Chap 6+7: Iterative Methods

Part b: Krylov Subspace Methods (KSMs)

Introduction: Arnoldi + Lanczos

Lots of Methods:

For spd. problems like Poisson, Conjugate Gradients most common (CG):

Later: decision tree to choose algorithm (Fig 6.8)

KSMs: only requires a "black box" routine to compute $y = Ax$

advantages: - can write a library without knowing details about how A is represented

- can solve problems when explicit A not available, eg requires doing complicated simulation

- if we do have explicit representation of A as a sparse matrix, can reorganize algorithms to avoid communication. Typically $y = Ax$ requires reading A from main memory to cache each time you do $y = Ax$; reorganizations allow one to do multiple matrix-vector multiplies while reading A once

- How to extract information about A given only ability to do $y = Ax$

Given starting vector y_1 (say $y_1 = b$ if solving $Ax = b$) compute

$y_2 = Ay_1$, $y_3 = Ay_2$, ... $y_n = Ay_{n-1} = A^{n-1} y_1$

Let $K = [y_1, y_2, \ldots y_n]$

$AK = [Ay_1, Ay_2, \ldots Ay_n] = [y_2, y_3 \ldots, y_n, A^n y_1]$

Suppose K nonsingular, $c = -K^{-1} A^n y$,

$$AK = K[e_2, e_3, \ldots, e_n, -c] = K \cdot C$$

$$C = K^{-1} A K = \begin{bmatrix} 0 & 0 & & & -c_1 \\ 1 & 0 & & & -c_2 \\ 0 & 1 & \ddots & & -c_3 \\ \vdots & 0 & \ddots & & 1 \\ \vdots & \vdots & & & -c_n \end{bmatrix}$$

C is upper Hessenberg and a companion matrix $\Rightarrow$ its characteristic polynomial is $p(x) = x^n + \sum_{i=1}^{n} c_i x^{i-1}$

So just using $y = Ax$ we can reduce A to a "simple form" and could imagine using C for solving $Ax = b$ or $Ax = \lambda x$

Disadvantages:

(1) K likely to be dense, especially if b dense, so solving with K likely to be hard

(2) K likely to be ill conditioned because columns $A^i b$ are running power method $\Rightarrow$ all converging to be parallel to eigenvector of largest eigenvalue

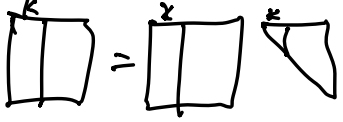
KSMs address these disadvantages: don't compute K, instead an orthogonal Q such that leading k columns of Q span same subspace as leading k columns of K. This space is called

Krylov Subspace:

$$\text{span}\{y_1, y_2, \ldots y_k\} = \text{span}\{y_1, Ay_1, A^2y_1, \ldots A^{k-1}y_1\} = \mathcal{K}_k(A, y_1)$$

relationship between K and Q is

$K = QR$



Also don't need all of K, just compute first $k \ll n$ columns, settle for "best approx" to $Ax=b$ or $Ax=\lambda x$ in $\mathcal{K}_k(A, y_1)$

Many definitions of "best" $\Rightarrow$
many algorithms, all used

$K = QR \rightarrow C = K^{-1} A K \Rightarrow$
$C = R^{-1} Q^T A Q R$

$\Rightarrow$ (*) $RCR^{-1} = Q^T A Q$ = upper Hessenberg $= H$

upper triangular

upper Hessenberg

if $A = A^T \Rightarrow H = H^T \Rightarrow H = T =$ tridiagonal

Compute columns of Q and H one at a time $Q = [q_1, q_2, \cdots q_n]$

(*) $AQ = QH$

$\rightarrow A q_j = \sum_{i=1}^{j+1} q_i H(i,j)$

q_i are orthonormal $\Rightarrow$ multiply by q_m^T

$q_m^T A q_j = \sum_{i=1}^{j+1} H(i,j) \cdot q_m^T q_i = H(m,j)$
for $1 \le m \le j$

$H(j+1,j) \cdot q_{j+1} = A q_j - \sum_{i=1}^{j} q_i H(i,j)$

yields Arnoldi Algorithm:

Arnoldi Alg for partial reduction of A to Hessenberg form:

