Welcome back to Ma221! Lecture 38, Nov 27

Krylov Subspace Methods (KSMs)
for $Ax=b$ and $Ax=\lambda x$

Intro: Arnoldi + Lanczos

Many KSMs for $Ax=b$, depending on structure of A

1) General A: GMRES
   Generalized Minimum Residual
2) SPD A (eg Poisson): CG
   Conjugate gradients

see Fig 6.8 in text for decision tree

Unlike Splitting Methods: only need "black box" for Ax, sometimes A^Tx too

Eg don't need diag(A) like Jacobi

1) can write algorithms that are very general, leave details of $A\cdot x$ to user
2) can solve problems where A not explicit, eg Ax from a complicated simulation, or from 'automatically differentiating' a program to get $f'(x)$, eg Tensorflow

If you do have access to A, can optimize to avoid communication, important when $A\cdot x$ requires moving A from slow to fast memory, best case reduces the comm. cost of

k $A\cdot x$'s to cost of 1 (depends on sparsity of A)

How to extract info about A from $A\cdot x$

Given starting vector y_1 (eg. $y_1 = b$, from $Ax=b$)

Compute $y_2 = Ay_1$, $y_3 = Ay_2$, ... $y_{i+1} = Ay_i$

$y_n = A^{n-1} y_1$

$$K = [y_1, y_2, \dots, y_n]^{n\times n}$$

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

If K nonsingular, let $c = -K^{-1} A^n y_1$

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

$$C = K^{-1} A\cdot K = \begin{bmatrix} 0 & 0 & & 0 & -c_1 \\ 1 & 0 & & \vdots & -c_2 \\ 0 & 1 & & \vdots & \vdots \\ \vdots & 0 & & \vdots & \vdots \\ \vdots & \vdots & & \vdots & \vdots \\ & \vdots & & 1 & -c_n \end{bmatrix}$$ upper Hessenberg companion matrix

$$p(x) = \det(xI - C) = x^n + \sum_{i=1}^{n} c_i x^{i-1}$$

Is this useful?

Cons:

(1) K likely dense even if A sparse so solving $Kx=b$ expensive

2) K likely very ill-conditioned

because $y_i \rightarrow$ evec for λ_{max}

Krylov subspace methods address these problems:

(1) Instead of K, compute orthogonal Q:
leading k columns of Q span same space as leading k columns of K

(2) only compute few leading columns of Q, until some convergence criterion satisfied

Define Krylov Subspace:

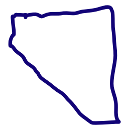
$$= \text{span}\{y_1, \ldots, y_k\}$$
$$= \text{span}\{y_1, Ay_1, A^2y_1 \ldots A^{k-1}y_1\}$$
$$= \mathcal{K}_k(A, y_1)$$

Relationship between K and Q: $K = QR$

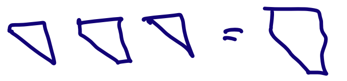
Find "best" solution to $Ax = b$, or $Ax = \lambda x$ inside $\mathcal{K}_k(A, y_1)$

many defs of "best" $\Longrightarrow$ many algorithms

How to compute Q column by column:

$K^{-1}AK = C =$ 

$K = QR \Rightarrow R^{-1}Q^TAQR = C$

(*) $Q^TAQ = RCR^{-1} =$  $= H$

HW Q 6.11

$A = A^T \Rightarrow Q^TAQ = (Q^TAQ)^T = H^T = H = \begin{bmatrix} \ddots & \ddots & & 0 \\ \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \ddots \\ 0 & & \ddots & \ddots \end{bmatrix}$

(*) $Q^TAQ = H \Rightarrow AQ = QH$

equate column j of both sides

$$Aq_j = \sum_{i=1}^{j+1} q_i H_{ij}$$

q_j orthogonal to other q_i

$$q_m^T A q_j = \sum_{i=1}^{j+1} q_m^T q_i H_{ij} = H_{mj} \qquad 1 \le m \le j$$

$$H_{j,j+1} q_{j+1} = Aq_j - \sum_{i=1}^{j} q_i H_{ij}$$

Arnoldi Algorithm for (partial) reduction to upper Hessenberg form:

```
q_1 = y_1/||y_1||_2
for j = 1 to k
    z = Aq_j
    for i = 1 to j
        H_ij = q_i^T z      }  MGS on z
        z = z - H_ij q_i    }
    end for
    H_{j+1,j} = ||z||_2
    if H_{j+1,j} = 0, quit
```

```
        $q_{j+1} = z / H_{j+1,j}$
    end for
```

q_j called Arnoldi vectors

cost $= k$ $A \cdot x$'s

$+ O(k \cdot n^2)$ flops for MGS

What have we learned about A after k steps?

