r(k-1))$

$p(k+1)=y(k)+\mu(k+1)\cdot p(k)$

until $\|r(k)\|_2$ small enough

Thm 6.9: equivalent to CG on (*)

(HW Q 6.14) prove it!

(see errata on page 317)

Where to get a good M?

goals: (1) cheap to multiply by M^{-1}

(2) cond $(M^{-1}A) \ll$ cond (A)

lots of preconditioners, depends on A

(1) A s.p.d $M=\text{diag}(A_{11}, A_{22}, \ldots, A_{nn})$

"Jacobi preconditioning"

$A=M-(M-A) = M-K$

$R_j=M^{-1}K=M^{-1}(M-A)=I-M^{-1}A$

$M^{-1}A=I-R_j$

$\rho(R_j) < 1 \iff M^{-1}A$ well conditioned

Thm (Van der Sluis, 1969) if M diagonal then $\mathrm{diag}(A_{11}, A_{22}, \ldots, A_{nn})$ within factor of n of optimal, i.e. minimizing $\mathrm{cond}(M^{-1}A)$

(2) $M = \mathrm{diag}(\tilde{A}_{11}, \tilde{A}_{22} \ldots, \tilde{A}_{kk})$

$$A = \begin{bmatrix} \tilde{A}_{11} & & \\ & \tilde{A}_{22} & \\ & & \tilde{A}_{33} \end{bmatrix}$$

where each $\tilde{A}_{ii}$ is a square block on diagonal; more expensive than Jacobi preconditioning (called block Jacobi)

Van der Sluis generalizes: If M can be any block diagonal matrix, choosing as above within factor k of minimal condition # of $M^{-1}A$

(D., 2023)

(3) "Incomplete Cholesky"

compute a partial Cholesky decomp. $LL^T \approx A$ by limiting fill-in of L

simplest: no new nonzeroes in L

$$L_{ij} \neq 0 \Rightarrow A_{ij} \neq 0$$

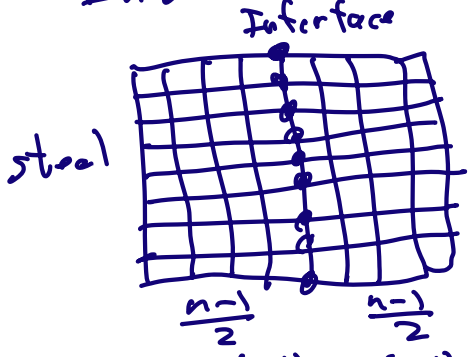
$\Rightarrow$ triangular solves with L, L^T costs

same # flops as multiplying by A

(4) a few steps of multigrid
(perhaps just applied to $\tilde{A}_{ii}$)

(5) Domain Decomposition (sec 6.10 of text)

Ex: discretize FE problem on 2D mesh
Left is discretization of steel
Right " " " concrete
Interface in between



$$A = \begin{bmatrix} A_{ss} & 0 & A_{si} \\ 0 & A_{cc} & A_{ci} \\ A_{is} & A_{ic} & A_{ii} \end{bmatrix} \quad \begin{matrix} \frac{n(n-1)}{2} & \frac{n(n-1)}{2} & n \end{matrix}$$

$$A = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ A_{is}A_{ss}^{-1} & A_{ic}A_{cc}^{-1} & I \end{bmatrix} \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & S \end{bmatrix} \begin{bmatrix} A_{ss} & 0 & A_{si} \\ 0 & A_{cc} & A_{ci} \\ 0 & 0 & I \end{bmatrix}$$

S = Schur complement

$$= A_{ii} - A_{is}A_{ss}^{-1}A_{si} - A_{ic}A_{cc}^{-1}A_{ci}$$

$$A^{-1} = \begin{bmatrix} A_{ss}^{-1} & 0 & -A_{ss}^{-1}A_{si} \\ 0 & A_{cc}^{-1} & -A_{cc}^{-1}A_{ci} \\ 0 & 0 & I \end{bmatrix} \cdot \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & S^{-1} \end{bmatrix} \cdot \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ -A_{is}A_{ss}^{-1} & -A_{ic}A_{cc}^{-1} & I \end{bmatrix}$$

How to cheaply multiply by A^{-1} (approximately)

Good idea if have good preconditioners for A_{ss} and A_{cc}

Need to multiply by

$(-A_{is}A_{ss}^{-1})x = -A_{is}(A_{ss}^{-1}x)$

$\Rightarrow$ use preconditioner for A_{ss} to approximate $A_{ss}^{-1}x$

ditto for $-A_{ic}A_{cc}^{-1}x$

Multiplying by S^{-1} (approximately)

$Sx = (A_{ii}x) - A_{is}(A_{ss}^{-1}(A_{si}x)) - A_{ic}(A_{cc}^{-1}(A_{ci}x))$

use preconditioners for A_{ss} and A_{cc}

Given Sx use CG to multiply by S^{-1}, using all preceding tricks

Often S much better conditioned than A,

Thm. $\text{cond}(S) \le \text{cond}(A)$ if A s.p.d.

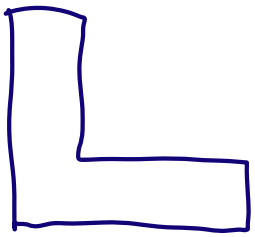
why often much better?

smaller: dimension n vs $\frac{n(n-1)}{2}$

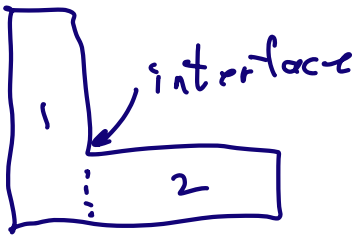
Poisson: $\kappa(S) \approx \sqrt{\kappa(A)}$

Called "non over lapping domain decomposition"

Also: "over lapping domain decomp"

Ex: 

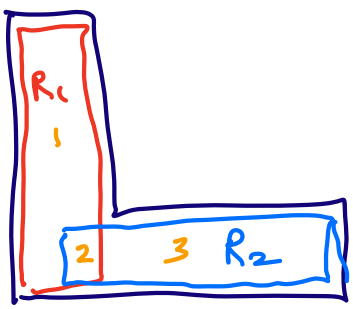
solve Poisson on L-shaped domain

Non overlapping 

use fast Poisson solver on each rectangle

disadvantage: data moves slowly across interface (same problem that multigrid addressed)

Solution: let domains overlap



→

$$\begin{bmatrix} A_{11} & A_{12} & \\ A_{21} & A_{22} & A_{23} \\ & A_{32} & A_{33} \end{bmatrix}$$

$$R_1 = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$ for rectangle 1

$$R_2 = \begin{bmatrix} A_{22} & A_{23} \\ A_{32} & A_{33} \end{bmatrix}$$ for rectangle 2

$$M^{-1} = \begin{bmatrix} R_1^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & R_2^{-1} \end{bmatrix}$$

"overlapping block Jacobi"
"additive Schwartz"

Apply idea from Gauss-Seidel to use most recent data:
Solve with R_1^{-1}, then use updated solution to solve with R_2^{-1}

"multiplicative Schwartz"