1 Lecture 11

Eigenvalue Problems

Definition 1.1. ALgebraic multiplicity of an eigenvalue counts the number of times the eigenvalue occurs as a root of the characteristic equation. Geometric multiplicity counts how many linearly independent eigenvectors correspond to an eigenvalue. It is a proveable fact that algebraic multiplicity is at least as much as the geometric multiplicity. If algebraic multiplicity is greater than geometric multiplicity, we say that the matrix is defective.

Theorem 1.2. Similar matrices share the same eigenvalues.

Theorem 1.3. If a matrix is defective, then it cannot have an eigenvector basis.

Proposition 1.4. A matrix is diagonalizable iff it has an eigenvector basis.

Proposition 1.5. The following matrix transformations change eigenvalues and eigenvectors as follows:

- $A \to (A - \sigma I)$ causes $\lambda \to \lambda - \sigma$.
- $A \to A^{-1}$ causes $\lambda \to \frac{1}{\lambda}>$
- $A \to A^k$ causes $\lambda \to \lambda^k$.
- If $A = PXP^{-1}$, then X has the same eigenvalues but every eigenvector v now becomes $P^{-1}v$.

Proposition 1.6. Suppose that we perturb a diagonal matrix A with some matrix E. Then the distance between any eigenvalue u of $A+E$ to an eigenvalue of A λ_k closest to u is bounded by $k(A)\|E\|$.

2 Lecture 12

Problem C. haracteristic Polynomial For which of the following reasons is the characteristic polynomial of a matrix NOT useful, in general, for computing the eigenvalues of the matrix?

Select all that apply: Its coefficients may not be well determined numerically. Its roots may be difficult to compute. Its roots may be sensitive to perturbations in the coefficients. None of these

Solution 2.1. Yes, the roots change if the coefficients change – hence any perturbation will affect the roots; determining the coefficients numerically is also problematic, simply because numerical comptutation requires rounding error and truncation error. The second is a given.

Problem P. roblem Transformations and Spectral Radius Which of the following transformations preserve the spectral radius of a matrix A?

Select all that apply: Powers None of these Shift Polynomial Inversion

Solution 2.2. Obviously, none of them, since they all change the eigenvalues.

Problem D. iagonalizability Of the classes of n×n matrices listed below, which is the smallest class of matrices that are not necessarily diagonalizable by a similarity transformation?

Choice* normal matrices all matrices real symmetric matrices matrices with n distinct eigenvalues

Solution 2.3. A spectral theorem asserts that normal matrices are unitarily diagonalizable; a theorem asserts that real symmetric matrices have an orthogonal basis – hence they are diagonalizable; in general, any matrix with an eigenbasis is diagonalizable.

Problem L. et $A = XDX^{-1}$. Suppose $\hat{A} = A + \delta A = \hat{X}\hat{D}(\hat{X})^{-1}$. Which matrix is $\hat{A}$ similar to?

Solution 2.4. $X^{-1}\hat{A}X = X^{-1}AX + X^{-1}\delta AX = D + X^{-1}\delta AX$

Definition 2.5. The eigenvector corresponding to the largest eigenvalue will hence forth be called the maximal eigenvector.

Remark 2.6. Power iteration can fail because of certain reasons; certain of these can be remedied:

- No component alongside the dominant eigenvector.
  - Rounding error usually introduces some component – and a random vector will usually include the component.
- Overflow of entries in the vector being iterated upon:
  - Normalize the vector
- There is no one dominant eigenvector, because two distinct eigenvalues of equal, maximal magnitude exist.

Definition 2.7. The rayleight quotient is the quantity

$$\frac{x^T Ax}{x^T x}$$

Remark 2.8. Let $e_k = \left\|v_1^k - x_1\right\|$ where v_1^k is the estimate of x_1 at the kth iteration.

It can be shown that error $\approx c\left|\frac{\lambda_2}{\lambda_1}\right|^k$, which implies that

$$\frac{\|e_{k+1}\|}{\|e_k\|} = \left|\frac{\lambda_2}{\lambda_1}\right|$$

This is called linear convergence, because the error in the next iteration is a linear (think $y = ax$) scaling of the previous error.

Power iteration obviously costs $O(n^2)$ at each iteration, because we're repeatedly multiplying a matrix by a vector.

