

Lecture 18 Classical Iterative Methods

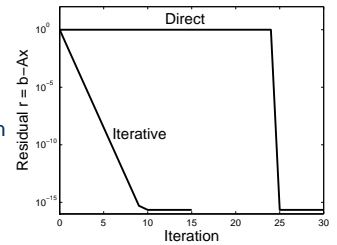
MIT 18.335J / 6.337J
Introduction to Numerical Methods

Per-Olof Persson
November 14, 2006

1

Iterative Methods for Linear Systems

- *Direct methods* for solving $Ax = b$, e.g. Gaussian elimination, compute an exact solution after a finite number of steps (in exact arithmetic)
- *Iterative algorithms* produce a sequence of approximations $x^{(1)}, x^{(2)}, \dots$ which hopefully converges to the solution, and
 - may require less memory than direct methods
 - may be faster than direct methods
 - may handle special structures (such as sparsity) in a simpler way



2

Two Classes of Iterative Methods

- *Stationary methods* (or classical iterative methods) finds a splitting $A = M - K$ and iterates $x^{(k+1)} = M^{-1}(Kx^{(k)} + b) = Rx^{(k)} + c$
 - Jacobi, Gauss-Seidel, Successive Overrelaxation (SOR), and Symmetric Successive Overrelaxation (SSOR)
- *Krylov subspace methods* use only multiplication by A (and possibly by A^T) and find solutions in the Krylov subspace $\{b, Ab, A^2b, \dots, A^{k-1}b\}$
 - Conjugate Gradient (CG), Generalized Minimal Residual (GMRES), BiConjugate Gradient (BiCG), etc

3

The Model Poisson Problem

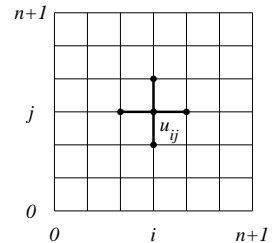
- Test problem for linear solvers: Discretize Poisson's equation in 2-D:

$$-\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = 1$$

on a square grid using centered finite difference approximations:

$$4u_{ij} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1} = h^2$$

- Dirichlet conditions $u = 0$ on boundaries
- Grid spacing $h = 1/(n+1)$
- Total of n^2 unknowns u_{ij}
- A "typical problem" despite the simplicity



4

The Model Problem in MATLAB

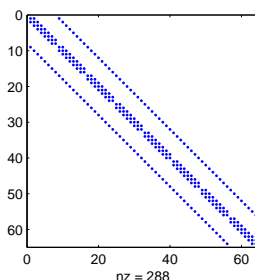
- In MATLAB:


```
n=8; h=1/(n+1); e=ones(n,1);
A1=spdiags([-e,2*e,-e],[-1:1,n,n]);
A=kron(A1,speye(n,n))+kron(speye(n,n),A1);
f=h^2*ones(n^2,1);
```

or simply

```
A=delsq(numgrid('S',n+2));
```

- Resulting linear system $Au = f$ is sparse and banded



5

Eigenvalues of Model Problem

- A is symmetric positive definite, with eigenvalues $\lambda_{ij} = \lambda_i + \lambda_j$, $i, j = 1, \dots, n$, where

$$\lambda_k = 2 \left(1 - \cos \frac{\pi k}{n+1}\right)$$

are eigenvalues of the 1-D Laplace operator

- Largest eigenvalue $\lambda_n = 2(1 - \cos \frac{\pi n}{n+1}) \approx 4$
- Smallest eigenvalue $\lambda_1 = 2(1 - \cos \frac{\pi}{n+1}) \approx \pi^2/(n+1)^2$
- Condition number $\kappa(A) = \lambda_n/\lambda_1 \approx 4(n+1)^2/\pi^2$

6

Stationary Iterative Methods

- Iterative methods for $Ax = b$ that can be written

$$x^{(k+1)} = Rx^{(k)} + c$$

with constant R are called *stationary* iterative methods

- A *splitting* of A is a decomposition $A = M - K$ with nonsingular M
- Stationary iterative method from splitting:

$$Ax = Mx - Kx = b \implies x = M^{-1}Kx + M^{-1}b = Rx + c$$

- The iteration $x^{(k+1)} = Rx^{(k)} + c$ converges to the solution $x = A^{-1}b$ if and only if the spectral radius $\rho(R) < 1$
- *Proof*. Blackboard

