7/8/14

Math 5600

- Now building up to the most widely used general purpose algorithm for computing all eigenvalues and eigenvectors of a general dense matrix.

- good example for typical process in mathematics: start with something simple and let it grow into something quite sophisticated.

- We'll start with the <u>power method</u>

- Suppose $A \in \mathbb{R}^{n \times n}$ has a <u>dominant</u> eigenvalue

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$$

$$Ax_i = \lambda_i x_i$$

- Given $q^{(0)}$

- For $k = 0, 1, 2, \dots$

$$q^{(k+1)} = A q^{(k)}$$

- Suppose $q^{(0)} = \sum_{j=1}^{n} \alpha_j x_j$

$$q^{(1)} = A q^{(0)} = \sum_{j=1}^{n} \alpha_j A x_j = \sum_{j=1}^{n} \alpha_j \lambda_j x_j$$

$$q^{(k)} = \sum_{j=1}^{n} \alpha_j \lambda_j^k x_j$$

- Eventually the λ_1 term dominates, and $q^{(k)}$ is close to an eigenvector.

- Of course we need to guard against floating point over and underflow, i.e., we need to normalize.

For $k = 0, 1, 2, \dots$

$$z^{(k+1)} = A q^{(k)}$$

$$q^{(k+1)} = \frac{z^{(k+1)}}{\| z^{(k+1)} \|}$$

- How do we estimate $\lambda = \lambda_1$?

- Suppose q is an approximation of the dominant eigenvector.

-
$$F(\lambda) = \| Aq - \lambda q \|^2 \qquad \| \ \| = \| \ \|_2$$

$$= (Aq - \lambda q)^T (Aq - \lambda q) \qquad \|q\| = 1$$

$$= \min$$

$$F(\lambda) = q^T A^T A q - \lambda q^T A q - \lambda q^T A^T q + \lambda^2 \underbrace{q^T q}_{1}$$

$$\nabla F(\lambda) = 2 \lambda q^T q - \lambda q^T (A + A^T) q = 0$$

$$\lambda = \frac{1}{2} q^T (A + A)^T q$$

if $\|q\| \neq 1$ $\quad \lambda = \dfrac{q^T (A + A^T) q}{q^T q}$

if A is symmetric

$$\lambda = \frac{q^T A q}{q^T q}$$

is the <u>Rayleigh Quotient</u>

- What can go wrong?
  - no dominant evalue
    - λ_1 real, multiple
      converge to a vector in the corresponding invariant subspace
    - λ_1 complex
      get an oscillation.
      e.g. $A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$

$$q^{(0)} = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \quad q^{(1)} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \quad q^{(2)} = \begin{bmatrix} -1 \\ 1 \end{bmatrix} \cdots$$

  - what if $\alpha_1 = 0$
    - technically you'd get convergence to another eigenvector.
    - but round-off errors bail you out
  - $\frac{|\lambda_1|}{|\lambda_2|}$ close to 1.
    slow convergence, we'll have to address that.

- Modifications of power method:
- shift of origin

Apply the power method to

$B = A - \mu I$ for some μ

eigenvalues of B are $\lambda_i - \mu$ $i = 1, \ldots, n$

power method will converge to dominant eigenvalue of B

- Inverse iteration.

Apply the power method to A^{-1}

- of course we don't compute A^{-1}

$q^{(0)}$ given

Solve:
$$\left.\begin{aligned} A z^{(k+1)} &= q^{(k)} \\ q^{(k+1)} &= \frac{z^{(k+1)}}{\| z^{(k)} \|} \end{aligned}\right\} k = 0, 1, \ldots$$

- Shift of origin can be combined with inverse iteration
- Apply the PM to $(A-\mu I)^{-1}$

$$A - \mu I = LU$$

$q^{(0)}$ given

$$L\,y^{(k+1)} = q^{(k)}$$

$$U\,z^{(k+1)} = y^{(k+1)}$$

$$q^{(k+1)} = \frac{z^{(k+1)}}{\| z^{(k+1)} \|}$$

- converges to the dominant eigenvalue η of $A-\mu I$

- corresponding eigenvalue of A is λ

$$\eta = \frac{1}{\lambda - \mu}$$

$$\lambda = \mu - \frac{1}{\eta}$$

- so we could find any evalue of A in principle.

- even complex.

- question: when do we terminate the iteration.

- Interesting complication

  - We want μ close to λ
  - But if μ is an eigenvalue then A is singular.
  - the closer μ is to λ the more ill-conditioned is A

- It's OK see Peters and Wilkinson, 1971

- The power method is temperamental!

- Interestingly EISPACK and LAPACK do not provide an option of computing a single evector/value pair.

- It's just hard to make a general

- nonetheless the PM is the basis of the QR algorithm for finding all evalues/vectors.

- How about using the PM to find several evalues/vectors.

- what about

$$Y^{(0)} \text{ given} \quad Y^{(0)} \in \mathbb{R}^{n \times p}$$

$$Z^{(k+1)} = A Y^{(k)} \qquad k = 0, 1, \ldots$$

$$Y^{(k+1)} = \frac{Z^{(k+1)}}{\|Z^{(k+1)}\|}$$

- This is like running the PM independently in each column of Z

- no good. We have to make sure that the columns remain linearly independent.

- how independent?

- how about orthonormal?

- Orthogonal iteration:

$Q_0 = Q \quad n \times r \quad$ (possibly complex)

$$Q^H Q = I$$

For $k = 1, 2, \ldots$

$$Z_k = A Q_{k-1}$$

compute $\quad Q_k R_k = Z_k$

if $r = 1$ this is just the PM

- moreover, as far as the first column of Q is concerned, this is just the PM

- moreover, as far as the first s columns of Q are concerned this is just orthogonal iteration with r replaced by s.

- This is a key ingredient. We are running r orthogonal iterations simultaneously.

- So how about $\underline{r = n}$?

$$T_k = Q_k^H A Q_k$$

T_k is similar to A.

- Now observe this

since $A Q_{k-1} = Z_k = Q_k R_k$

$$T_{k-1} = Q_{k-1}^H A Q_{k-1} = Q_{k-1}^H Z_k = \underbrace{Q_{k-1}^H Q_k} R_k$$

$$T_k = Q_k^H A Q_k = Q_k^H A Q_{k-1} Q_{k-1}^H Q_k$$

$$= Q_k^H Z_k Q_{k-1}^H Q_k$$

$$= Q_k^H Q_k R_k Q_{k-1}^H Q_k$$

$$= R_k \underbrace{Q_{k-1}^H Q_k}$$

- So we can think of orthogonal iteration with $r = n$ as

$$T_0 = A$$

For $k = 1, 2, 3, \ldots$

Factor $T_{k-1} = Q_{k-1} R_{k-1}$

set $T_k = R_{k-1} Q_{k-1}$

(not Q_{k-1} is what used to be $Q_{k-1}^H Q_k$)

- more succinctly:

For $k = 1, 2, \ldots$

$$A = QR$$
$$A = RQ$$

- This is equivalent to running n orthogonal iterations simultaneously

- it converges if

$$|\lambda_1| > |\lambda_2| > \ldots > |\lambda_n|$$

- it's expensive and converges only slowly