Definition 2.9. Inverse power iteration is the algorithm that repeatedly hits x with A^{-1}. Its statistics:

- The error rate is

$$\frac{|1/\lambda_{n-1}|}{|1/\lambda_n|} = \frac{|\lambda_n|}{|\lambda_{n-1}|}$$

- The cost is initially $O(n^3)$ since we solve for y in $Ay = x$. Thereafter, the cost is $O(n^2)$.
- The largest eigenvalue is $\left|\frac{1}{\lambda_n}\right|$

Definition 2.10. Shifted inverse power iteration iterates on x using $(A-\sigma I)^{-1}$.

The dominant eigenvector corresponds to the eigenvalue $\frac{1}{\lambda'-\sigma}$ where λ' is the closest eigenvalue of A to σ. If λ'' is the second closest eigenvalue to A then the error rate is given by

$$\frac{|\lambda' - \sigma|}{|\lambda'' - \sigma|}$$

The cost is initially $O(n^3)$ but then $O(n^2)$ thereafter.

Definition 2.11. Shifted Rayleigh iteration leverages both power iteration and the Rayleight quotient method in order to determine the maximal eigenvector:

- Compute the rayleigh quotient to obtain σ_k.
  - Multiply $(A - \sigma_k I)^{-1}$ by x_k.
    - In reality, by multiplication, we mean compute the LU factorizaton of $A - \sigma_k I$ and then use it
- This will presumably give us an eigenvector that corresponds to the eigenvector closest to σ.
  - Note that σ was not chosen with any prior intent however – it just happens to be our estimate of the eigenvalue that corresponds to some initial x, and that initial x is our estimate of some eigenvector. to compute the multiplication.
- This is good, because it can be shown to converge at a quadratic rate – whereas ordinary power iteration convergence is linear.
  - This is bad, because we have to factor $A-\sigma_k I$ every iteration, which can be expensive.
  - This may not also converge to an eigenvector we want to converge to apriori.

Definition 2.12. Simultaneous iteration seeks to determine the maximal p eigenvectors; or rather a basis for them.

Initialize some X_0, an $n \times p$ matrix.
$X_{k+1} = AX_k$
⟨⟨ You can perform orthonormalization if needbe ⟩⟩

Remark 2.13. This algorithm is problematic, because the column vectors in X_k may tend to the dominant eigenvector. As a result, X_k becomes increasingly more and more ill conditioned. This is bad, in and of itself, because we typically desire that $X_k \to X$, which we define to be a basis for the span of the dominant p eigenvectors.

Definition 2.14. One remedy to this problem is to orthonormalize the iterate of the basis. The resulting algorithm is called orthonormal iteration.

In what follows, assume that we use the full unreduced QR factorization.

> Initialize some Q_0, corresponding to our guess of a basis for the dominant p eigenvectors.
> Q_0 is an orthogonal matrix that is $n \times n - p$
> While $\|Q_{k+1} - Q_k\| >$ some tolerance
> $X_{k+1} := AQ_k$
> $Q_{k+1}R_{k+1} := X_{k+1}$

Notice that at each iteration, we have $A = Q_{k+1}R_{k+1}Q_k^T$. If $Q_k \to Q$ then $A = QRQ^T$ (R also converges as a consequence), so that we have found some (allegedy dominant) p eigenvectors in Q and their eigenvalues in the $p \times p$ upper triangular R.

Remark 2.15. Assume now that we use the reduced QR factorization so that Q is $n \times r$ and R is $r \times r$.

The span of Q_k is the span of X_k at any point. Thus $X_{k+1} = AQ_k$ gives us a set of column vectors whose span is the same as the span of AX_k. Thus, we can use Q_k as a proxy to X_k.

Why is span(Q_k) =span(X_k) – since otherwise, the QR factorization of X_k (which we assume to be of rank r – or else – we are not determining distinct eigenvectors) will reduce the rank of X_k.

Definition 2.16. In QR iteration, we decompose A_k as Q_kR_k and then obtain $A_{k+1} = R_kQ_k$. Notice that $A_{k+1} = Q_k^TA_kQ_k$ so that if Q_k converges then A_{k+1} is now (this is not yet underestood) really an upper triangular matrix with eigenvalues along its diagonal.

