A short course on: Preconditioned Krylov subspace methods

Yousef Saad
University of Minnesota
Dept. of Computer Science and Engineering

Universite du Littoral, Jan 19-30, 2005

Outline

Part 1

- Introd., discretization of PDEs
- Sparse matrices and sparsity
- Basic iterative methods (Relaxation..)

Part 2

- Projection methods
- Krylov subspace methods

Part 3

- Preconditioned iterations
- Preconditioning techniques
- Parallel implementations

Part 4

- Eigenvalue problems
- Applications –

INTRODUCTION - MOTIVATION

Origins of Eigenvalue Problems

- **Structural Engineering [$Ku = \lambda Mu$]**
- **Electronic structure calculations [Shrödinger equation..]**
- **Stability analysis [e.g., electrical networks, mechanical system,..]**
- **Bifurcation analysis [e.g., in ¤uid ¤ow]**

➨ **Large sparse eigenvalue problems are among the most demanding calculations (in terms of CPU time) in scienti£c computing.**

New application in information technology

- **Search engines (google) rank web-sites in order to improve searches**
- **The google toolbar on some browsers (http://toolbar.google.com) - gives a measure of relevance of a page.**
- **The problem can be formulated as a Markov chain – Seek the dominant eigenvector**
- **Algorithm used: power method**
- **For details see:**

`http://www.iprcom.com/papers/pagerank/index.html`

The Problem

We consider the eigenvalue problem

$$Ax = \lambda x \text{ or } Ax = \lambda Bx$$

Typically: B is symmetric (semi) positive de£nite, A is symmetric or nonsymmetric

Requirements vary:

- Compute a few λ_i 's with smallest or largest real parts;
- Compute all λ_i's in a certain region of $\mathbb{C}$;
- Compute a few of the dominant eigenvalues;
- Compute all λ_i's.

Types of problems

* **Standard Hermitian (or symmetric real)** $\boxed{Ax = \lambda x}$, $A^H = A$

* **Standard non-Hermitian** $\boxed{Ax = \lambda x}$, $A^H \neq A$

* **Generalized**

$$Ax = \lambda Bx$$

Several distinct sub-cases (B SPD, B SSPD, B singular with large null space, both A and B singular, etc..)

* **Quadratic**

$$(A + \lambda B + \lambda^2 C)x = 0$$

* **Nonlinear**

$$A(\lambda)x = 0$$

EIGENVALUE PROBLEMS – BASICS

DENSE MATRIX CASE

- Background on eigenvalues/ eigenvectors/ Jordan form
- The Schur form
- Perturbation analysis, condition numbers..
- Power method, subspace iteration algorithms
- The QR algorithm
- Practical QR algorithms: use of Hessenberg form and shifts
- The symmetric eigenvalue problem.

Basic de£nitions and properties

A complex scalar λ is called an eigenvalue of a square matrix A if there exists a nonzero vector u in $\mathbb{C}^n$ such that $Au = \lambda u$. The vector u is called an *eigenvector* of A associated with λ. The set of all eigenvalues of A is the 'spectrum' of A. Notation: $\Lambda(A)$.

- $\lambda \in \Lambda(A)$ **iff the columns of** $A - \lambda I$ **are linearly dependent.**
- **... equivalent to saying that its rows are linearly dependent. So: there is a nonzero vector** w **such that**

$$w^H(A - \lambda I) = 0$$

- w^H **is called a left eigenvector of** A **(**u **is a right eigenvector)**
- $\lambda \in \Lambda(A)$ **iff** $\boxed{\det(A - \lambda I) = 0}$

Basic de£nitions and properties (cont.)

- An eigenvalue is a root of the Characteristic polynomial:

$$p_A(\lambda) = \det(A - \lambda I)$$

- So there are n eigenvalues (counted with their multiplicities).

- The multiplicity of these eigenvalues as roots of p_A are called algebraic multiplicities.

- The geometric multiplicity of an eigenvalue λ_i is the number of linearly independent eigenvectors associated with λ_i.

- Geometric multiplicity is $\leq$ algebraic multiplicity.

- An eigenvalue is simple if its (algebraic) multiplicity is one.

- It is semi-simple if its geometric and algebraic multiplicities are equal.

Example: Consider

$$A = \begin{pmatrix} 1 & 2 & -4 \\ 0 & 1 & 2 \\ 0 & 0 & 2 \end{pmatrix}$$

What are the eigenvalues of A? their algebraic multiplicities? their geometric multiplicities? Is one a semi-simple eigenvalue?

Same questions if a_{33} is replaced by one.

- Two matrices A and B are **similar** if there exists a nonsingular matrix X such that

$$B = XAX^{-1}$$

- De£nition: A is **diagonalizable** if it is similar to a diagonal matrix

- THEOREM: A matrix is diagonalizable iff it has n linearly independent eigenvectors
- THEOREM (Schur form): Any matrix is unitarily similar to a triangular matrix, i.e., for any A there exists a unitary matrix Q and an upper triangular matrix R such that

$$A = QRQ^H$$

- Any Hermitian matrix is unitarily similar to a **real diagonal** matrix, (i.e. its Schur form is real diagonal).

Special case: symmetric / Hermitian matrices

- Consider the Schur form of a real symmetric matrix A:

$$A = QRQ^H$$

Since $A^H = A$ then $R = R^H$ ➨

Eigenvalues of A are real

In addition, Q can be taken to be real when A is real.

$$(A - \lambda I)(u + iv) = 0 \rightarrow (A - \lambda I)u = 0 \quad \text{and} \quad (A - \lambda I)v = 0$$

- Can select eigenvectors to be real.

There is an orthonormal basis of eigenvectors of A

The min-max theorem

Label eigenvalues increasingly:

$$\lambda_1 \geq \lambda_2 \geq \cdots \lambda_n$$

The eigenvalues of a Hermitian matrix A are characterized by the relation

$$\lambda_k = \max_{S,\ \dim(S)=k} \quad \min_{x \in S, x \neq 0} \quad \frac{(Ax,x)}{(x,x)}$$

➨ **Consequence:**

$$\lambda_1 = \max_{x \neq 0} (Ax,x)/(x,x) \qquad \lambda_n = \min_{x \neq 0} (Ax,x)/(x,x)$$

The Law of interia

- A matrix A with m negative, z zero, and p positive eigenvalues, has inertia $[m, z, p]$.

Sylvester's Law of inertia: If X is an $n \times n$ nonsingular matrix, then A and $X^T A X$ have the same inertia.

Example: Suppose that $A = LDL^T$ where L is unit lower triangular, and D diagonal. How many negative eigenvalues does A have?

Example: Assume that A is tridiagonal. How many operations are required to determine the number of negative eigenvalues of A?

Example: Devise an algorithm based on the inertia theorem to compute the i-th eigenvalue of a tridiagonal matrix.

Perturbation analysis

- **General questions: If A is perturbed how does an eigenvalue change? How about an eigenvector?**

- **Also: sensitivity of an eigenvalue to perturbations**

THEOREM [Gerschgorin]

$$\forall\, \lambda \in \Lambda(A), \quad \exists\, i \quad \text{such that} \quad |\lambda - a_{ii}| \leq \sum_{\substack{j=1 \\ j \neq i}}^{j=n} |a_{ij}| \,.$$

- **In words: An eigenvalue λ of A is located in one of the closed discs $D(a_{ii}, \rho_i)$ with $\rho_i = \Sigma_{j \neq i} |a_{ij}|$.**

Gerschgorin's theorem - example

- Find a region of the complex plane where the eigenvalues of the following matrix are located:

$$A = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 2 & 0 & 1 \\ -1 & -2 & -3 & 1 \\ \frac{1}{2} & \frac{1}{2} & 0 & -4 \end{pmatrix}$$

- Re£nement: if disks are all disjoint then each of them contains one eigenvalue

- Re£nement: can combine row and column version of the theorem (column version obtained by applying theorem to A^H).

Bauer-Fike theorem

THEOREM [Bauer-Fike] Let $\tilde{\lambda}, \tilde{u}$ be an approximate eigenpair with $\|\tilde{u}\|_2 = 1$, and let $r = A\tilde{u} - \tilde{\lambda}\tilde{u}$ ('residual vector'). Assume A is diagonalizable: $A = XDX^{-1}$, with D diagonal. Then

$$\exists\, \lambda \in \Lambda(A) \quad \text{such that} \quad |\lambda - \tilde{\lambda}| \leq \text{cond}_2(X)\|r\|_2 \,.$$

- Very restrictive result - also not too sharp in general.

- Alternative formulation. If E is a perturbation to A then for any eigenvalue $\tilde{\lambda}$ of $A + E$ there is an eigenvalue λ of A such that:

$$|\lambda - \tilde{\lambda}| \leq \text{cond}_2(X)\|E\|_2 \,.$$

◇Prove this result from the previous one.

Conditioning of Eigenvalues

- Assume that λ is a simple eigenvalue with right and left eigenvectors u and w^H respectively. Consider the matrices:

$$A(t) = A + tE$$

- Eigenvalue $\lambda(t)$, eigenvector $u(t)$.

- Conditioning of λ of A relative to E is the $|d\lambda(t)/dt|$ at $t = 0$.

- Write

$$A(t)u(t) = \lambda(t)u(t)$$

- then multiply both sides to the left by w^H

$$w^H(A + tE)u(t) = \lambda(t)w^H u(t) \quad \rightarrow$$

$$\begin{aligned} \lambda(t)w^H u(t) &= w^H Au(t) + tw^H Eu(t) \\ &= \lambda w^H u(t) + tw^H Eu(t). \end{aligned}$$

Hence,

$$\frac{\lambda(t)-\lambda}{t} w^H u(t) \; = w^H E u(t)$$

- Take the limit at $t = 0$,

$$\lambda'(0) = \frac{w^H E u}{w^H u}$$

- Note: the left and right eigenvectors associated with a simple eigenvalue cannot be orthogonal to each other.