7

Choosing a Splitting

- Find a splitting $A = M - K$ such that
 - $Rx = M^{-1}Kx$ and $c = M^{-1}b$ are easy to evaluate
 - $\rho(R)$ is small
- Example: $M = I$ makes M^{-1} trivial, but probably not $\rho(R)$ small
- Example: $M = A$ gives $K = 0$ and $\rho(R) = \rho(M^{-1}K) = 0$, but expensive M^{-1}
- We will study splittings based on the diagonal and the upper/lower triangular parts of A

$$A = D - L - U$$

8

The Jacobi Method

- The simplest splitting is the *Jacobi method*, where $M = D$ and $K = L + U$:

$$x^{(k+1)} = D^{-1}((L + U)x^{(k)} + b)$$

- In words: Solve for x_i from equation i , assuming the other entries fixed
- Implementation of model problem: Trivial in MATLAB, but temporary array required with for-loops. The following code:

for $i = 1$ to n

for $j = 1$ to n

$$u_{i,j}^{(k+1)} = (u_{i-1,j}^{(k)} + u_{i+1,j}^{(k)} + u_{i,j-1}^{(k)} + u_{i,j+1}^{(k)} + h^2)/4$$

performs one step of the Jacobi method

9

Convergence of the Jacobi Method

- The following results can be shown: The Jacobi method K converges if
 - A is strictly row diagonally dominant: $|a_{ii}| > \sum_{j \neq i} |a_{ij}|$, or
 - A is weakly row diagonally dominant: $|a_{ii}| \geq \sum_{j \neq i} |a_{ij}|$ with strict inequality at least once, and A is *irreducible* (strongly connected graph)
- Our model problem is weakly row diagonally dominant and irreducible, so the Jacobi method converges
- More specifically, the splitting is $A = D - L - U = 4I - (4I - A)$, so $R_J = (4I)^{-1}(4I - A) = I - A/4$ with eigenvalues $1 - \lambda_{ij}/4$, and

$$\rho(R_J) = \max_{i,j} |1 - \lambda_{ij}/4| = |1 - \lambda_{11}/4| = \cos \frac{\pi}{n+1} \approx 1 - \frac{\pi^2}{2(n+1)^2}$$
- Therefore, it converges with a constant factor every $O(n^2)$ iteration

10

The Gauss-Seidel Method

- In the *Gauss-Seidel method*, we choose $M = D - L$ (which is triangular and therefore easy to invert) and iterate:

$$x^{(k+1)} = (D - L)^{-1}(Ux^{(k)} + b)$$

- In words: While looping over the equations, use the most recent values x_i
- Implementation of model problem: Still easy in MATLAB using `\`, and for-loops are actually easier than Jacobi since no temporary array:

for $i = 1$ to n

for $j = 1$ to n

$$u_{i,j}^{(k+1)} = (u_{i-1,j}^{(k+1)} + u_{i+1,j}^{(k)} + u_{i,j-1}^{(k+1)} + u_{i,j+1}^{(k)} + h^2)/4$$

- The order matters! For Cartesian grids, the *red-black ordering* updates in a checkerboard pattern

11

The Successive Overrelaxation Method

- In the *Successive overrelaxation method*, or SOR, the Gauss-Seidel step is extrapolated a factor ω :

$$x^{(k+1)} = \omega \tilde{x}_i^{(k+1)} + (1 - \omega)x_i^{(k)}$$

where \tilde{x} is the Gauss-Seidel iterate

- $\omega = 1$: Gauss-Seidel, $\omega > 1$: *overrelaxation*, $\omega < 1$: *underrelaxation*
- In matrix form:

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

12

Convergence of Gauss-Seidel and SOR

- It can be shown that with a symmetric positive definite matrix A , Gauss-Seidel and SOR converges with $0 < \omega < 2$
- In general hard to choose ω for SOR, but if spectral radius of the Jacobi method $\rho(R_J)$ is known, the optimal $\omega = 2 / \left(1 + \sqrt{1 - \rho(R_J)}\right)$
- For the model problem with red-black ordering:
 - Gauss-Seidel is twice as fast as Jacobi
 - For SOR, the optimal $\omega = 2 / \left(1 + \sin \frac{\pi}{n+1}\right)$, giving a spectral radius

$$\rho(R_{\text{SOR}}) \approx 1 - \frac{2\pi}{n+1}$$

which is n times faster than Jacobi/Gauss-Seidel, or a constant factor improvement every $O(n)$ iteration

13

The Symmetric Successive Overrelaxation Method

- To obtain an iteration matrix similar to a symmetric matrix, apply two SOR steps in opposite directions:

$$x^{(k+1)} = B_1 B_2 x^{(k)} + \omega(2 - \omega)(D - \omega U)^{-1} D (D - \omega L)^{-1} b,$$

$$B_1 = (D - \omega U)^{-1} (\omega L + (1 - \omega)D)$$

$$B_2 = (D - \omega L)^{-1} (\omega U + (1 - \omega)D)$$

- This *Symmetric successive overrelaxation method*, or SSOR, is useful as preconditioner for symmetric matrices
- By itself not very different from two steps of SOR

14