Remark 2.17. It is instructive to look at the iterates of simultaneous iteration and QR iteration:

$$\begin{aligned} \hat{Q}_1R_1 &= X_0 \\ X_1 &= A\hat{Q}_1 \\ \hat{Q}_2R_2 &= X_1 \\ X_2 &= A\hat{Q}_2 \end{aligned}$$

$$\hat{Q}_3R_3 = X_2 \implies \hat{Q}_3R_3\hat{Q}_2^{\ H} = A \implies \hat{Q_{k+1}}R_{k+1}\hat{Q}_k^{\ H} = A$$

QR iteration:

$$
\begin{aligned}
Q_1 R_1 = A_0 &= A \\
A_1 &= R_1 Q_1 \\
Q_2 R_2 &= A_1 \\
A_2 &= R_2 Q_2 \\
\implies A_2 &= Q_2^H A_1 Q_2 \\
\implies A_{k+1} &= Q_{k+1}^H A_k Q_{k+1}
\end{aligned}
$$

There is an equivalence between the two forms of iteration that can be expressed as follows:

Suppose that we have $\hat{Q_{k-1}}$ and that we wish to form $X_{k-1} = A\hat{Q_{k-1}}$ and then factorize the resultant product as $\hat{Q}_k R_k$. Assume further that have already factorized A_{k-1} as $Q_k R_k$. Then we can compute X_{k-1} without any work that we would normally do for simultaneous iteration:

$$
X_{k-1} = A\hat{Q_{k-1}} = \hat{Q_{k-1}}\hat{Q_{k-1}}^H A\hat{Q_{k-1}} = \hat{Q_{k-1}} A_{k-1} = Q_{k-1}\hat{Q}_k R_k
$$

If we set $\hat{Q}_k := \hat{Q_{k-1}} Q_k$, and appeal to the uniquess of QR, then we are done.

Definition 2.18. In QR iteration with shifting, we factor $A_k - \sigma_k I$ as $Q_k R_k$ and then compute

$$
A_{k+1} = R_k Q_k + \sigma I
$$

Note that since $R_k = Q_k^T (A_k - \sigma_k I)$, it follows that

$$
A_{k+1} = Q_k^T (A_k - \sigma_k I) Q_k + \sigma I = Q_k^T A_k Q_k
$$

Remark 2.19. Once again, it is instructive to see what the iterates of QR iteration with shifting look like:

$$
\begin{aligned}
Q_1 R_1 &= (A_0 - \sigma_0 I) \\
A_1 &= R_1 Q_1 + \sigma_0 I \\
\implies A_{k+1} &= Q_{k+1}^H (A_k - \sigma_k I) Q_{k+1} + \sigma_0 I \\
&= Q_{k+1}^H A_k Q_{k+1}
\end{aligned}
$$

Definition 2.20. The Schur decomposition of a matrix $A = QUQ^H$ expresses A as similar to an upper triangular matrix U. A proof of this theorem is given in `http://people.inf.ethz.ch/arbenz/ewp/Lnotes/chapter2.pdf` at page 6. We now list some properties pertaining to the matrix:

- Since $AQ = QU$, we can conclude that $AQ_1 = Q_1(U_{1,1})$. From this, it follows that the first schur vector Q_1 is an eigenvector.

- If Q and U are real, then QU is a QR decomposition – obviously.
- If A is real symmetric, then it must be true that U is diagonal, in which case it contains all the eigenvalues of A.
  - Moreover, since $U = Q^H AQ$, one can shown that $U^* = U$, implying that all the eigenvalues are real. A proof near the proof listed above additionally explains that eigenvectors corresponding to distinct eigenvalues are orthonormal.

Proposition 2.21. With the schur form of a matrix, we can also use the triangular matrix U to construct eigenvectors.

Proof. Subtract λI from U. Then we get

$$\begin{bmatrix} U_{1,1} & uU_{1,3} \\ & 0v^T \\ & U_{3,1} \end{bmatrix}$$

Note that $U_{1,1}$ is also triangular, u is a column vector, $U_{1,3}$ a rectangular block, v a column vector and $U_{3,1}$ another triangular matrix. Then

$$\begin{bmatrix} -U_{1,1}^{-1}u \\ 1 \end{bmatrix}$$

is an eigenvector. □

3 Lecture 13

Problem S. uppose we are given the matrix

$$A = \begin{bmatrix} 1 & 0 \\ 3 & 0.1 \end{bmatrix}$$

Then how many power iterations do we need in order to reduce the error by 10^{-10}.