Actual conditioning of an eigenvalue, given a perturbation "in the direction of E" is the modulus of the above quantity.

- In practice, one only has an estimate of $\|E\|$ for some norm

$$|\lambda'(0)| \leq \frac{\|Eu\|_2 \|w\|_2}{|(u,w)|} \leq \|E\|_2 \frac{\|u\|_2 \|w\|_2}{|(u,w)|}$$

De£nition. **The condition number of a simple eigenvalue λ of an arbitrary matrix A is de£ned by**

$$\mathrm{cond}(\lambda) = \frac{1}{\cos\theta(u,w)}$$

in which u and w^H are the right and left eigenvectors, respectively, associated with λ.

Example: **Consider the matrix**

$$A = \begin{pmatrix} -149 & -50 & -154 \\ 537 & 180 & 546 \\ -27 & -9 & -25 \end{pmatrix}$$

► $\Lambda(A) = \{1, 2, 3\}$. **Right and left eigenvectors associated with** $\lambda_1 = 1$**:**

$$u = \begin{pmatrix} 0.3162 \\ -0.9487 \\ 0.0 \end{pmatrix} \quad \text{and} \quad w = \begin{pmatrix} 0.6810 \\ 0.2253 \\ 0.6967 \end{pmatrix}$$

So:

$$\boxed{\text{cond}(\lambda_1) \approx 603.64}$$

- **Perturbing a_{11} to -149.01 yields the spectrum:**

$$\{0.2287, 3.2878, 2.4735\}.$$

- **as expected..**

- **For Hermitian (also normal matrices) every simple eigenvalue is well-conditioned, since $\text{cond}(\lambda) = 1$.**

The power method

- Basic idea is to generate the sequence of vectors $A^k v_0$ where $v_0 \neq 0$ – then normalize.

- Most commonly used normalization: ensure that the largest component of the approximation is equal to one.

ALGORITHM : 1. The Power Method

1. Choose a nonzero initial vector $v^{(0)}$.
2. For $k = 1, 2, \ldots,$ until convergence, Do:
3. $\quad v^{(k)} = \frac{1}{\alpha_k} A v^{(k-1)}$ where
4. $\quad \alpha_k = \mathrm{argmax}_{i=1,\ldots,n} |(A v^{(k-1)})_i|$
5. EndDo

- $\mathrm{argmax}_{i=1,..,n} |x_i| \equiv$ the component x_i with largest modulus

Convergence of the power method

THEOREM Assume that there is one and only one eigenvalue λ_1 of A of largest modulus and that λ_1 is semi-simple. Then either the initial vector v_0 has no component in the invariant subspace associated with λ_1 or the sequence of vectors generated by the algorithm converges to an eigenvector associated with λ_1 and α_k converges to λ_1.

Proof in the diagonalizable case.

- v_k is = vector $A^k v_0$ normalized by a certain scalar $\hat{\alpha}_k$ in such a way that its largest component is 1.

- Decompose the initial vector v_0 as $v_0 = \sum_{i=1}^{p} \gamma_i u_i$ where the u_i's are the eigenvectors associated with the λ_i's, $i = 1, \ldots, n$.

- Note that $A^k u_i = \lambda_i^p u_i$

$$
\begin{aligned}
v_k &= \frac{1}{scaling} \times \sum_{i=1}^{n} \lambda_i^k \gamma_i u_i \\
&= \frac{1}{scaling} \times \left[\lambda_1^k \gamma_1 u_1 + \sum_{i=2}^{n} \lambda_i^k \gamma_i^k u_i \right] \\
&= \frac{1}{scaling'} \times \left[u_1 + \sum_{i=2}^{n} \left(\frac{\lambda_i}{\lambda_1} \right)^k \frac{\gamma_i}{\gamma_1} u_i \right]
\end{aligned}
$$

- Second term inside bracket converges to zero. QED

- Proof suggests that the convergence factor is given by

$$\rho_D = \frac{|\lambda_2|}{|\lambda_1|}$$

where λ_2 is the second largest eigenvalue in modulus.

Example: Consider a 'Markov Chain' matrix of size $n = 55$. Dominant eigenvalues are $\lambda = 1$ and $\lambda = -1$ ➨ the power method applied directly to A fails. (Why?)

➠ We can consider instead the matrix $I + A$ The eigenvalue $\lambda = 1$ is then transformed into the (only) dominant eigenvalue $\lambda = 2$

Iteration	Norm of diff.	Res. norm	Eigenvalue
20	0.639D-01	0.276D-01	1.02591636
40	0.129D-01	0.513D-02	1.00680780
60	0.192D-02	0.808D-03	1.00102145
80	0.280D-03	0.121D-03	1.00014720
100	0.400D-04	0.174D-04	1.00002078
120	0.562D-05	0.247D-05	1.00000289
140	0.781D-06	0.344D-06	1.00000040
161	0.973D-07	0.430D-07	1.00000005

The Shifted Power Method

➨ In previous example shifted A into $B = A + I$ before applying power method. We could also iterate with $B(\sigma) = A + \sigma I$ for any positive σ

Example: With $\sigma = 0.1$ we get the following improvement.

Iteration	Norm of diff.	Res. Norm	Eigenvalue
20	0.273D-01	0.794D-02	1.00524001
40	0.729D-03	0.210D-03	1.00016755
60	0.183D-04	0.509D-05	1.00000446
80	0.437D-06	0.118D-06	1.00000011
88	0.971D-07	0.261D-07	1.00000002

➤ **Question:** What is the best shift-of-origin σ to use?

When all eigenvalues are real and such that

$$\lambda_1 > \lambda_2 \geq \lambda_2 \geq \cdots \geq \lambda_n,$$

then the value of σ which yields the best convergence factor is:

$$\sigma_{opt} = \frac{\lambda_2 + \lambda_n}{2}$$

Inverse Iteration

Observation: The eigenvectors of A and A^{-1} are identical.

- Idea: use the power method on A^{-1}.
- Will compute the eigenvalues closest to zero.
- Shift-and-invert Use power method on $\boxed{(A - \sigma I)^{-1}}$. ➤ will compute eigenvalues closest to σ.
- Advantages: fast convergence in general.
- Drawbacks: need to factor A (or $A - \sigma I$) into LU..

Subspace iteration

- Generalizes the power method

ALGORITHM : 2. ***Orthogonal iteration***

1. **Start:** $Q_0 = [q_1, \ldots, q_m]$
2. **Iterate:** ***Until convergence do,***
3. $X := AQ_{k-1}$
4. $X = Q_k R$ ***(QR factorization)***
5. ***EndDo***

- Normalization in step 4 is similar to the scaling used in the power method.

- Improvement: normalize only once in a while.

ALGORITHM : 3. Subspace Iteration with Projection

Start: ***Choose*** $Q_0 = [q_0, \ldots, q_m]$

Iterate: ***For*** $k = 1, \ldots,$ ***until convergence do:***

Compute $\hat{Z} = AQ_{k-1}$.

$\hat{Z} = ZR_Z$ ***(QR factorization)***

$B = Z^H AZ$

Compute the Schur factorization $B = YRY^H$

$Q_k = ZY$

EndDo

- **Again: no need to orthogonalize + project at each step.**

- **Assume** $|\lambda_1| \geq |\lambda_2| \geq \cdots |\lambda_m| > |\lambda_{m+1}| \geq \cdots \geq |\lambda_n|$**, then convergence rate for** λ_1 **is (generally)**

$$|\lambda_{m+1}/\lambda_1|$$

The QR algorithm

- The most common method for solving small (dense) eigenvalue problems. The basic algorithm:

ALGORITHM : 4. *QR without shifts*

1. Until Convergence Do:

2. Compute the QR factorization $A = QR$

3. Set $A := RQ$

4. EndDo

- "Until Convergence" means "Until A becomes close enough to an upper triangular matrix"

- **Note:** $A_{new} = RQ = Q^H(QR)Q = Q^H AQ$
- A_{new} **is similar to** A **throughout the algorithm .**
- **Above basic algorithm is never used in practice. Two variations:**

(1) use shift of origin and

(2) Transform A **into Hessenberg form..**

Practical QR: Shifts of origin

Observation: (from theory): Last row converges fastest. Convergence is dictated by $\frac{|\lambda_n|}{|\lambda_{n-1}|}$

- We will now consider only the real symmetric case.
- Eigenvalues are real.
- $A^{(k)}$ remains symmetric throughout process.
- As k goes to in£nity the last column and row (except $a_{nn}^{(k)}$) converge to zero quickly.,,
- and $a_{nn}^{(k)}$ converges to lowest eigenvalue.

$$A^{(k)} = \left(\begin{array}{ccccc|c} . & . & . & . & . & a \\ . & . & . & . & . & a \\ . & . & . & . & . & a \\ . & . & . & . & . & a \\ . & . & . & . & . & a \\ \hline a & a & a & a & a & a \end{array}\right)$$

- **Idea: Apply QR algorithm to $A^{(k)} - \mu I$ with $\mu = a_{nn}^{(k)}$. Note: eigenvalues of $A^{(k)} - \mu I$ are shifted by μ, and eigenvectors are the same.**

ALGORITHM : 5. *QR with shifts*

1. Until row $a_{in}, 1 \leq i < n$ *converges to zero DO:*

2. *Obtain next shift (e.g.* $\mu = a_{nn}$*)*

3. $A - \mu I = QR$

5. *Set* $A := RQ + \mu I$

6. EndDo

➨ **Convergence is cubic at the limit! [for symmetric case]**

Result of algorithm:

$$A^{(k)} = \left(\begin{array}{ccccc|c} . & . & . & . & . & 0 \\ . & . & . & . & . & 0 \\ . & . & . & . & . & 0 \\ . & . & . & . & . & 0 \\ . & . & . & . & . & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & \lambda_n \end{array}\right)$$

Next step: de¤ate, i.e., apply above algorithm to $(n-1) \times (n-1)$ upper triangular matrix.

