MATH 676

–

Finite element methods in scientific computing

Wolfgang Bangerth, Texas A&M University

Lecture 35:

What preconditioner to use

Introduction

Parts 1+2: Simple preconditioners for simple problems

Observations on iterative solvers

The finite element method provides us with a linear system

$$A\,x \;=\; b$$

that we then need to solve.

Basic observations:

- For sparse direct solvers, speed of solution only depends on sparsity pattern
- For iterative solvers, performance also depends on the *values* in A
- Performance measures:
  – number of iterations
  – cost of every iteration

Observations on iterative solvers

The finite element method provides us with a linear system

$$A\,x \;=\; b$$

that we then need to solve.

Factors affecting performance of iterative solvers:

- Symmetry of a matrix
- Whether A is definite
- Condition number of A
- How the eigenvalues of A are clustered

- Whether A is reducible/irreducible

Observations on iterative solvers

Example 1: Using CG to solve

$$A\,x \;=\; b$$

where A is SPD, each iteration reduces the residual by a factor of

$$r \;=\; \frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1} \;<\; 1$$

- For a tolerance ε we need $n \;=\; \frac{\log \epsilon}{\log r}$ iterations
- For the Laplace matrix, this is $r \;=\; \frac{c-h}{c+h} \;<\; 1$

$$n \;=\; O\left(\frac{\log \epsilon}{h}\right)$$

Observations on iterative solvers

Example 2: When solving

$$A\,x \;=\; b$$

where A has the form

$$A \;=\; \begin{pmatrix} a_{11} & 0 & 0 & \cdots \\ 0 & a_{22} & 0 & \cdots \\ 0 & 0 & a_{33} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

then every decent iterative solver converges in 1 iteration.

Note 1: This, even though condition number may be large

Note 2: This is true, in particular, if $A=I$.

The idea of preconditioners

Idea: When solving

$$A\,x \;=\; b$$

maybe we can find a matrix P^{-1} and instead solve

$$P^{-1}A\,x \;=\; P^{-1}b$$

Observation 1: If $P^{-1}A \sim D$ then solving should require less iterations

Corollary: The perfect preconditioner is a multiple of the inverse matrix, i.e., $P^{-1}=A^{-1}$.

The idea of preconditioners

Idea: When solving

$$A\,x \;=\; b$$

maybe we can find a matrix P^{-1} and instead solve

$$P^{-1}A\,x \;=\; P^{-1}b$$

Observation 2: Iterative solvers only need matrix-vector multiplications, no element-by-element access.

Corollary: It is sufficient if P^{-1} is just an operator (see step-20 as an example).

The idea of preconditioners

Idea: When solving

$$A\,x \;=\; b$$

maybe we can find a matrix P^{-1} and instead solve

$$P^{-1}A\,x \;=\; P^{-1}b$$

Observation 3: There is a tradeoff:
fewer iterations vs *cost of preconditioner.*

Corollary: Preconditioning only works if P^{-1} is cheap to compute and if P^{-1} is cheap to *apply* to a vector.

Example: $P^{-1}=A^{-1}$ does not qualify.

The idea of preconditioners

Notes on the following lectures:

- For quantitative analysis, one typically needs to consider the *spectrum* of operators and preconditioners

- Here, the goal is simply to get an “intuition” on how preconditioners work

Constructing preconditioners

Part 1:
Constructing simple preconditioners for elliptic problems

(1952 – 1980s)

Constructing preconditioners

Remember: When solving the preconditioned system

$$P^{-1} A x = P^{-1} b$$

then the best preconditioner is $P^{-1}=A^{-1}$.

Problem: (i) We can't compute it efficiently. (ii) If we could, we would not need an iterative solver.

But: Maybe we can approximate $P^{-1}\sim A^{-1}$.

Idea 1: Do we know of other iterative solution techniques?

Idea 2: Use incomplete decompositions.

Constructing preconditioners

Approach 1: Remember the oldest iterative techniques!

To solve $Ax = b$ we can use *defect correction*:

- Under certain conditions, the iteration:

$$x^{(k+1)} = x^{(k)} - P^{-1}\left(A x^{(k)} - b\right)$$

  will converge to the exact solution x

- Unlike Krylov-space methods, convergence is linear
- The best preconditioner is again $P^{-1} \sim A^{-1}$

Constructing preconditioners

Approach 1: Remember the oldest iterative techniques!

Preconditioned defect correction for $Ax = b$, $A = L+D+U$:

- Jacobi iteration:

$$x^{(k+1)} = x^{(k)} - \omega D^{-1}\left(A x^{(k)} - b\right)$$

- The Jacobi preconditioner is then

$$P^{-1} = \omega D^{-1}$$

  which is easy to compute and apply.

Note: We don't need the scaling ("relaxation") factor.

Constructing preconditioners

Approach 1: Remember the oldest iterative techniques!