Solution 3.1. The convergence rate is given by $\left|\frac{\lambda_2}{\lambda_1}\right|$; hence the rate is 10^{-1} and we need at least 10 iterations. Note that we additionally assume that the starting vector has components in both directions corresponding to the eigenvalue.

Problem A. ssume that we are given an LU factorization from a previous computation. What is the cost of performing inverse power iteration for k iterations on an $n \times n$ matrix.

Solution 3.2. The cost is $O(kn^2)$. Recall that LU factorizations, once calculated, allow us to solve inverse problems in $O(n^2)$ time.

Problem E. stimate the rayleight quotient of a given matrix for a given eigenvector. Trivial; whence the solution is ommitted.

Problem P. roperties of the Schur Form Consider the Schur decomposition of a matrix A=QUQH where U is upper-triangular and Q is unitary. The Schur vectors are the columns of Q. While of the following are true?

Select all that apply: If A is real, Q is orthogonal and U is real The first Schur vector is an eigenvector of A The Schur form of a matrix is unique so long as its eigenvalues are not all the same If A is real symmetric, the Schur decomposition is the eigenvalue decomposition of A If Q and U are real, QU is a QR decomposition of AQ

Solution 3.3.

- If A is real, Q is orthogonal and U is real
  - A coudl be real but Q and U still complex.
- The first Schur vector is an eigenvector of A
  - True – see above.
- The Schur form of a matrix is unique so long as its eigenvalues are not all the same
  - In the proof of the schur decomposition, the eigenvalue (which is entry occupying the first diagonal entry of Λ) was chosen without any specificity – so that any eigenvalue can be chosen there.
- If A is real symmetric, the Schur decomposition is the eigenvalue decomposition of A
  - Yes, see above.
  - If Q and U are real, QU is a QR decomposition of AQ.
  - Yes, can be observed from the form of the schur decomposition itself.

Problem Q. R Iteration for Special Matrices Which of the following statements about doing one iteration of QR decomposition on a matrix An×n is true?

Select all that apply: For a Hessenberg matrix A, it takes O(n2) time. For a Hessenberg matrix A, it takes O(n3) time. For a tridiagonal matrix A, it takes O(n) time. For a tridiagonal matrix A, it takes O(n2) time.

Solution 3.4. Imagine that we use Givens' rotations. Take the kth column as an example. After finding the rotation that takes the $k+1$th row to the kth row, we need to apply this same rotation to the remaining $n-k$ columns to the right of the kth column. Thus, even if the givens' rotation takes $O(n)$ time to both compute and apply its effects to the $k$the column, the cost of its application to the remaining columns results in $O(n^2)$. By contrast, in a tridiagonal matrix, after finding the rotation that "settles" the kth column, the rotation only need be applied to the $k+1$th column, since the rotation will change the kth entry of that column; no other column to the right will be affected. Thus, this results in an $O(n)$ cost.

Remark 3.5. To arrive at the schur form of a matrix, we need to perform some unspecified number of QR factorizations. Each factorization takes $O(n^3)$. Thus the cost can often near $O(n^4)$.

Remark 3.6. A better technique is to use a trick involving Householder Transformations:

- Having computed a householder vector h_i that nullifies everything beneath and including row $i+2$, and having formed H_i which is the transformation that achieves this, carry out $H_i A (H_i)^T$. What will this do? It can be shown that this will zero out all entries past and including the $i+2$ row. Doing this for each column, we will attain Hessenberg form. Having attained Hessenberg form. That is $A = \prod_{i=1}^{n} (H_i)^T H (\prod_{i=1}^{n} H_i)$,which is a similarity transform to A, meaning that the eigenvalues for A are contained in H. We can now perform QR iteration on H. It costs $O(n^2)$ to perform the necessary number of Givens rotations to QR factorize the matrix. Suppose that we obtain at the very first iteration the factors $Q_1 R_1 = H$. Observe that $A_2 = R_1 Q_1 = R_1 H R_1^{-1}$. Further reacall that post or pre multiplying a Hessenberg matrix by an upper triangular matrix preserves the form of the Hesenberg matrix. It therefore follows that A_2 is Hessenberg as well; thus every QR iteration now requires $O(n^2)$ as opposed to $O(n^3)$.