Practical QR: Use of the Hessenberg Form

Recall: Upper Hessenberg matrix is such that

$$a_{ij} = 0 \text{ for } j < i - 1$$

Observation: The QR algorithm preserves Hessenberg form (tridiagonal form in symmetric case). Results in substantial savings.

- **1-st step: reduce A to Hessenberg form. Then (2nd step) apply QR algorithm to resulting matrix.**

- **It is easy to adapt the Householder factorization to reduce a matrix into Hessenberg form – [similarity transformation]**

- **Consider the £rst step only on a 6×6 matrix.**

- We want $H_1 A H_1^T = H_1 A H_1$ to have the form:

$$\begin{pmatrix} \star & \star & \star & \star & \star & \star \\ \star & \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star & \star \\ 0 & \star & \star & \star & \star & \star \end{pmatrix}$$

- Choose a w in $H_1 = I - 2ww^T$ so that $(H_1 A)[2:n, 1] = 0$
- Apply to left $B = H_1 A$. Then apply to right $A_1 = B H_1$.

Observation: the Householder matrix H_1 which transforms the column $A(:, 1)$ into e_1 works only on rows 2 to n. When applying H_1^T to the right of $B = H_1 A$, only columns 2 to n will be altered

- 1st column retains the same pattern (zeros below row 2)

QR for Hessenberg matrices

- Need the "implicit Q theorem"

Suppose that $Q^T A Q$ is an unreduced upper Hessenberg matrix. Then columns 2 to n of Q are determined uniquely (up to signs) by the first column of Q.

Implication: In order to compute $A_{i+1} = Q_i^T A Q_i$ we can:

- Compute the first column of Q_i [easy: = scalar $\times A(:,1)$]
- Choose other columns so Q_i = unitary, and A_{i+1} = Hessenberg.

Example: With $n = 6$:

$$A = \begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

1. Choose $G_1 = G(1, 2, \theta_1)$ so that $(G_1 A)_{21} = 0$

$$\Rightarrow \quad A_1 = G_1^T A G_1 = \begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ + & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

2. Choose $G_2 = G(2,3,\theta_2)$ so that $(G_2 A_1)_{31} = 0$

$$\Rightarrow \quad A_2 = G_2^T A_1 G_2 = \begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & + & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

3. Choose $G_3 = G(3,4,\theta_3)$ so that $(G_3 A_2)_{42} = 0$

$$\Rightarrow \quad A_3 = G_3^T A_2 G_3 = \begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & + & * & * \end{pmatrix}$$

4. Choose $G_4 = G(4,5,\theta_4)$ so that $(G_4 A_3)_{53} = 0$

$$\blacktriangleright \quad A_4 = G_4^T A_3 G_4 = \begin{pmatrix} * & * & * & * & * \\ * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

- **Process known as "Bulge chasing"**
- **Similar idea for the symmetric (tridiagonal) case**

The QR algorithm for symmetric matrices

- **Most important method used : reduce to tridiagonal form and apply the QR algorithm with shifts.**

- **Householder transformation to Hesseenberg form yields a tridiagonal matrix because**

$$HAH^T = A_1$$

is symmetric and also of Hessenberg form ➨ it is tridiagonal symmetric.

Tridiagonal form is preserved by QR similarity transformation

Practical method

- How to implement the QR algorithm with shifts?

- It is best to use Givens rotations – can do a shifted QR step without explicitly shifting the matrix..

- Two most popular shifts:

  $s = a_{nn}$ and $s =$ smallest e.v. of $A(n-1:n, n-1:n)$

THE SINGULAR VALUE DECOMPOSITION

- The SVD – existence - properties.
- Pseudo-inverses and the SVD
- Use of SVD for least-squares problems
- Applications of the SVD

The Singular Value Decomposition (SVD)

For any real $n \times m$ matrix A there exists orthogonal matrices $U \in \mathbb{R}^{n \times n}$ and $V \in \mathbb{R}^{m \times m}$ such that

$$A = U \Sigma V^T$$

where Σ is a diagonal matrix with nonnegative diagonal entries.

$$\sigma_{11} \geq \sigma_{22} \geq \cdots \sigma_{pp} \geq 0 \text{ with } p = \min(m, n)$$

- **The σ_{ii} are called singular values of A. Denoted simply by σ_i.**

Proof: **[one among many!] Let $\sigma_1 = \|A\|_2 = \max_{x, \|x\|_2 = 1} \|Ax\|_2$**

There exists a pair of unit vectors v_1, u_1 such that

$$Av_1 = \sigma_1 u_1$$

- Complete v_1 into an orthonormal basis of $\mathbb{R}^m$

$$V \equiv [v_1, V_2] = m \times m \quad \text{unitary}$$

- Complete u_1 into an orthonormal basis of $\mathbb{R}^m$

$$U \equiv [u_1, U_2] = n \times n \quad \text{unitary}$$

- Then, it is easy to show that

$$AV = U \times \begin{pmatrix} \sigma_1 & w^T \\ 0 & B \end{pmatrix} \quad \rightarrow \quad U^T A V = \begin{pmatrix} \sigma_1 & w^T \\ 0 & B \end{pmatrix} \equiv A_1$$

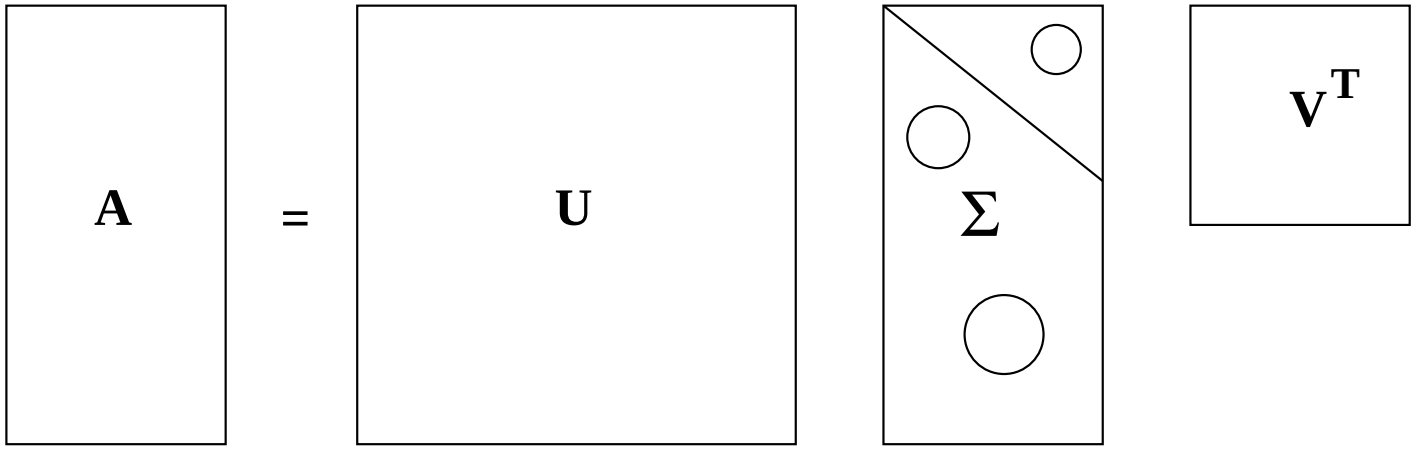
- Observe that

$$\left\| A_1 \begin{pmatrix} \sigma_1 \\ w \end{pmatrix} \right\|_2 \geq \sigma_1^2 + \|w\|^2 = \sqrt{\sigma_1^2 + \|w\|^2} \left\| \begin{pmatrix} \sigma_1 \\ w \end{pmatrix} \right\|_2$$

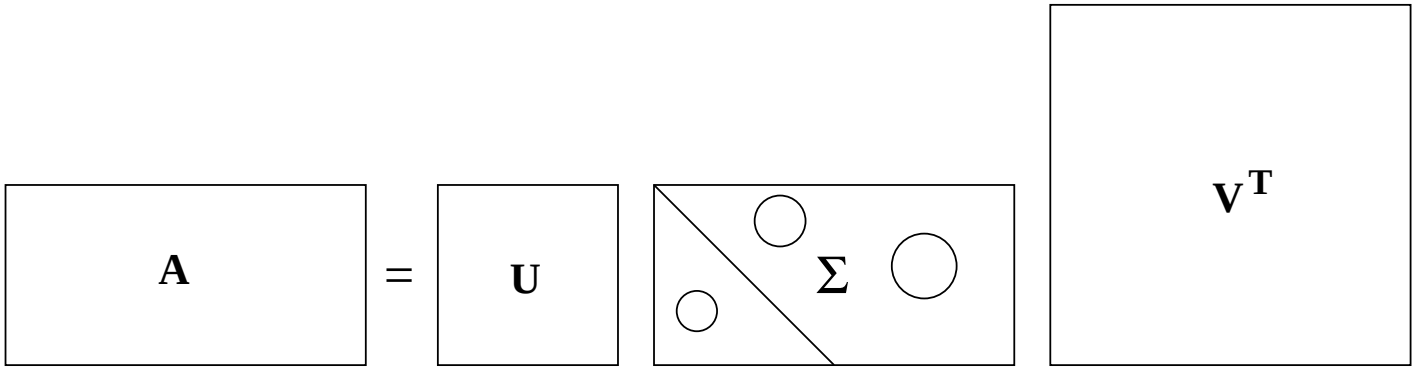
- This shows that w must be zero [why?]

- Complete the proof by an induction argument.