$$Q = \begin{array}{cc} k & n-k \\ \left[\, Q_k \right| & \left. Q_u \,\right] \end{array} \qquad Q_k = [q_1, \ldots, q_k]$$

q_{k+1} known

$$H = Q^T A Q = [Q_k, Q_u]^T A [Q_k, Q_u]$$

$$= \left[\begin{array}{c|c} Q_k^T A Q_k & Q_k^T A Q_u \\ \hline Q_u^T A Q_k & Q_u^T A Q_u \end{array}\right]$$

$$= \left[\begin{array}{c|c} H_k & H_{uk} \\ \hline H_{ku} & H_u \end{array}\right]$$

H upper Hessenberg $\Rightarrow$

H_k and H_u upper Hessenberg

$$H_{ku} = \begin{bmatrix} & \boxed{\bullet} \\ 0 & \end{bmatrix} \quad H_{k+1,k}$$

H_k and H_{ku} known

H_u and H_{uk} unknown

If $A = A^T \Rightarrow H = T = \left[\begin{array}{cccc|c} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \ddots & & \\ & \ddots & \ddots & \beta_{k-1} & \\ & & \beta_{k-1} & \alpha_k & \beta_k \\ \hline & & & \beta_k & \ddots \end{array}\right]$

Equate column j of $AQ = QT$

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

Multiply both sides by $q_j^T \Rightarrow$

$q_j^T A q_j = \alpha_j$

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

```
$q_1 = y_1 / \|y_1\|_2$ , $\beta_0 = 0$, $q_0 = 0$
for $j = 1$ to $k$
    $z = A \cdot q_j$
    $\alpha_j = q_j^T z$
    $z = z - \alpha_j \cdot q_j - \beta_{j-1} q_{j-1}$   ... MGS
    $\beta_j = \|z\|_2$
    if $\beta_j = 0$, quit
    $q_{j+1} = z / \beta_j$
end for
```

How do we use Arnoldi or Lanczos to solve $Ax=b$ or $Ax=\lambda x$?

Consider $Ax=\lambda x$: use evals of H_k (or T_k) as approx. evals of A

To estimate error:

$$H_k y = \lambda y$$

$$H\begin{bmatrix} y \\ 0 \end{bmatrix} = \begin{bmatrix} H_k & H_{uk} \\ H_{ku} & H_u \end{bmatrix}\begin{bmatrix} y \\ 0 \end{bmatrix} = \begin{bmatrix} H_k y \\ H_{ku} y \end{bmatrix} = \begin{bmatrix} \lambda y \\ H_{k+1,k} y_k \cdot e_1 \end{bmatrix}$$

$$AQ = QH$$

$$A\underbrace{\left(Q\begin{bmatrix} y \\ 0 \end{bmatrix}\right)}_{\text{approx evec}} = \lambda\left(Q\begin{bmatrix} y \\ 0 \end{bmatrix}\right) + \underbrace{q_{k+1} H_{k+1,k} \cdot y_k}_{\text{"error"}}$$

$$\|\text{error}\|_2 = |H_{k+1,k} \cdot y_k|$$

so if $|H_{k+1,k} \cdot y_k|$ small $\Rightarrow$ evec/eval pair $\left(Q\begin{bmatrix} y \\ 0 \end{bmatrix}, \lambda\right)$ has small residual

If: $A = A^T$, Thm 5.5 $\Rightarrow$ $|H_{k+1,k}\, y_k|$ bounds distance from λ to nearest eval of A (see Fig 7.2 in text)

Solving $Ax=b$: find "best" approx to $A^{-1}b$ in $\mathcal{K}_k$: $x_k \in \mathcal{K}_k$

1) Choose x_k to minimize $\|x_k - x\|$ $\quad x = A^{-1}b$

but don't have enough info, x "unknown"

2) Choose x_k to minimize residual

$\|r_k\|_2 \qquad r_k = b - Ax_k$

2 Algorithms:

A general: use GMRES

$A = A^T$: use MINRES

3) Choose x_k so $r_k \perp \mathcal{K}_k \qquad r_k^T Q_k = 0$

"orthogonal residual property"

or a Galerkin condition

$A = A^T \Rightarrow$ use SYMMLQ

A general $\Rightarrow$ variant of GMRES

4) A s.p.d $\Rightarrow$ define norm $\|r\|_{A^{-1}} = (r^T A^{-1} r)^{1/2}$

"best" solution minimizes

$$\begin{aligned}
\|r_k\|_{A^{-1}}^2 &= r_k^T A^{-1} r_k \\
&= (b - Ax_k)^T A^{-1} (b - Ax_k) \\
&= (Ax - Ax_k)^T A^{-1} (Ax - Ax_k) \\
&= (x - x_k)^T A \, (x - x_k) \\
&= \| x - x_k \|_A^2
\end{aligned}$$

alg is Conjugate Gradients

Thm: A s.p.d $\Rightarrow$ defs 3) and 4)

are equivalent

only need 3 vectors, + few dot products

and axpy's per iteration