Remark 3.7. In a Krylov subspace method, we restrict our focus to finding a few eigenvect

<++>

Definition 3.8.

- Suppose that we begin with an initial vector x_0. Let

$$K_n = \begin{bmatrix} x_0 & Ax_0 & ... & A^{n-1}x_0 \end{bmatrix}$$

Observe that

$$AK_n = \begin{bmatrix} Ax_0 & A^2x_0 & ... & A^n x_0 \end{bmatrix} = K_n \underbrace{\begin{bmatrix} e_2 & e_3 & ... & e_n & K_n^{-1} A^n x_0 \end{bmatrix}}_{H}$$

That is we found that $K_n^{-1} A K_n = H$, where H is upper Hesenberg.

Remark 3.9. So what? Why is this important? We wish to work with H to determine its eigenvalues. The foregoing reveals that we will need to compute K_n if we wish to compute H. The problem is that K_n will tend to multiples of the dominant eigenvector of A, since K_n is obtained by performing power iteration a few times on x_0. This makes K_n highly ill conditioned, so that the Hesenberg form that we would obtain by mat-mat multiplication is likely not useful. What we will therefore do is that we will find a $Q_n R_n$ factorization for K_n.

Remark 3.10.

$$Q_n^H A Q_n = (K_n R_n)^{-1} A (K_n R_n) = (R_n)^{-1} K_n^{-1} K_n K_n R_n = (R_n)^{-1} H R_n \cong C$$

17

where C is some Hesenberg matrix in addition to H.

Proposition 3.11. The vectors of Q_n can each be computed alone.

Proof.

$$AQ_n = Q_n C$$

For notational convenience, let $Q = Q_n$

Let us focus on the jth column. Suppose that $Q_n =$

$$[q_1 \quad q_2 \dots q_n] \implies Aq_j = \sum_{i=1}^{j+1} q_i C_{i,j}$$

Now solve for q_2 assuming that we know q_1 and that we have orthonormalized q_1. This gives us:

$$\begin{aligned} &\implies Aq_1 = \sum_{i=1}^{2} q_i C_{i,1} \\ &\implies Aq_1 = q_1 C_{1,1} + q_2 C_{2,1} \\ &\implies Aq_1 - q_1 C_{1,1} = q_2 C_{2,1} \end{aligned}$$

Note that Aq_1 is the "next" vector in a Krylov subspace that one would compute with knowledge of q_0.

Now in general $C_{i,j} = q_i^H A q_j$. In particular $C_{1,1} = q_1^H A q_1$. If we let $u_1 = (Aq_1)$, then we see that $C_{1,1} = q_1^H u_1$ and that

$$Aq_1 - q_1 C_{1,1} = u_1 - q_1 C_{1,1}$$

This is tantamount to orthogonalizign u_1 by removing all of the projections of u_1 onto previous q_k (in this case, only q_1). Thus if we set $q_2 = \frac{u_1}{\|u_1\|}$, then we will have effectivel found the vector q_2 that is part of the orthonormal set of Krylov space vectors. □

Remark 3.12. This can be formalized as part of an algorithm:

Algorithm 4.9 Arnoldi Iteration

$\boldsymbol{x}_0$ = arbitrary nonzero starting vector
$\boldsymbol{q}_1 = \boldsymbol{x}_0 / \|\boldsymbol{x}_0\|_2$ { normalize }
for $k = 1, 2, \dots$
  $\boldsymbol{u}_k = \boldsymbol{A}\boldsymbol{q}_k$ { generate next vector }
  for $j = 1$ **to** k { subtract from new vector
    $h_{jk} = \boldsymbol{q}_j^H \boldsymbol{u}_k$ its components in all
    $\boldsymbol{u}_k = \boldsymbol{u}_k - h_{jk}\boldsymbol{q}_j$ preceding vectors }
  end
  $h_{k+1,k} = \|\boldsymbol{u}_k\|_2$
  if $h_{k+1,k} = 0$ **then** stop { stop if matrix is reducible }
  $\boldsymbol{q}_{k+1} = \boldsymbol{u}_k / h_{k+1,k}$ { normalize }
end

Remark 3.13. Note that this algorithm provides us with a means of obtaining the Hesenberg matrix that the original matrix A is orthogonally similar to. The vectors q are not so important save that we need them in order to determine the values in the matrix C (which stands for the Hesenberg matrix).