Case 1:



Case 2:



The "thin" SVD

- Consider the Case-1. It can be rewritten as

$$A = [U_1 U_2] \begin{pmatrix} \Sigma_1 \\ 0 \end{pmatrix} V^T$$

Which gives:

$$A = U_1 \Sigma_1 V^T$$

where U_1 is $n \times m$ (same shape as A), and Σ_1 and V are $m \times m$

- referred to as the "thin" SVD. Important in practice.

◇Show how to obtain the thin SVD from the QR factorization of A and the SVD of an $m \times m$ matrix

Some properties. Assume that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0 \text{ and } \sigma_{r+1} = \cdots = \sigma_p = 0$$

Then:

- $rank(A) = r$ = number of nonzero singular values.
- $Ran(A) = span\{u_1, u_2, \ldots, u_r\}$
- $Null(A) = span\{v_{r+1}, v_{r+2}, \ldots, v_m\}$
- The matrix A admits the SVD expansion:

$$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T$$

Properties of the SVD (continued)

- $\|A\|_2 = \sigma_1$ **= largest singular value**
- $\|A\|_F = \left(\sum_{i=1}^{r} \sigma_i^2\right)^{1/2}$
- **When** A **is an** $n \times n$ **nonsingular matrix then** $\|A^{-1}\|_2 = 1/\sigma_n$ **= inverse of smallest s.v.**

Let $k < r$ **and**

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$$

then

$$\min_{rank(B)=k} \|A - B\|_2 = \|A - A_k\|_2 = \sigma_{k+1}$$

De£ne the $r \times r$ matrix

$$\Sigma_1 = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$$

- **Let $A \in \mathbb{R}^{n \times m}$ and consider now $A^T A$ (which is of size $m \times m$)**

$$A^T A = V \Sigma^T \Sigma V^T \qquad \rightarrow \qquad A^T A = V \underbrace{\begin{pmatrix} \Sigma_1^2 & 0 \\ 0 & 0 \end{pmatrix}}_{m \times m} V^T$$

- **This gives the spectral decomposition of $A^T A$. Similarly, U gives the eigenvectors of AA^T.**

$$AA^T = U \underbrace{\begin{pmatrix} \Sigma_1^2 & 0 \\ 0 & 0 \end{pmatrix}}_{n \times n} U^T$$

Important: **$A^T A = V D_1 V^T$ and $AA^T = U D_2 U^T$ give the SVD factors U, V up to signs!**

◇Compute the singular value decomposition the matrix:

$$A = \begin{pmatrix} 1 & 0 & 2 & 0 \\ 0 & 0 & -2 & 1 \end{pmatrix}$$

◇Find the matrix B of rank 1 which is the closest to the above matrix in the 2-norm sense.

◇What is the pseudo-inverse of A? What is the pseudo-inverse of B?

◇Find the vector x of smallest norm which minimizes $\|b - Ax\|_2$ with $b = (1, 1)^T$

◇Find the vector x of smallest norm which minimizes $\|b - Bx\|_2$ with $b = (1, 1)^T$

Pseudo-inverse of an arbitrary matrix

The pseudo-inverse of A is given by

$$A^{\dagger} = V \begin{pmatrix} \Sigma_1^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^T$$

Moore-Penrose conditions: **The pseudo inverse of a matrix is uniquely determined by these four conditions:**

(1) $AXA = A$ **(2)** $XAX = X$

(3) $(AX)^H = AX$ **(4)** $(XA)^H = XA$

► **In the full-rank overdetermined case,** $A^{\dagger} = (A^T A)^{-1} A^T$

Least-squares problems and the SVD

➨ **SVD can give much information about solving overdetermined and underdetermined linear systems –**

Let A be an $n \times m$ matrix and $A = U\Sigma V$ its SVD with $r = rank(A)$, $V = [v_1, \ldots, v_m]$ $U = [u_1, \ldots, u_n]$. Then

$$x_{LS} = \sum_{i=1}^{r} \frac{u_i^T b}{\sigma_i} v_i$$

minimizes $\|b - Ax]\|_2$ and has the smallest 2-norm among all possible minimizers. In addition,

$$\rho_{LS} \equiv \|b - Ax_{LS}\|_2 = \|z\|_2 \quad \text{with} \quad z = [u_{r+1}, \ldots, u_n]^T b$$

Least-squares problems and pseudo-inverses

- **A restatement of the £rst part of the previous result:**

Consider the general linear least-squares problem

$$\min_{x \in S} \|x\|_2 \qquad S = \{x \in \mathbb{R}^m \mid \|b - Ax\|_2 \min\}$$

This problem always has a unique solution given by

$$x = A^{\dagger} b$$

Ill-conditioned systems and the SVD

- Let A be $n \times n$ (square matrix) and $A = U\Sigma V^T$ its SVD
- Solution of $Ax = b$ is $x = A^{-1}b = \sum_{i=1}^{n} \frac{u_i^T b}{\sigma_i} v_i$
- When A is very ill-conditioned, it may have many small singular values. The division by these small σ_i's will amplify any noise in the data. Result: solution may be meaningless.
- Remedy: use regularization, i.e., truncate the SVD by only keeping the $\sigma_i's$ that are larger than a threshold τ.
- This gives the truncated SVD solution (SVD regularization:)

$$x_{TSVD} = \sum_{\sigma_i \geq \tau} \frac{u_i^T b}{\sigma_i} v_i$$

- Many applications [e.g., Image processing,..]

Numerical rank and the SVD

- Assume that the original matrix A is exactly of rank k.
- The computed SVD of A will be the SVD of a nearby matrix $A + E$.
- Easy to show that $|\hat{\sigma}_i - \sigma_i| \leq \alpha \, \sigma_1 \mathrm{eps}$
- Result: zero singular values will yield small computed singular values
- Determining the "numerical rank:" treat singular values below a certain threshold δ as zero. Practical problem : need to set δ.

LARGE SPARSE EIGENVALUE PROBLEMS

General Tools for Solving Large Eigen-Problems

- **Projection techniques – Arnoldi, Lanczos, Subspace Iteration;**
- **Preconditioninings: shift-and-invert, Polynomials, ...**
- **De¤ation and restarting techniques**

Good computational codes combine these three ingredients

A few popular solution Methods

- **Subspace Iteration [Now less popular – sometimes used for validation]**
- **Arnoldi's method (or Lanczos) with polynomial acceleration [Stiefel '58, Rutishauser '62, YS '84,'85, Sorensen '89,...]**
- **Shift-and-invert and other preconditioners. [Use Arnoldi or Lanczos for $(A - \sigma I)^{-1}$.]**
- **Davidson's method and variants, Generalized Davidosn's method [Morgan and Scott, 89], Jacobi-Davidsion**
- **Emerning method: Automatic Multilevel Substructuring (AMLS).**

Projection Methods for Eigenvalue Problems

General formulation:

Projection method onto K orthogonal to L

- **Given: Two subspaces K and L of same dimension.**
- **Find: $\tilde{\lambda}, \tilde{u}$ such that**

$$\tilde{\lambda} \in \mathbb{C}, \tilde{u} \in K; \quad (\tilde{\lambda} I - A)\tilde{u} \perp L$$

Two types of methods:

Orthogonal projection methods: situation when $L = K$.

Oblique projection methods: When $L \neq K$.

Rayleigh-Ritz projection

Given: a subspace X known to contain good approximations to eigenvectors of A.

Question: How to extract good approximations to eigenvalues/ eigenvectors from this subspace?

Answer: Rayleigh Ritz process.

Let $Q = [q_1, \ldots, q_m]$ an orthonormal basis of X. Then write an approximation in the form $\tilde{u} = Qy$ and obtain y by writing

$$Q^H(A - \tilde{\lambda}I)\tilde{u} = 0$$

$\blacktriangleright$ $Q^H A Q y = \tilde{\lambda} y$

Procedure:

1. Obtain an orthonormal basis of X
2. Compute $C = Q^H AQ$ (an $m \times m$ matrix)
3. Obtain Schur factorization of C, $C = YRY^H$
4. Compute $\tilde{U} = QY$

Property: if X is (exactly) invariant, then procedure will yield exact eigenvalues and eigenvectors.

Proof: Since X is invariant, $(A - \tilde{\lambda}I)u = Qz$ for a certain z. $Q^H Qz = 0$ implies $z = 0$ and therefore $(A - \tilde{\lambda}I)u = 0$.

➨ Can use this procedure in conjunction with the subspace obtained from subspace iteration algorithm

Subspace Iteration

- **Original idea:** projection technique onto a subspace if the form

$Y = A^k X$

- In practice: Replace A^k by suitable polynomial [Chebyshev]

Advantages:
- Easy to implement (in symmetric case);
- Easy to analyze;

Disadvantage: Slow.

- Often used with polynomial acceleration: $A^k X$ replaced by $C_k(A)X$.

Typically C_k = Chebyshev polynomial.

Algorithm: Subspace Iteration with Projection

1. **Start:** Choose an initial system of vectors $X = [x_0, \ldots, x_m]$ and an initial polynomial C_k.

2. **Iterate:** Until convergence do:

   (a) Compute $\hat{Z} = C_k(A)X_{old}$.

   (b) Orthonormalize $\hat{Z}$ into Z.

   (c) Compute $B = Z^H A Z$ and use the QR algorithm to compute the Schur vectors $Y = [y_1, \ldots, y_m]$ of B.

   (d) Compute $X_{new} = ZY$.

   (e) Test for convergence. If satis£ed stop. Else select a new polynomial $C'_{k'}$ and continue.