$q_1 = b / \|b\|_2$

for $j = 1$ to k

$z = A q_j$

for $i = 1$ to j ... run Modified Gram-Schmidt (MGS) on z

$H(i,j) = q_i^T \cdot z$

$z = z - H(i,j) \cdot q_i$

end for

$H(j+1, j) = \|z\|_2$

$q_{j+1} = z / H(j+1, j)$

end for

The q_j vectors called Arnoldi vectors,

Cost: k multiplications by A plus $O(k^2 n)$ for MGS

If we stop at $k < n$ steps, what have we learned about A?

$$Q = [Q_k, Q_u] \qquad Q_k = [q_1, \ldots, q_k]$$
$$Q_u = [q_{k+1}, \ldots, q_n]$$

$$H = Q^T A Q = [Q_k, Q_u]^T A [Q_k, Q_u]$$
$$= \begin{bmatrix} Q_k^T A Q_k & Q_k^T A Q_u \\ Q_u^T A Q_k & Q_u^T A Q_u \end{bmatrix} = \begin{bmatrix} H_k & H_{ku} \\ H_{uk} & H_u \end{bmatrix}$$

H upper Hessenberg so
H_k and H_u are too, and
H_{ku} has one nonzero entry in upper right corner, namely $h_{(k+1,k)}$

$\Rightarrow$ H_k, H_{ku} known, H_{uk}, H_u unknown

when A symmetric so is $H \Rightarrow$
$H = T$ tridiagonal

$$= \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \alpha_2 & \beta_2 & 0 \\ & \beta_2 & \alpha_3 & \ddots \\ & 0 & \ddots & \ddots \\ & & & \alpha_n \end{bmatrix}$$

Equating column j on both sides of

$AQ = QT$ (when $A = A^T$)

$$A q_j = \beta_{j-1} q_{j-1} + \alpha_j q_j + \beta_j \cdot q_{j+1}$$

$$\Rightarrow \quad q_j^T A q_j = \alpha_j$$

$\longrightarrow$ leads to Lanczos Algorithm

Lanczos Algorithm for (partial) reduction of $A = A^T$ to tridiagonal form T

$q_1 = b / \|b\|_2$, $b_0 = 0$, $q_0 = 0$

for $j = 1$ to k

$z = A q_j$

$\alpha_j = q_j^T z$

$z = z - \alpha_j q_j - \beta_{j-1} \cdot q_{j-1}$

$\beta_j = \|z\|_2$

$q_{j+1} = z / \beta_j$

endfor

Notation: $\mathcal{K}_k(A,b) = \text{span}\{q_1, \ldots, q_k\}$
or $\mathcal{K}_k$ for short

We call H_k (T_k for Lanczos) the projection of A onto $\mathcal{K}_k$

KSMs: will find "best" solution to $Ax=b$ or $Ax=\lambda x$ inside $\mathcal{K}_k$

For $Ax=\lambda x$: use eigenvalues of H_k or T_k as approximate eigenvalues of A

Suppose $H_k y = \lambda y$

$$H\begin{bmatrix} y \\ 0 \end{bmatrix} = \begin{bmatrix} H_k & H_{uk} \\ H_{ku} & H_u \end{bmatrix} \cdot \begin{bmatrix} y \\ 0 \end{bmatrix} = \begin{bmatrix} H_k y \\ H_{ku} y \end{bmatrix}$$

$$= \begin{bmatrix} \lambda y \\ H(k+1,k) \cdot y_k \cdot e_1 \end{bmatrix}$$

Since $AQ = QH$

$$A \cdot \left(Q \cdot \begin{bmatrix} y \\ 0 \end{bmatrix}\right) = \lambda \left(Q \begin{bmatrix} y \\ 0 \end{bmatrix}\right) + q_{k+1} H(k+1,k) y_k$$

$\Rightarrow$ if "residual" small ($H(k+1,k) y_k$ small)

then eigenvalue/vector pair

$(\lambda, y') = (\lambda, Q\begin{bmatrix} y \\ 0 \end{bmatrix})$ has a small

residual $\| Ay' - \lambda y' \|_2 = | H(k+1,k) \cdot y_k |$

When $A = A^T$ Thm 5.5 $\Rightarrow$

$| H(k+1,k) y_k |$ is an upper bound

on error in λ

$\rightarrow$ Pictures of how eigenvalues

of T_k converge as k increases

(Fig 7.2 in text)