This is done as follows: the crux of this algorithm is the equation

$$Aq_k = \sum_{i=1}^{k+1} C_{ik} q_i$$

Assuming that we already know q_j for $j \leq k$, it is our goal to determine C_{ik} for all $i \leq k+1$. To do this, we observe that $q_i^H A q_k = C_{ik}$. Thus, we know all values of C_{ik} where $i \leq k$. To determine $C_{k+1,k}$, observe that

$$\underbrace{Aq_k - \sum_{i=1}^{k} C_{ik} q_i}_{\gamma} = C_{k+1,k} q_{k+1} = \underbrace{(Aq_k) - \sum_{i=1}^{k} q_i^H (Aq_k) q_i}_{\gamma} = C_{k+1,k} q_{k+1}$$

Since q_{k+1} is destined to be part of an orthogonal matrix, we know that $C_{k+1,k}$ is $\|\gamma\|$ and that q_{k+1} is $\gamma/\|\gamma\|$.

Observe that the algorithm above unfolds like Gram Schmidt in that we take the next vector Aq_k and then subtract away the projection of this vector onto previous q_i where $i \leq k-1$.

Proposition 3.14. We can compute the eigenvectors corresponding to the k greatest eigenvalues in modulus.

Proof. We found that $Q_n^h A Q_n \cong H$, an upper Hesenberg matrix. Define

$$Q_k = [q_1 \quad \dots q_k]$$

to be the $n \times k$ matrix that consists of the first k Arnoldi vectors. Define

$$U_k = [q_{k+1} \quad \dots q_n]$$

to be the matrix consisting of the remaining k uncomputed vectors. It follows that

$$Q_n^H A Q_n = \begin{bmatrix} Q_k^h \\ U_k^h \end{bmatrix} A \begin{bmatrix} Q_k & U_k \end{bmatrix}$$

□

4 Lecture 14

Nonlinear equations

Problem Q. R Iteration Convergence For which of the following classes of matrices will QR iteration always converge and produce all the eigenvalues?

Select all that apply: Symmetric matrix Diagonal matrix Orthogonal matrix Upper Triangular matrix General matrix with complex eigenvalues

Solution 4.1. Stupid trick question. The answer is a diagonal matrix and an upper triangular matrix. For we don't need to invoke the QR algorithm on these matrices; we already know their eigenvalues (just read the diagonal).

QR is never guaranteed to always converge (ie in a finite number of steps), because there is proveably no algorithm that can give us the eigenvalues for matrices with length at least 5.

Problem K. rylov Subspace Conditioning Given a matrix A with condition number =100 and an initial vector x0, how many additional Krylov vectors can be calculated while ensuring the relative error of each vector remains less than or equal to 1×10−6 assuming IEEE double precision.

Solution 4.2. The answer is 5. The relative error is bounded by the condition number multiplied by the relative input error. At the kth iteration the relative input error is the relative output error of the $k-1$th iteration where the relative input input error of the 0th iteration is given by machine epsilon, ie $\approx 2\times 10^{-16}$. Therefore, relative output error at iteration k is $100^k 2 \times 10^{-16}$, meaning that we can let k be at most 5.

Problem S. uppose A is a general n×n matrix and Q=[q1q2…qk] is an orthogonal matrix with qjKj(A,b), the Krylov subspace associated with A. How many nonzero elements are in the matrix QTAQ?

Solution 4.3. See the notes previously that explain that if we gather the fist k Arnoldi vectors into a matrix Q then Q^TAQ is upper Hesenberg and $k \times k$, meaning it has k^2 non zero entries.

Problem Q. k is the matrix obtained by orthogonalizing the columns of Ck=[bAbA2b…Ak−1b]. Which of the following is equal to QTkb?