THEOREM: Let $S_0 = span\{x_1, x_2, \ldots, x_m\}$ and assume that S_0 is such that the vectors $\{Px_i\}_{i=1,\ldots,m}$ are linearly independent where P is the spectral projector associated with $\lambda_1, \ldots, \lambda_m$. Let $\mathcal{P}_k$ the orthogonal projector onto the subspace $S_k = span\{X_k\}$. Then for each eigenvector u_i of A, $i = 1, \ldots, m$, there exists a unique vector s_i in the subspace S_0 such that $Ps_i = u_i$. Moreover, the following inequality is satis£ed

$$\|(I - \mathcal{P}_k)u_i\|_2 \leq \|u_i - s_i\|_2 \left(\left| \frac{\lambda_{m+1}}{\lambda_i} \right| + \epsilon_k \right)^k , \qquad (1)$$

where ϵ_k tends to zero as k tends to in£nity.

KRYLOV SUBSPACE METHODS

KRYLOV SUBSPACE METHODS

Principle: Projection methods on Krylov subspaces, i.e., on

$$K_m(A, v_1) = \text{span}\{v_1, Av_1, \cdots, A^{m-1}v_1\}$$

- probably the most important class of projection methods [for linear systems and for eigenvalue problems]
- many variants exist depending on the subspace L.

Properties of K_m. Let $\mu =$ deg. of minimal polynom. of v. Then,

- $K_m = \{p(A)v | p =$ polynomial of degree $\leq m-1\}$
- $K_m = K_\mu$ for all $m \geq \mu$. Moreover, K_μ is invariant under A.
- $dim(K_m) = m$ iff $\mu \geq m$.

ARNOLDI'S ALGORITHM

- **Goal: to compute an orthogonal basis of K_m.**
- **Input: Initial vector v_1, with $\|v_1\|_2 = 1$ and m.**

ALGORITHM : 6. ***Arnoldi's procedure***

For $j = 1, ..., m$ ***do***

Compute $w := Av_j$

For $i = 1, \ldots, j$, ***do*** $\begin{cases} h_{i,j} := (w, v_i) \\ w := w - h_{i,j} v_i \end{cases}$

$h_{j+1,j} = \|w\|_2$; $\qquad v_{j+1} = w/h_{j+1,j}$

End

Result of Arnoldi's algorithm

Let

$$\overline{H}_m = \begin{pmatrix} x & x & x & x & x \\ x & x & x & x & x \\ & x & x & x & x \\ & & x & x & x \\ & & & x & x \\ & & & & x \end{pmatrix} \qquad H_m = \begin{pmatrix} x & x & x & x & x \\ x & x & x & x & x \\ & x & x & x & x \\ & & x & x & x \\ & & & x & x \end{pmatrix}$$

1. $V_m = [v_1, v_2, ..., v_m]$ **orthonormal basis of** K_m**.**

2. $AV_m = V_{m+1}\overline{H}_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^T$

3. $V_m^T A V_m = H_m \equiv \overline{H}_m -$ **last row.**

Appliaction to eigenvalue problems

- Write approximate eigenvector as $\tilde{u} = V_m y$ + Galerkin condition

$$(A - \tilde{\lambda} I) V_m y \;\perp\; \mathcal{K}_m \rightarrow V_m^H (A - \tilde{\lambda} I) V_m y = 0$$

- Approximate eigenvalues are eigenvalues of H_m

$$H_m y_j = \tilde{\lambda}_j y_j$$

Associated approximate eigenvectors are

$$\tilde{u}_j = V_m y_j$$

Typically a few of the outermost eigenvalues will converge £rst.

Restarted Arnoldi

In practice: Memory requirement of algorithm implies restarting is necessary

ALGORITHM : 7. **Restarted Arnoldi (computes rightmost eigenpair)**

1. **Start:** ***Choose an initial vector*** v_1 ***and a dimension*** m.
2. **Iterate:** ***Perform*** m ***steps of Arnoldi's algorithm.***
3. **Restart:** ***Compute the approximate eigenvector*** $u_1^{(m)}$
4. ***associated with the rightmost eigenvalue*** $\lambda_1^{(m)}$.
5. ***If satis£ed stop, else set*** $v_1 \equiv u_1^{(m)}$ ***and goto 2.***

Example:

Small Markov Chain matrix [Mark(10) , dimension = 55]. Restarted Arnoldi procedure for computing the eigenvector associated with the eigenvalue with algebraically largest real part. We use $m = 10$.

m	$\Re(\lambda)$	$\Im(\lambda)$	Res. Norm
10	0.9987435899D+00	0.0	0.246D-01
20	0.9999523324D+00	0.0	0.144D-02
30	0.1000000368D+01	0.0	0.221D-04
40	0.1000000025D+01	0.0	0.508D-06
50	0.9999999996D+00	0.0	0.138D-07

Restarted Arnoldi (cont.)

- Can be generalized to more than *one* eigenvector :

$$v_1^{(new)} = \sum_{i=1}^{p} \rho_i u_i^{(m)}$$

- However: often does not work well – (hard to £nd good coef£-cients ρ_i's)

- Alternative : compute eigenvectors (actually Schur vectors) one at a time.

- Implicit de¤ation.

Hermitian case: The Lanczos Algorithm

- **The Hessenberg matrix becomes tridiagonal :**

$$A = A^H \quad \text{and} \quad V_m^H A V_m = H_m \quad \rightarrow H_m = H_m^H$$

- **We can write**

$$H_m = \begin{pmatrix} \alpha_1 & \beta_2 & & & & \\ \beta_2 & \alpha_2 & \beta_3 & & & \\ & \beta_3 & \alpha_3 & \beta_4 & & \\ & & . & . & . & \\ & & & . & . & . \\ & & & & \beta_m & \alpha_m \end{pmatrix} \tag{2}$$

- **Consequence: three term recurrence**

$$\boxed{\beta_{j+1} v_{j+1} = A v_j - \alpha_j v_j - \beta_j v_{j-1}}$$

ALGORITHM : 8. *Lanczos*

1. *Choose an initial vector* v_1 *of norm unity. Set* $\beta_1 \equiv 0, v_0 \equiv 0$
2. *For* $j = 1, 2, \ldots, m$ *Do:*
3. $w_j := Av_j - \beta_j v_{j-1}$
4. $\alpha_j := (w_j, v_j)$
5. $w_j := w_j - \alpha_j v_j$
6. $\beta_{j+1} := \|w_j\|_2$. *If* $\beta_{j+1} = 0$ *then Stop*
7. $v_{j+1} := w_j / \beta_{j+1}$
8. *EndDo*

Hermitian matrix + Arnoldi $\longrightarrow$ Hermitian Lanczos

- In theory v_i's de£ned by 3-term recurrence are orthogonal.
- However: in practice severe loss of orthogonality;

Observation [Paige, 1981]: **Loss of orthogonality starts suddenly, when the £rst eigenpair has converged. It is a sign of loss of linear indedependence of the computed eigenvectors. When orthogonality is lost, then several the copies of the same eigenvalue start appearing.**

Reorthogonalization

- **Full reorthogonalization – reorthogonalize v_{j+1} against all previous v_i's every time.**
- **Partial reorthogonalization – reorthogonalize v_{j+1} against all previous v_i's only when needed [Parlett & Simon]**
- **Selective reorthogonalization – reorthogonalize v_{j+1} against computed eigenvectors [Parlett & Scott]**
- **No reorthogonalization – Do not reorthogonalize - but take measures to deal with 'spurious' eigenvalues. [Cullum & Willoughby]**

LANCZOS BIORTHOGONALIZATION

The Lanczos biorthogonalization ($A^H \neq A$)

ALGORITHM : 9. **The Lanczos Bi-Orthogonalization Procedure**

1. **Choose** v_1, w_1 **such that** $(v_1, w_1) = 1$. **Set** $\beta_1 = \delta_1 \equiv 0$, $w_0 = v_0 \equiv 0$
2. **For** $j = 1, 2, \ldots, m$ **Do:**
3. $\alpha_j = (Av_j, w_j)$ [$\alpha_j = (Av_j - \beta_j v_{j-1}, w_j)$]
4. $\hat{v}_{j+1} = Av_j - \alpha_j v_j - \beta_j v_{j-1}$ [$\hat{v}_{j+1} = (Av_j - \beta_j v_{j-1}) - \alpha_j v_j$]
5. $\hat{w}_{j+1} = A^H w_j - \bar{\alpha}_j w_j - \delta_j w_{j-1}$ [$\hat{w}_{j+1} = (A^H w_j - \delta_j w_{j-1}) - \bar{\alpha}_j w_j$]
6. $\delta_{j+1} = |(\hat{v}_{j+1}, \hat{w}_{j+1})|^{1/2}$. **If** $\delta_{j+1} = 0$ **Stop**
7. $\beta_{j+1} = (\hat{v}_{j+1}, \hat{w}_{j+1})/\delta_{j+1}$
8. $w_{j+1} = \hat{w}_{j+1}/\bar{\beta}_{j+1}$
9. $v_{j+1} = \hat{v}_{j+1}/\delta_{j+1}$
10. **EndDo**

- **Builds a pair of biorthogonal bases for the two subspaces**

$$\mathcal{K}_m(A, v_1) \quad \text{and} \quad \mathcal{K}_m(A^H, w_1)$$

- **Many choices for $\delta_{j+1}, \beta_{j+1}$ in lines 7 and 8. Only constraint:**

$$\delta_{j+1}\beta_{j+1} = (\hat{v}_{j+1}, \hat{w}_{j+1})$$

Let

$$T_m = \begin{pmatrix} \alpha_1 & \beta_2 & & & \\ \delta_2 & \alpha_2 & \beta_3 & & \\ & . & . & . & \\ & & \delta_{m-1} & \alpha_{m-1} & \beta_m \\ & & & \delta_m & \alpha_m \end{pmatrix}.$$

- $v_i \in \mathcal{K}_m(A, v_1)$ **and** $w_j \in \mathcal{K}_m(A^T, w_1)$**.**

If the algorithm does not break down before step m, then the vectors $v_i, i = 1, \ldots, m$, and $w_j, j = 1, \ldots, m$, are biorthogonal, i.e.,

$$(v_j, w_i) = \delta_{ij} \quad 1 \leq i,\ j \leq m\ .$$

Moreover, $\{v_i\}_{i=1,2,\ldots,m}$ is a basis of $\mathcal{K}_m(A, v_1)$ and $\{w_i\}_{i=1,2,\ldots,m}$ is a basis of $\mathcal{K}_m(A^H, w_1)$ and

$$\begin{aligned} AV_m &= V_m T_m + \delta_{m+1} v_{m+1} e_m^H, \\ A^H W_m &= W_m T_m^H + \bar{\beta}_{m+1} w_{m+1} e_m^H, \\ W_m^H A V_m &= T_m\ . \end{aligned}$$

➤ If θ_j, y_j, z_j are, respectively an eigenvalue of T_m, with associated right and left eigenvectors y_j and z_j respectively, then corresponding approximations for A are

Ritz value	Right Ritz vector	Left Ritz vector
θ_j	$V_m y_j$	$W_m z_j$

[Note: terminology is abused slightly - Ritz values and vectors normally refer to Hermitian cases.]