Preconditioned defect correction for $Ax = b$, $A = L+D+U$:

- Gauss-Seidel iteration:

$$x^{(k+1)} = x^{(k)} - \omega \left(L+D\right)^{-1} \left(A x^{(k)} - b\right)$$

- The Gauss-Seidel preconditioner is then

$$P^{-1} = \omega \left(L+D\right)^{-1} \qquad \text{i.e. } h = P^{-1} r \text{ solves } \left(L+D\right) h = \omega r$$

  which is easy to compute and apply as $L+D$ is triangular.

Note 1: We don't need the scaling (“relaxation”) factor.
Note 2: This preconditioner is not symmetric.

Constructing preconditioners

Approach 1: Remember the oldest iterative techniques!

Preconditioned defect correction for $Ax = b$, $A = L+D+U$:

- SOR (Successive Over-Relaxation) iteration:

$$x^{(k+1)} = x^{(k)} - \omega \left(D+\omega L\right)^{-1} \left(A x^{(k)} - b\right)$$

- The SOR preconditioner is then

$$P^{-1} = \left(D+\omega L\right)^{-1}$$

Note 1: This preconditioner is not symmetric.
Note 2: We again don't care about the constant factor in P.

Constructing preconditioners

Approach 1: Remember the oldest iterative techniques!

Preconditioned defect correction for $Ax = b$, $A = L+D+U$:

- SSOR (Symmetric Successive Over-Relaxation) iteration:

$$x^{(k+1)} = x^{(k)} - \frac{1}{\omega(2-\omega)}(D+\omega U)^{-1} D (D+\omega L)^{-1}(A x^{(k)} - b)$$

- The SSOR preconditioner is then

$$P^{-1} = (D+\omega U)^{-1} D (D+\omega L)^{-1}$$

Note: This preconditioner is now symmetric if *A* is symmetric!

Constructing preconditioners

Approach 1: Remember the oldest iterative techniques!

Common observations about preconditioners from stationary iterations:

- Have been around for a long time
- Generally useful for small problems (<100,000 DoFs)
- Not particularly useful for larger problems

Constructing preconditioners

Approach 2: Approximations to A^{-1}

Idea 1: Incomplete decompositions

- Incomplete LU (ILU):
  Perform an LU decomposition on A but only keep elements of L, U that fit into the sparsity pattern of A

- Incomplete Cholesky (IC):
  LL^T decomposition if A is symmetric

- Many variants:
  – strengthen diagonal
  – augment sparsity pattern
  – thresholding of small/large elements

Constructing preconditioners

Approach 2: Approximations to A^{-1}

Idea 2: Sparse Approximate Inverse (SPAI)

- Find a matrix P^{-1} so that

$$\|A\,P^{-1} - I\|_F \;\rightarrow\; \min$$

- Yields independent least-squares problems for each row
- Can be restricted to matrices P^{-1} that are also sparse

Constructing preconditioners

Part 2:
Constructing simple preconditioners
for other, simple problems

(1952 – 1990s)

Elliptic problems

Case 1: Symmetric elliptic problems:

- Diffusion-reaction equation:

$$-\mu \Delta u + c u = f$$

- Biharmonic equation:

$$\mu \Delta^2 u = f$$

- Linear elasticity:

$$-\lambda \nabla (\nabla \cdot \vec{u}) - \mu \nabla \cdot (\nabla \vec{u} + \nabla \vec{u}^T) = \vec{f}$$

- Laplace with spatially variable coefficients

$$-\nabla \cdot a(x) \nabla u = f$$

These all lead to symmetric, positive definite matrices for which the same preconditioners work as for the Laplace eq.

Non-elliptic, scalar problems

Case 2: Scalar non-symmetric problems:

- Advection-diffusion equation:

$$-\mu \Delta u + \beta \cdot \nabla u = f$$

- Pure advection equation (transport equation):

$$\beta \cdot \nabla u = f$$

- Reaction terms:

$$-\mu \Delta u + \beta \cdot \nabla u + c u = f$$

- Biharmonic equation:

$$\Delta^2 u = f$$

Non-elliptic, scalar problems

Scalar non-symmetric problems:

- Defect correction-based preconditioners typically continue to work
- Quality often deteriorates with deviation from standard Laplace matrix
- ILU is a robust choice
- Biharmonic equation requires special care:
  – with appropriate elements leads to SPD matrix that can be treated like the Laplace matrix
  – often converted into mixed (vector-valued) problem

Summary

Preconditioners for “simple” problem:

- Defect correction-based preconditioners are the simplest choice
- Limited by slow speed of information propagation
- ILU/IC frequently better but more complex and limited by memory requirements/CPU time
- Both kinds work reasonably well for “small” problems
- For elliptic problems we today have much better methods: *Geometric* and *Algebraic Multigrid (GMG, AMG)*

MATH 676

–

Finite element methods in scientific computing

Wolfgang Bangerth, Texas A&M University