
Lecture Notes 9, Math/Comp 128, Math 250

Misha Kilmer
Tufts University

October 31, 2007

Computing Eigenvalues

Recall that we are considering a **two Phase** process

1. Reduction (via similarity transforms) to upper Hessenberg (tridiagonal, in the hermitian case) form (call this H) using Householder reflections or Givens rotations. $O(m^3)$ flops. Direct method. Mathematically,

$$H = Q_{m-2}^* \cdots Q_2^* Q_1^* A Q_1 \cdots Q_{m-2}$$

* Note that we have to keep track of the Q 's somehow if we ultimately want eigenvectors, and not just the eigenvalues.

2. Apply an iterative method to compute the eigenvalues (eigenvectors) of H .

Householder Reduction

$$\underbrace{\begin{bmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{bmatrix}}_A \xrightarrow{Q_1^*} \underbrace{\begin{bmatrix} x & x & x & x & x \\ \mathbf{x} & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{bmatrix}}_{Q_1^*A} \xrightarrow{Q_1} \underbrace{\begin{bmatrix} x & \mathbf{x} & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ x & \mathbf{x} & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} & \mathbf{x} \end{bmatrix}}_{Q_1^*A Q_1}$$

Householder reduction

$$x = A(2 : 5, 1); v = \text{sign}(x_1)\|x\|_2 e_1 + x; v \rightarrow v/\|v\|_2$$

$$F = I - 2vv^*$$

$$Q_1^* = \begin{bmatrix} 1 & 0^T \\ 0 & F \end{bmatrix}$$

Householder reduction

$$Q_1^* A = \begin{bmatrix} a_1^T \\ (I - 2vv^*)A(2:5, :) \end{bmatrix}$$

And we can compute the bottom block as $A(2:5, :) - 2v(v^*A(2:5, :))$

Recall that $F = F^*$, so $Q_1^* = Q_1$

$$BQ_1 = \left[b_1 \quad (I - 2vv^*)A(:, 2 : 5) \right]$$

Compute the new columns as $A(:, 2 : 5) - 2(A(:, 2 : 5)v)v^*$.

Algorithm

for $k = 1:m-2$

- $x = ?$
- $v_k = ?$
- $v_k = v_k / \text{norm}(v_k);$
- $A() =$
- $A() =$

Algorithm

for $k = 1:m-2$

- $x = A(k + 1 : m, k)$;
- $v_k = \text{sign}(x(1))\text{norm}(x)e_1 + x$;
- $v_k = v_k/\text{norm}(v_k)$;
- $A(k + 1 : m, k : m) = A(k + 1 : m, k : m) - 2v_k(v_k^*A(k + 1 : m, k : m))$;
- $A(k : m, k + 1 : m) = A(k : m, k + 1 : m) - 2(A(k : m, k + 1 : m)v_k)v_k^*$;

The Symmetric Case

In exact arithmetic, this must reduce to tridiagonal form. Why??

Stability for Phase I

Theorem 26.1 Let the Hessenberg reduction $A = QHQ^*$ of a matrix $A \in \mathbb{C}^{m \times m}$ be computed by Algorithm 26.1 on a computer satisfying the axioms (13.5) and (13.7), and let the computed factors \tilde{Q} and \tilde{H} be defined as indicated. Then

$$\tilde{Q}\tilde{H}\tilde{Q}^* = A + \delta A, \quad \frac{\|\delta A\|}{\|A\|} = O(e_{mach})$$

for some $\delta A \in \mathbb{C}^{m \times m}$.

Phase II – Symmetric Matrices Only

Chapters 27-30 for symmetric matrices only. Then we know that the eigenvalues are real and the eigenvectors are orthonormal.

First Phase II option, power iteration.

Power Iteration

Choose v so that $\|v\| = 1$, expand in the eigenbasis.

Recall $v = c_1q_1 + c_2q_2 + \dots + c_mq_m$, and

$$v^{(k)} = \alpha_k A^k v = \alpha_k (c_1 \lambda_1^k q_1 + \dots + c_m \lambda_m^k q_m)$$

$$v^{(k)} = \alpha_k \lambda_1^k \left(q_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^k q_2 + \dots + \left(\frac{\lambda_m}{\lambda_1}\right)^k q_m \right)$$

α_k is a normalization constant

Power Iteration

$v^{(0)} = v$ for $k=1:\text{maxits}$

- $w = Av^{(k-1)}$
- $v^{(k)} = \frac{w}{\|w\|}$
- $\lambda^{(k)} = (v^{(k)})^T Av^{(k)}$

The right-hand side of the last equation is called the **Raleigh quotient**

Why (intuitively) do we think this might be converging to λ_1 ?