Advantages and disadvantages

Advantages:

- Nice three-term recurrence – requires little storage in theory.
- Computes left and a right eigenvectors at the same time

Disadvantages:

- Algorithm can breakdown or nearly breakdown.
- Convergence not too well understood. Erratic behavior
- Not easy to take advantage of the tridiagonal form of T_m.

Look-ahead Lanczos

Algorithm breaks down when:

$$(\hat{v}_{j+1}, \hat{w}_{j+1}) = 0$$

Three distinct situations.

- **'lucky breakdown' when either $\hat{v}_{j+1}$ or $\hat{w}_{j+1}$ is zero. In this case, eigenvalues of T_m are eigenvalues of A.**
- **$(\hat{v}_{j+1}, \hat{w}_{j+1}) = 0$ but of $\hat{v}_{j+1} \neq 0$, $\hat{w}_{j+1} \neq 0 \longrightarrow$ serious breakdown. Often possible to bypass the step (+ a few more) and continue the algorithm. If this is not possible then we get an ...**
- **... Incurable break-down. [very rare]**

Look-ahead Lanczos algorithms deal with the second case. See Parlett 80, Freund and Nachtigal '90.... Main idea: when break-down occurs, skip the computation of v_{j+1}, w_{j+1} and de£ne v_{j+2}, w_{j+2} from v_j, w_j. For example by orthogonalizing $A^2 v_j$ ... Can de£ne v_{j+1} somewhat arbitrarily as $v_{j+1} = Av_j$. Similarly for w_{j+1}.

➨ Drawbacks: (1) projected problem no longer tridiagonal (2) dif£-cult to know what constitutes near-breakdown.

DEFLATION

De¤ation

- Very useful in practice.
- Different forms: locking (subspace iteration), selective orthogonalization (Lanczos), Schur de¤ation, ...

A little background

Consider Schur canonical form

$$A = URU^H$$

where U is a (complex) upper triangular matrix.

- **Vector columns $u_1, \ldots, u_n$ called Schur vectors.**

- **Note: Schur vectors depend on each other, and on the order of the eigenvalues**

Wiedlandt De¤ation: **Assume we have computed a right eigenpair λ_1, u_1. Wielandt de¤ation considers eigenvalues of**

$$A_1 = A - \sigma u_1 v^H$$

Note:

$$\Lambda(A_1) = \{\lambda_1 - \sigma, \lambda_2, \ldots, \lambda_n\}$$

Wielandt de¤ation preserves u_1 as an eigenvector as well all the left eigenvectors not associated with λ_1.

➨ **An interesting choice for v is to take simply $v = u_1$. In this case Wielandt de¤ation preserves Schur vectors as well.**

- **It is possible to apply this procedure successively:**

ALGORITHM : 10 ▪ ***Explicit De¤ation***

1. $A_0 = A$
2. ***For*** $j = 0 \ldots \mu - 1$ ***Do:***
3. ***Compute a dominant eigenvector of*** A_j
4. ***De£ne*** $A_{j+1} = A_j - \sigma_j u_j u_j^H$
5. ***End***

- **Computed $u_1, u_2., ..$ form a set of Schur vectors for A.**

- **Alternative: implicit de¤ation (within a procedure such as Arnoldi).**

De¤ated Arnoldi: When £rst eigenvector converges, we freeze it as the £rst vector of $V_m = [v_1, v_2, \ldots, v_m]$. Arnoldi starts working at column v_2. Orthogonalization is still done against $v_1, ..., v_j$ at step j. Each new converged eigenvector will be added to the 'locked' set of eigenvectors.

For $k = 1, \ldots . NEV$ do: /* Eigenvalue loop */

1. For $j = k, k+1, ..., m$ do: /* Arnoldi loop*/

- **Compute $w := Av_j$.**
- **Orthonormalize w against $v_1, v_2, \ldots, v_j \ \rightarrow \ v_{j+1}$**

2. Compute next approximate eigenpair $\tilde{\lambda}, \tilde{u}$.

3. Orthonormalize $\tilde{u}$ against $v_1, \ldots, v_j$ ➽ Result = $\tilde{s}$ = approximate Schur vector.

4. De£ne $v_k := \tilde{s}$.

5. If approximation not satisfactory go to 1.

6. Else de£ne $h_{i,k} = (Av_k, v_i)$, $i = 1, .., k$,

Thus, for $k = 2$:

$$V_m = \left[\underbrace{v_1, v_2}_{Locked}, \overbrace{v_3, \ldots, v_m}^{active} \right]$$

$$H_m = \left(\begin{array}{cc|cccc} * & * & * & * & * & * \\ & * & * & * & * & * \\ \hline & & * & * & * & * \\ & & * & * & * & * \\ & & & * & * & * \\ & & & & * & * \end{array} \right)$$

➨ **Similar techniques in Subspace iteration [G. Stewart's SRRIT]**

Example: **Matrix Mark(10) – small Markov chain matrix $(N = 55)$.**

- **First eigenpair by iterative Arnoldi with $m = 10$.**

m	$\Re e(\lambda)$	$\Im m(\lambda)$	Res. Norm
10	0.9987435899D+00	0.0	0.246D-01
20	0.9999523324D+00	0.0	0.144D-02
30	0.1000000368D+01	0.0	0.221D-04
40	0.1000000025D+01	0.0	0.508D-06
50	0.9999999996D+00	0.0	0.138D-07

Computing the next 2 eigenvalues of Mark(10).

Eig.	Mat-Vec's	$\Re e(\lambda)$	$\Im m(\lambda)$	Res. Norm
2	60	0.9370509474	0.0	0.870D-03
	69	0.9371549617	0.0	0.175D-04
	78	0.9371501442	0.0	0.313D-06
	87	0.9371501564	0.0	0.490D-08
3	96	0.8112247133	0.0	0.210D-02
	104	0.8097553450	0.0	0.538D-03
	112	0.8096419483	0.0	0.874D-04
	⋮	⋮	⋮	⋮
	⋮	⋮	⋮	⋮
	152	0.8095717167	0.0	0.444D-07

PRECONDITIONING - DAVIDSON'S METHOD

Preconditioning eigenvalue problems

➠ **Goal: To extract good approximations to add to a subspace in a projection process. Result: faster convergence.**

➠ **Best known technique: Shift-and-invert; Work with**

$$B = (A - \sigma I)^{-1}$$

➠ **Some success with polynomial preconditioning [Chebyshev iteration / least-squares polynomials]. Work with**

$$B = p(A)$$

➠ **Above preconditioners preserve eigenvectors. Other methods (Davidson) use a more general preconditioner M.**

Shift-and-invert preconditioning

Main idea: to use Arnoldi, or Lanczos, or subspace iteration for the matrix $B = (A - \sigma I)^{-1}$. The matrix B need not be computed explicitly. Each time we need to apply B to a vector we solve a system with B.

- Factor $B = A - \sigma I = LU$. Then each solution $Bx = y$ requires solving $Lz = y$ and $Ux = z$.

How to deal with complex shifts?

- If A is complex need to work in complex arithmetic.

- If A is real, it is desirable that Arnoldi/ Lanczos algorithms work with a real matrix.

- Idea: Instead of using $B = (A - \sigma I)^{-1}$ use

$$B_+ = \Re e(A - \sigma I)^{-1} = \frac{1}{2}\left[(A - \sigma I)^{-1} + (A - \bar{\sigma} I)^{-1}\right]$$

or

$$B_- = \Im m(A - \sigma I)^{-1} = \frac{1}{2i}\left[(A - \sigma I)^{-1} - (A - \bar{\sigma} I)^{-1}\right]$$

- Little difference between the two.

- Result: $\boxed{B_- = \theta(A - \sigma I)^{-1}(A - \bar{\sigma} I)}$ with $\theta = \Im m(\sigma)$.

Preconditioning by polynomials

Main idea:

Iterate with $p(A)$ instead of A in Arnoldi or Lanczos,..

- **Used very early on in subspace iteration [Rutishauser, 1959.]**
- **Usually not as reliable as Shift-and-invert techniques but less demanding in terms of storage.**

Question: How to £nd a good polynomial (dynamically)?

Approaches:

1. Use of Chebyshev polynomials over ellipses
2. Use polynomials based on Leja points
3. Least-squares polynomials over polygons
4. Polynomials from previous Arnoldi decompositions

Polynomial £lters and implicit restart

- **Goal: to apply polynomial £lter of the form**

$$p(t) = (t - \theta_1)(t - \theta_2) \dots (t - \theta_q)$$

by exploiting the Arnoldi procedure.