Solution 4.4. Q_k will contain column vectors such that only the first vector q_1 is orthonormal with b. In fact that $\langle q_1, b\rangle = \langle q_1, \frac{b}{\|b\|}\|b\|\rangle = \|b\|\langle q_1, \frac{b}{\|b\|}\rangle = \|b\|e_1$.

Problem C. oding question:

Solution 4.5. Here the solution is to recall that once we factorize $K_n = Q_n R_n$ then the upper Hesenberg matrix $K_n^T A K_n$ can be alternatively be expressed as

$$Q_n^T A Q$$

since this simplifies to

$$R_n K_n^{-1} A K R_n$$

which is still upper Hesenberg.

Question 4.6. Why do we favor symmetric problems in terms of condition number?

Remark 4.7. Arnoldi is a way to get eigenvalues especially if you cannot store the entire matrix. Why? Since if we compute $Q_k^T A Q_k$, we get a matrix consisting of the first k ritz eigenvalues, which approximate the k eigenvalues greatest in modulus.

Proposition 4.8. We can compute the SVD of a square matrix:

Proof. The following is a naive way of computing the SVD:

Suppose that we have already obtained the eigenvalues and eigenvectors of A^TA as part of a matrix V. Note that the eigenvectors constitute an eigenbasis, since A^TA is symmetric and thus has an eigenvector basis. It follows that A^TA is diagonalizable. That is we have:

$$A^TA = V\Sigma^2V^T$$

We know that the diagonal matrix can be represented as the square of a matrix, because A^TA is positive semidefinite and, hence, has non-negative eigenvalues.

$$A = U\Sigma V^T \implies U = AV\Sigma^{-1}$$

Indeed, we will find that U is orthogonal, since

$$U^TU = \Sigma^{-1}V^TA^TAV\Sigma^{-1} = I$$

Note that we seem to have assumed that A^TA is full rank.

□

Definition 4.9. A non linear equation $f : \mathbb{R}^n \to \mathbb{R}$ is solved when we find the tuple x such that $f(x) = 0$. If we are given a function $g : \mathbb{R}^n \to \mathbb{R}$ and we want to find the x such that $g(x) = c$, that we can alternatively find the x such that $f(x) = g(x) - c = 0$.

Remark 4.10. We have three tools in order to assert the existence of a solution.

- The intermediate value theorem tells us that if $f(a) < 0$ and $f(b) > 0$ and f is continuous, then there is a c such that $a < c < b$ such that $f(c) = 0$.
- The inverse function theorem tells us that if $f : \mathbb{R}^n \to \mathbb{R}$ and the Jacobian of f is non-singular at a point x, then there is a neighborhood B of $f(x)$ such that for every $y \in B$, there is an x such that $f(x) = y$.
- **Definition 4.11.** A function $g : \mathbb{R}^n \to \mathbb{R}^n$ is a contraction if it holds that for some $0 \leq \alpha < 1$, we have $\|g(x) - g(y)\| \leq \alpha\|x - y\|$.
  - The contraction mapping theorem tells us that if S is a closed set and g a contraction such that $g(S) \subseteq S$, then there is a unique fixed point in S.
    - For if there were two fixed points, x and y, then we would not have $\|g(x) - g(y)\| \leq \alpha\|x - y\|$.

* We often reduce root finding problems of the form $f(x) = 0$ to fixed point problems $g(x) = f(x) + x$.

Proposition 4.12. The condition number of solving a root problem is the same as the condition number of evaluating the inverse function at 0.

Proof.

Lemma 4.13. The condition number of evaluating a function f at x is $|f'(x)|$.

Proof. Suppose that we perturb the input x by h; then the relative difference in the output is $f(x+h) - f(x) = hf'(x)$ using the Taylor approximation. It follows that

$$\left|\frac{f(x+h) - f(x)}{h}\right| = |f'(x)|$$

□

Using the preceding lemma, it follows that the function that we wish to find the absolute condition number of is $x \mapsto f^{-1}(x)$. Note that this function has derivative given by $\frac{1}{f'(f^{-1}(x))}$. Whence the condition number is given by

$$\frac{1}{f'(x^*)}$$

where x^* is the root such that $f(x^*) = 0$.