Convergence

Theorem 27.1 Suppose $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \dots \geq |\lambda_m| \geq 0$ and $c_1 \neq 0$. Then

$$\|v^{(k)} - (\pm q_1)\| = O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)$$

and

$$|\lambda^{(k)} - \lambda_1| = O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^{2k}\right)$$

Issues

In summary, this can be SLOW if the relative gap between $|\lambda_2|$ and $|\lambda_1|$ is not large.
(See homework)

Why do we apply this in Phase II (to the upper Hessenberg matrix)? Note the amount of work per iteration is a matrix vector product, and recall that we are restricting ourselves to real, symmetric matrices.

Note this also only gets us one piece of information – the largest magnitude eigenvalue (eigenvector) (Google matrix).

Alternatives that address convergence

Alternative Number 1: Inverse Iteration.

Recall that the eigenvalues of $A - \mu I$ are $\lambda_i - \mu, i = 1, \dots, m$.

Now, Let's think about the eigenvalues of $(A - \mu I)^{-1}$ (we have to assume μ is NOT an eigenvalue of A or we're in trouble!).

Fact: If λ is an eigenvalue of an invertible matrix B , then $1/\lambda$ is an eigenvalue of B^{-1} .

Fact: The eigenvectors are the same for B and B^{-1} .

Proof on board.

How to choose μ ? We want to improve the convergence rate...

Let's do the expansion, and try to devise the algorithm...

Inverse Iteration

Start: v is some vector with unit length

for $k=1:\text{maxits}$

- **Solve** $(A - \mu I)w = v^{(k-1)}$ for w
- $v^{(k)} = w/\|w\|$
- $\lambda^{(k)} = (v^{(k)})^T A v^{(k)}$

Inverse Iteration

Theorem 27.2 If λ_J is the closest eigenvalue to μ and λ_K is the second closest (i.e. $|\mu - \lambda_J| < |\mu - \lambda_K| \leq |\mu - \lambda_j|, j \neq J$), and $q_J^T v \neq 0$, then

$$\|v^{(k)} - (\pm q_J)\| = O\left(\left|\frac{\mu - \lambda_J}{\mu - \lambda_K}\right|^k\right), \quad |\lambda^{(k)} - \lambda_J| = O\left(\left|\frac{\mu - \lambda_J}{\mu - \lambda_K}\right|^{2k}\right).$$

Issues?

What if μ is really close to an eigenvalue? Then isn't the matrix very ill-conditioned?
Can we trust the estimate of w ?

Note the potential expense per iteration.

- What's the cost of the solve, if we only apply this to matrices that have been run through Phase I already?
- Can use this approach to pick out eigenvalues and eigenvectors that are not necessarily the largest in magnitude, just by choosing μ .
- One of the **most valuable tools of numerical linear algebra** b/c it is the standard method of calculating one or more eigenvectors if the eigenvalue(s) are already known.
- Of course, how do you get μ if you do not know an eigenvalue exactly? (Let's talk briefly about Gerschgorin's Theorem)

Gerschgorin's Theorem

How might you get some clue as to the eigenvalue distribution for your matrix?

Theorem (see also problem 24.2) Every eigenvalue of A lies in at least one of the m circular disks in the complex plane with centers a_{ii} and radii $\sum_{j \neq i} |a_{ij}|$. Moreover, if n of these disks form a connected domain that is disjoint from the other $m - n$ disks, then there are precisely n eigenvalues of A within this domain.

Try to apply to this example.

$$A = \begin{bmatrix} 8 & 1 & 0 \\ 1 & 4 & \epsilon \\ 0 & \epsilon & 1 \end{bmatrix}, \quad |\epsilon| < 1$$

Combination: Rayleigh-Quotient Iteration

If we had a good μ close to our desired λ , the Inv. Iteration is good at finding the corresponding eigenvector. Note that we used the Rayleigh Quotient to get our estimate of the eigenvector in the case it wasn't already known. Put the ideas together, so that the shifts are continually updated, based on our most recent estimate of λ .

$v^{(0)}$ is a random vector with unit norm $\lambda^{(0)} = (v^{(0)})^T A v^{(0)}$

for $k = 1:\text{maxits}$

- **Solve** $(A - \lambda^{(k-1)}I)w = v^{(k-1)}$ for w
- $v^{(k)} = w/\|w\|$
- $\lambda^{(k)} = (v^{(k)})^T A v^{(k)}$

RQ Iteration

Theorem 27.3 RQ Iteration converges to an eigenvalue/eigenvector pair for all except a set of measure zero of starting vectors $v^{(0)}$. When it converges, the convergence is ultimately cubic as k goes to ∞ in the following sense:

$$\|v^{(k+1)} - (\pm q_J)\| = O(\|v^{(k)} - (\pm q_J)\|^3)$$

and

$$|\lambda^{(k+1)} - \lambda_J| = O(|\lambda^{(k)} - \lambda_J|^3).$$

See example

$$A = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 3 & 1 \\ 1 & 1 & 4 \end{bmatrix} .$$