Assume $AV_m = V_m H_m + \beta_m v_{m+1} e_m^T$

and consider £rst factor: $(t - \theta_1)$

$$(A - \theta_1 I)V_m = V_m(H_m - \theta_1 I) + \beta_m v_{m+1} e_m^T$$

Let $H_m - \theta_1 I = Q_1 R_1$**. Then,**

$$\begin{aligned}
(A - \theta_1 I)V_m &= V_m Q_1 R_1 + \beta_m v_{m+1} e_m^T \quad \rightarrow \\
(A - \theta_1 I)(V_m Q_1) &= (V_m Q_1) R_1 Q_1 + \beta_m v_{m+1} e_m^T Q_1 \rightarrow \\
A(V_m Q_1) &= (V_m Q_1)(R_1 Q_1 + \theta_1 I) + \beta_m v_{m+1} e_m^T Q_1
\end{aligned}$$

Notation: $R_1 Q_1 + \theta_1 I \equiv H_m^{(1)}$; $(b_{m+1}^{(1)})^T \equiv e_m^T Q_1$; $V_m Q_1 = V_m^{(1)}$

- $$AV_m^{(1)} = V_m^{(1)} H_m^{(1)} + v_{m+1}(b_{m+1}^{(1)})^T$$

- Note that $H_m^{(1)}$ is upper Hessenberg.

- Similar to an Arnoldi decomposition.

Observe:

- $R_1 Q_1 + \theta_1 I \equiv$ matrix resulting from one step of the QR algorithm with shift θ_1 applied to H_m.

- First column of $V_m^{(1)}$ is a multiple of $(A - \theta_1 I)v_1$.

- The columns of $V_m^{(1)}$ are orthonormal.

Can now apply second shift in same way:

$$(A - \theta_2 I) V_m^{(1)} = V_m^{(1)} (H_m^{(1)} - \theta_2 I) + v_{m+1} (b_{m+1}^{(1)})^T \quad \rightarrow$$

Similar process: $(H_m^{(1)} - \theta_2 I) = Q_2 R_2$ **then** $\times Q_2$ **to the right:**

$$(A - \theta_2 I) V_m^{(1)} Q_2 = (V_m^{(1)} Q_2)(R_2 Q_2) + v_{m+1} (b_{m+1}^{(1)})^T Q_2$$

$$A V_m^{(2)} = V_m^{(2)} H_m^{(2)} + v_{m+1} (b_{m+1}^{(2)})^T$$

Now:

First column of $V_m^{(2)}$ **= scalar** $\times (A - \theta_2 I) v_1^{(1)}$

= scalar $\times (A - \theta_2 I)(A - \theta_1 I) v_1$

- **Note that** $(b_{m+1}^{(2)})^T = e_m^T Q_1 Q_2 = [0, 0, \cdots, 0, q_1, q_2, q_3]$

- **Let:** $\hat{V}_{m-2} = [\hat{v}_1, \ldots, \hat{v}_{m-2}]$ **consist of £rst** $m-2$ **columns of** $V_m^{(2)}$ **and** $\hat{H}_{m-2}$ **= leading principal submatrix of** H_m**. Then**

$$A\hat{V}_{m-2} = \hat{V}_{m-2}\hat{H}_{m-2} + \hat{\beta}_{m-1}\hat{v}_{m-1}e_m^T \quad \text{with}$$

$$\hat{\beta}_{m-1}\hat{v}_{m-1} \equiv q_1 v_{m+1} + h_{m-1,m-2}^{(2)} v_{m-1}^{(2)} \quad \|\hat{v}_{m-1}\|_2 = 1$$

- **Result: An Arnoldi process of** $m-2$ **steps with the initial vector** $p(A)v_1$**.**

- **In other words: We know how to apply polynomial £ltering via a form of the Arnoldi process, combined with the QR algorithm.**

The Davidson approach

Goal: to use a more general preconditioner to introduce good new components to the subspace.

- **Ideal new vector would be eigenvector itself!**
- **Next best thing: an approximation to $(A - \mu I)^{-1} r$ where $r = (A - \mu I) z$, current residual.**
- **Approximation written in the form $M^{-1} r$. Note that M can vary at every step if needed.**

ALGORITHM : 11 . Davidson's method (Real symmetric case)

1. Choose an initial unit vector v_1. Set $V_1 = [v_1]$.
2. Until convergence Do:
3. For $j = 1, \ldots, m$ Do:
4. $w := Av_j$.
5. Update $H_j \equiv V_j^T A V_j$
6. Compute the smallest eigenpair μ, y of H_j.
7. $z := V_j y \quad r := Az - \mu z$
8. Test for convergence. If satis£ed Return
9. If $j < m$ compute $t := M_j^{-1} r$
10. compute $V_{j+1} := ORTHN([V_j, t])$
11. EndIf
12. Set $v_1 := z$ and go to 3
13. EndDo
14. EndDo

- Note: Traditional Davidson uses diagonal preconditioning: $M_j = D - \sigma_j I$.

- Will work only for some matrices

Other options:

- Shift-and-invert using ILU [negatives: expensive + hard to parallelize.]

- Filtering (by averaging)

- Filtering by using smoothers (multigrid style)

- Iterative solves [See Jacobi-Davidson]

CONVERGENCE THEORY

The Lanczos Algorithm in the Hermitian Case

Assume eigenvalues sorted increasingly

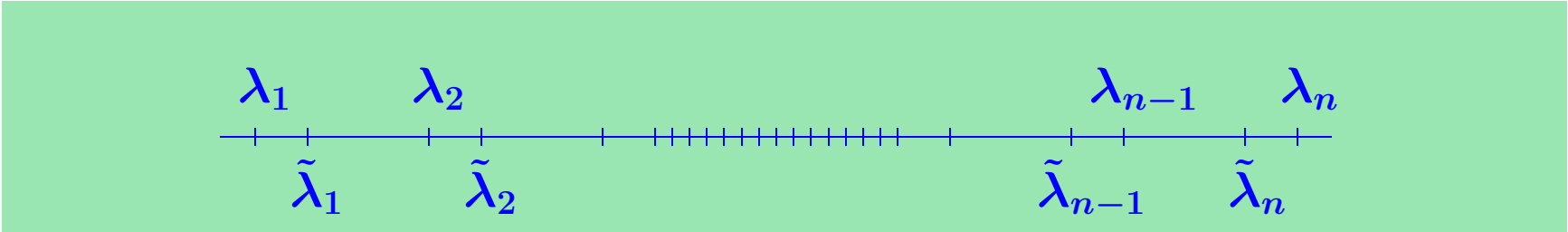
$$\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$$

- **Orthogonal projection method onto K_m;**
- **To derive error bounds, use the Courant characterization**

$$\tilde{\lambda}_1 = \min_{u \in K,\, u \neq 0} \frac{(Au,u)}{(u,u)} = \frac{(A\tilde{u}_1,\tilde{u}_1)}{(\tilde{u}_1,\tilde{u}_1)}$$

$$\tilde{\lambda}_j = \min_{\left\{ \begin{array}{l} u \in K,\, u \neq 0 \\ u \perp \tilde{u}_1,\ldots,\tilde{u}_{j-1} \end{array} \right.} \frac{(Au,u)}{(u,u)} = \frac{(A\tilde{u}_j,\tilde{u}_j)}{(\tilde{u}_j,\tilde{u}_j)}$$

- Bounds for λ_1 easy to £nd – similar to linear systems.
- Ritz values approximate eigenvalues of A inside out:



A-priori error bounds

Theorem [Kaniel, 1966]:

$$0 \leq \lambda_1^{(m)} - \lambda_1 \leq (\lambda_N - \lambda_1) \left[\frac{\tan \angle(v_1, u_1)}{T_{m-1}(1 + 2\gamma_1)} \right]^2$$

where $\gamma_1 = \frac{\lambda_2 - \lambda_1}{\lambda_N - \lambda_2}$**; and** $\angle(v_1, u_1)$ **= acute angle between** v_1 **and** u_1**.**

+ results for other eigenvalues. [Kaniel, Paige, YS]

Theorem [YS,1980]

$$0 \leq \lambda_i^{(m)} - \lambda_i \leq (\lambda_N - \lambda_1) \left[\kappa_i^{(m)} \frac{\tan \angle(v_i, u_i)}{T_{m-i}(1 + 2\gamma_i)} \right]^2$$

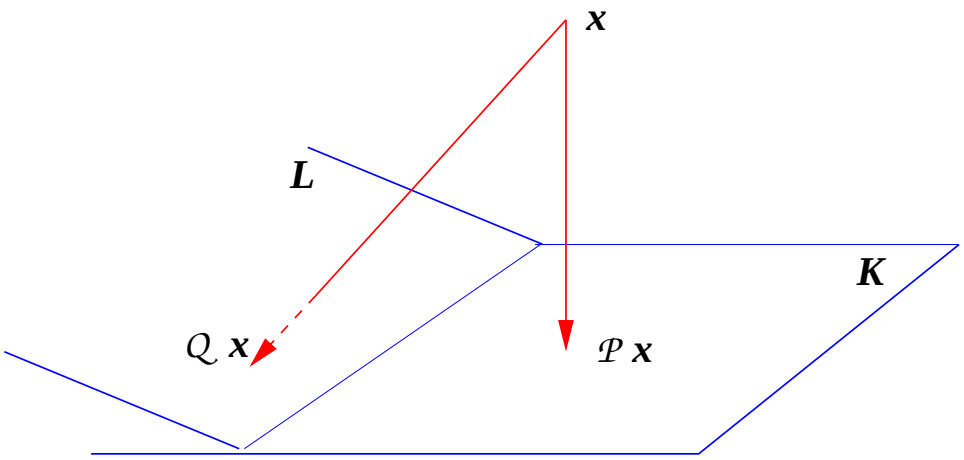
where $\gamma_i = \frac{\lambda_{i+1} - \lambda_i}{\lambda_N - \lambda_{i+1}}$, $\kappa_i^{(m)} = \Pi_{j<i} \frac{\lambda_j^{(m)} - \lambda_N}{\lambda_j^{(m)} - \lambda_i}$

