DN2222
Applied Numerical Methods
- part 2:
Numerical Linear Algebra

Lecture 7
Iterative methods
(Krylov Subspace Methods)

2012-11-21

Static Iterative Methods

- $P - Q = A$
- $Ax = b$
- $Px - Qx = b$
- $Px = Qx + b$
- $x = P^{-1}Qx + P^{-1}b$
- $x = Mx + c$
- $\implies x^{(m+1)} = Mx^{(m)} + c$, Convergence if $||M|| < 1$.

Static Method - Jacobi

- $P = diag(A)$
- Very simple to calculate.
- P^{-1} very simple to obtain.
- Convergence guaranteed if A is diagonally dominant (but can be very slow!)
- Converence is linear (of course).

Static Method - Gauss-Seidel

- $P = L(A) + diag(A)$
- Very simple to obtain.
- P^{-1} quite simple to obtain (triangular matrix).
- Convergence guaranteed if A is diagonally dominant (but can be very slow!)
- Faster than Jacobi, since immediately uses updated values.

Krylov Subspace Methods

- Are used both for **solving** $Ax = b$ and finding **eigenvalues** of A.
- Can be used when A is a "black box".
- Especially good when A is sparse (i.e. computing Ax requires few operations).
- Can be used even if A is not available, (eg Ax is available only as result of calling a routine).

- A variety of different Krylov subspace methods exist. Some are suitable for nonsymmetric matrices, and others assume symmetry and/or SPD.
- The different methods optimize different things, like $||x_k - x^*||$ or $||r_k|| = ||b - Ax_k||$.
- Most famous method: Conjugate Gradient Method.

Conjugate Gradient Method - CG

- Start with $x_0 = 0$, giving $r_0 = b$ and search direction $p_1 = b$.
- FOR $k = 1, 2, \ldots$

  $z = Ap_k$
  $v_k = (r_{k-1}^T r_{k-1})/(p_k^T z)$
  $x_k = x_{k-1} + v_k p_k$
  $r_k = r_{k-1} - v_k z$
  $\mu_{k+1} = (r_k^T r_k)/(r_{k-1}^T r_{k-1})$
  $p_{k+1} = r_k + \mu_{k+1} p_k$
- END

Preconditioning

- If we find a matrix M that is a good approximation of A, but is easy to invert (or rather easy to solve $My = c$) we can solve the preconditioned system.

$$M^{-1}Ax = M^{-1}b$$

- If M is very simple/crude, not much improvement is made.
- If M is very close to A, convergence is fast, however each round is very cumbersome.

- The simplest choice for M is the diagonal of A.
- Another popular choice is Incomplete Cholesky factorization.

$$A = LL^T + R = M + R$$

Preconditioned Conjugate Gradient Method - PCG

- Start with $x_0 = 0$, giving $r_0 = b$ and search direction $p_1 = M^{-1}b$. $y_0 = M^{-1}r_0$.
- FOR $k = 1, 2, \ldots$

  $z = Ap_k$
  $v_k = (y_{k-1}^T r_{k-1})/(p_k^T z)$
  $x_k = x_{k-1} + v_k p_k$
  $r_k = r_{k-1} - v_k z$
  $y_k = M^{-1} r_k$
  $\mu_{k+1} = (y_k^T r_k)/(y_{k-1}^T r_{k-1})$
  $p_{k+1} = y_k + \mu_{k+1} p_k$

- END

CG Iterative solutions

x_0	x_1	x_2	x_3	x_4	x_5
0.00000	1.83333	3.33333	4.50000	5.33333	5.83333
0.00000	3.66667	6.66667	9.00000	10.66667	10.66667
0.00000	5.50000	10.00000	13.50000	13.50000	13.50000
0.00000	7.33333	13.33333	13.33333	13.33333	13.33333
0.00000	9.16667	9.16667	9.16667	9.16667	9.16667

Solving $Ax = b$

- We seek the "best" approximative solution! How define "best"?

- By minimizing $||x_k - x^*||_2$?
  There is no such Krylov space method!

- By minimizing $||r_k||_2$!
  Done by MINRES ("Minimum Residual") for symmetric (indefinite) A and GMRES ("Generalised Minimum Residual") for non-symmetric A.

- By making r_k orthogonal to K, ie the space spanned by the vectors $A^(m)x_0$. Called the "orthogonal residual property" or "Galerkin condition". For symmetric A the SYMMLQ method works. For non-symmetric A the GMRES will do.

- When A is SPD, the conjugate gradient method will minimize $||x_k - x^*||_{A^{-1}}$ (where $||z||_A = \sqrt{z^T Az}$).

- If A is SPD, CG is the choice! Even if MINRES seems to be better, minimizing $||r||_2$ rather than $||r||_{A^{-1}}$, MINRES is more work, is less stable and thus often

gives a worse result.

- CG only needs to keep 4 vectors, and does very few operations.
- The search directions p_i are called *A-conjugate* since $p_k^T A p_j = 0$.
- The convergence rate of CG is proportional to the square root of $A : s$ condition number.
- However, the distribution of eigenvalues of A also affect the convergence rate.

- Also CG gain from preconditioning.

Trick?

- Any system $Ax = b$ can be changed into a SPD system using the normal equations $A^T Ax = A^T b$.
- It then includes the least-squares-problem $\min_x ||Ax - b||_2$.
- It may be very ill-conditioned if $\kappa(A)$ is large, but if not, the method is fast!

Decision Tree for Ax=b

see Figure 6.8 in Demmel, p321!

Iterative eigenvalue algorithms

- The power method $x_k = Ax_{k-1}$ has slow convergence if the largest eigenvalue is not well isolated. It is also not efficient - it throws away all computed x_j.
- If we save all the vectors we get the *Krylov subspace*

$$K_k(A, x_0) = \{x_0, Ax_0, A^2x_0, \ldots, A^{k-1}x_0\}$$

- The vectors in the Krylov space will be less and less independent. The *Arnoldi algorithm* will make an orthonormal basis from them.

- Start with the first vector. From the next, remove all dependency on the previous (one step from the Gram-Schmidt algorithm). Then normalize the remaining vector and put it as the next basis vector.

Arnoldi algorithm

- Start with $q_1 = x/||x||_2$
  - for k=1,2,...
  - $u = Aq_k$
  - for j=1,2,...,k.
    $h_{j,k} = q_j^H u$
    $u = u - q_j h_{j,k}$
  - $h_{k+1,k} = ||u||_2$
  - $q_{k+1} = u/h_{k+1,k}$

Lanczos algorithm

- For symmetric A
- Start with $q_1 = x/||x||_2$
  - for k=1,2,...
  - $u = Aq_k - q_{k-1}\beta_{k-1}$
  - $\alpha_k = q_k^H u$
  - $u = u - q_k\alpha_k$
  - $\beta_k = ||u||_2$
  - $q_{k+1} = u/\beta_k$

- The Lanczos uses much less operations per iteration than Arnoldi.
- Lanczos only saves the last two vectors.
- Thus Lanczos manages much bigger problems.
- However, orthogonality gets lost after a while. Re-orthogonalization is as costly as Arnoldi.

Spectral transformation

- A standard practice to find eigenvalue to a large matrix is to apply a Krylov space algorithm, Lanczos or Arnoldi, to a shift invert spectral transformation:

  $C = (A - \sigma B)^{-1} B$, with eigenvalues $\theta_j = 1/(\lambda_j - \sigma)$