For this reason, we submitted the proposition above that the condition number is the same as the condition number of evaluating the inverse. □

Definition 4.14. A function has a root of multiplicity m if it holds that

$$f^j(x) = 0$$

where $0 \leq j \leq m - 1$.

Remark 4.15. If a function has a root of high multiplicity then evaluating this root has a high condition number. For the function f will be very flat near the root x (since its derivatives at x are all 0), meaning that f^{-1} will be steep near 0, which implies that small input perturbations will have a high output perturbation.

Definition 4.16. Suppose that a function f outputs an estimate μ_k of some quantity μ at every iteration k. We say that f converges at rate r if

$$\lim_{k\to\infty} \|e_{k+1}\|/\|e_k\|^r \leq C$$

where $e_k = \mu_k - \mu$ and $0 \leq C < \infty$.

Remark 4.17. Suppose that $e_0 < 1$ is a starting error for two processes, one quadratically converging and the other linearly converging. In this case, quadratic convergence will converge faster (in fewer iterations) than linear convergence. For linear iteration will decrease the magnitude of the error vector by a constant factor every, say, k iterations whereas quadratic convergence will decrease the magnitude of the error vector by a factor of 2. Quadratic convergence will converge, to finite precision, in no more than 5 iterations, assuming that

the starting error vector is less in magnitude than 10^0, because as we discussed just now, floating point precision will only allow for accuracy for numbers up to 10^{-16}.

In light of this quick convergence near the solution, it is not meaningful to have operations with convergence rate greater than 2.

Definition 4.18. One of the following criteria for convergence if often employed, although all criterion have their flaws:

- $$|f(x)| < \epsilon$$
  - A very flat function (one with, for example, a root of high multiplicity) may become small at points that are not roots.
- $$\|x_k - x_{k+1}\| < \epsilon$$
  - An algorithm may fail to make progress, accounting for incremental differences.
- $$\frac{\|x_k - x_{k+1}\|}{\|x_k\|} < \epsilon$$
  - The same as above.

5 Lecture 15

Bisection Method

Definition 5.1. The bisection method operates on an interval $[a, b]$ where we assume that $sign\,(f(a)) = -sign\,(f(b))$, from which it follows that there exists a zero in $[a, b]$. We then do the following: Letting $m = a + b/2$, if $f(m)f(a) \geq 0$, we recurse on $[m, b]$ or else on $[a, m]$.

Remark 5.2. At any iteration the distance from either a or b to the root is at most $b - a$. We define $b - a$ to be the error at this iteration. A subsequent iteration will halve the interval, from which we conclude that the error rate is given by

$$e_k \leq e_{k+1}\frac{1}{2}$$

Remark 5.3. Since the magnitude in the error drops by 2^{-1} on every iteration, bisection will terminate in at most 52 iterations, since the error after 52 iterations is $2^{-52} = \epsilon_{mach}$, and we cannot represent any number thereafter. UNRESOLVED – well, shouldn't the relative error have to be 2^{-52} for us to make this conclusion.

Definition 5.4. Assume that there is a fixed point x^* of a smooth function g (continuously differentiable). Assume further that $g'(x^*) < 1$, which implies that $g' < 1$ on a neighborhood of x^*. Then it follows that

$$g(x_k) - g(x^*) = g'(\theta)(x_k - x^*)$$

where $g(x_k) = x_{k+1}$ and θ is some point in between $[x_k, x^*]$. If θ falls into the neigborhood where $g' < 1$, then the equation above is a contraction (here, we use the stronger definition) and it converges linearly.

Remark 5.5. Suppose that $g'(x^*) = 0$. Then using a second order taylor expansion, it holds that $g'(\theta) = g'(x^*) + g''(x^*)(\theta - x^*) \leq g''(x^*)(e_k)$. Here we assume that $\theta > x^*$, meaning that the fixed point is the point closer to ∞. Then it follows that

$$\underbrace{g(x_k) - g(x^*)}_{e_{k+1}} \leq g''(\theta) e_k^2$$

Remark 5.6. The fact that $g'(\delta) = g'(x^\star) + g''(x^\star)(x^\star - \delta)$ implies that as δ approaches x^*, the magnitude of the first derivative drops. In summary if this derivative drops enough that it is 0 at x^*, then convergence is quadratic and if, at least, the derivative is less than 1, then convergence is linear.