Theory for nonhermitian case

- More dif£cult. No convincing results on ‘global convergence’.
- Can get a general a-priori – a-posteriori error bound

Let $\mathcal{P}$ be the orthogonal projector onto K and

$\mathcal{Q}$ be the (oblique) projector onto K and orthogonally to L.



$$\begin{aligned} \mathcal{P}x &\in K, \quad x - \mathcal{P}x \perp K \\ \mathcal{Q}x &\in K, \quad x - \mathcal{Q}x \perp L \end{aligned}$$

Analysis

Approximate problem amounts to solving

$\mathcal{Q}(Ax - \lambda x) = 0, \quad x \quad \in K$ **or in operator form** $\mathcal{Q}A\mathcal{P}x = \lambda x$

➨ **Set** $A_m \equiv \mathcal{Q}A\mathcal{P}$

THEOREM. Let $\gamma = \|\mathcal{Q}(A-\lambda I)(I-\mathcal{P})\|_2$**. Then the residual norms of the pairs** $\lambda, \mathcal{P}u$ **and** λ, u **for the linear operator** A_m **satisfy, respectively**

$$\|(A_m - \lambda I)\mathcal{P}u\|_2 \leq \gamma \|(I - \mathcal{P})u\|_2$$

$$\|(A_m - \lambda I)u\|_2 \leq \sqrt{|\lambda|^2 + \gamma^2}\, \|(I - \mathcal{P})u\|_2 \,.$$

How to estimate $\|(I - \mathcal{P})u_i\|_2$?

- Assume that A is diagonizable and expand v_1 in the eigen-basis

$$v_1 = \Sigma_{j=1}^{N} \alpha_j u_j$$

- Assume $\alpha_i \neq 0$, $\|u_j\|_2 = 1$ for all j. Then:

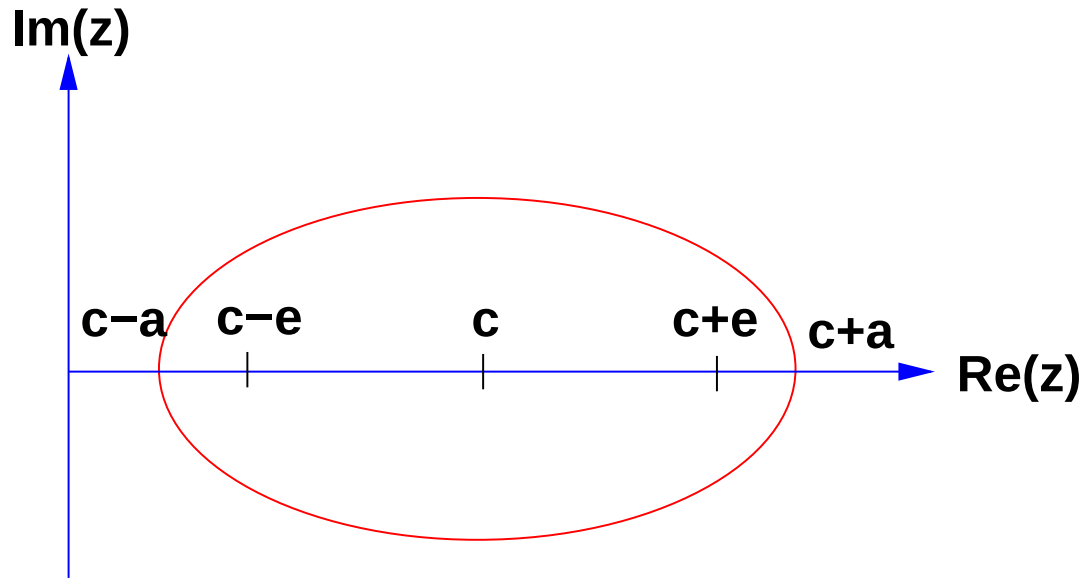
$$\|(I - \mathcal{P})u_i\|_2 \leq \xi_i \epsilon_i^{(m)}$$

where

$$\xi_i = \sum_{j \neq i} \left| \frac{\alpha_j}{\alpha_i} \right| \quad \text{and} \quad \epsilon_i^{(m)} = \min_{\substack{p \in \mathrm{P}_{m-1} \\ p(\lambda_i)=1}} \max_{j \neq i} |p(\lambda_j)|$$

Particular case $i = 1$

- **Assume:** $\Lambda(A)\backslash\{\lambda_1\}$ **is** $\subset$ **an ellipse** $E(c, e, a)$**.**



$$\epsilon_1^{(m)} \leq \frac{C_{m-1}\left(\frac{a}{e}\right)}{|C_{m-1}\left(\frac{\lambda_1 - c}{e}\right)|}$$

where C_{m-1} **= Chebyshev polynomial of degree** $m-1$ **of the £rst kind.**

HARMONIC RITZ VALUES

Harmonic Ritz values: Literature

- **Morgan ’91 [Hermitian case]**
- **Freund ’93 [Non-Hermitian case, Starting point: GMRES]**
- **Morgan ’93 [Nonhermitian case]**
- **Paige, Parlett, Van der Vorst ’95.**
- **Chapman & Y.S. ’95. [use in De¤ated GMRES]**
- **Many publications in the 40s and 50s (Intermediate eigenvalue problems, Lehman intervals, etc..)**

Harmonic Ritz values (continued)

Main idea: take $L = AK$ in projection process

➨ In context of Arnoldi's method. Write $\tilde{u} = V_m y$ then:

$$(A - \tilde{\lambda} I) V_m y \perp \{A V_m\}$$

Using $AV_m = V_{m+1} \underline{H}_m$ ➨

$$\underline{H}_m^H V_{m+1}^H \left[V_{m+1} \underline{H}_m y - \tilde{\lambda} V_m y \right] = 0$$

Notation: $H_m = \underline{H_m} -$ last row. Then

$$\underline{H}_m^H \underline{H}_m y - \tilde{\lambda} H_m^H y = 0$$

or

$$\left(H_m^H H_m + h_{m+1,m}^2 e_m e_m^H\right) y = \tilde{\lambda} H_m^H y$$

Remark:

Assume H_m is nonsingular and multiply both sides by H_m^{-H}. Then, the problem is equivalent to

$$\left(H_m + z_m e_m^H\right) y = \tilde{\lambda} y$$

with $z_m = h_{m+1,m}^2 H_m^{-H} e_m$.

- Modi£ed from H_m only in the last column.

Implementation within Davidson framework

- Slight varation to standard Davidson: Introduce $z_i = M_i^{-1} r_i$ to subspace. Proceed as in FGMRES: $v_{j+1} = Orthn(Az_j, V_j)$.

- From Gram-Schmidt process:

$$Az_j = \sum_{i=1}^{j+1} h_{ij} v_i$$

- Hence the relation

$$AZ_m = V_{m+1} \bar{H}_m$$

Approximation: λ, $\tilde{u} = Z_m y$

Galerkin Condition: $r \perp AZ_m$ gives the generalized problem

$$\bar{H}_m^H \bar{H}_m \, y = \lambda \, \bar{H}_m^H V_{m+1}^H Z_m \, y$$

Davidson's algorithm and two variants

DAVIDSON's ALGORITHM. 1

Start: select v_1**.**

For $j = 1, \ldots, m$ **Do:**

Update $V_j^H A V_j$**.**

Compute Ritz pair $\tilde{u}, \tilde{\lambda}$

Compute $r = A\tilde{u} - \tilde{\lambda}u$

$z = M^{-1}r$

$v_{j+1} = ORTHN(z, V_j)$

EndDo

DAVIDSON'S ALGORITHM. 2

Start: select r**.**

For $j = 1, \ldots, m$ **Do:**

$z = M^{-1}r$

$v_j = ORTHN(z, V_{j-1})$

Compute $w = Av_j$ **and**

Update $V_j^H A V_j$**.**

Compute Ritz pair $\tilde{u}, \tilde{\lambda}$

Compute $r = A\tilde{u} - \tilde{\lambda}u$

EndDo

➨ **Difference: start with a preconditioning operation instead of a matvec. In general minor differences.**

HARMONIC DAVIDSON

Start: select r. **Set** $v_1 = r/\|r\|_2$.

For $j = 1, \ldots, m$ **Do:**

$z_j = M^{-1} r$

Compute $w = Az_j$ **and**

$v_{j+1} = ORTHN(w, V_j, h_{:,j})$;

Update $G = \underline{H}_j^H \underline{H}_j$, **and** $S = \underline{H}_j^H V_{j+1}^H Z_j$;

Compute Ritz pair $\tilde{u}, \tilde{\lambda}$:

$Gy = \tilde{\lambda} Sy$, $\tilde{u} = Z_j y$

Compute $r = A\tilde{u} - \tilde{\lambda} u$

EndDo

- **Arnoldi part identical with that of FGMRES.**

Relation with GMRES (Freund '91)

The Harmonic Ritz values are the roots of the 'GMRES' polynomial:

$$\psi_m = \arg \min_{\psi \in \mathrm{P}_m,\ \psi(0)=1} = \|\psi(A)r_0\|_2$$

Proof. GMRES condition is:

$$\begin{aligned} \beta v_1 - AV_m y &\perp \{AV_m\} \\ \psi_m(A)v_1 &\perp \{AV_m\} \\ (A - \tilde{\lambda}_i I)V_m y_i &\perp \{AV_m\} \end{aligned}$$

➠ Same condition as that of Harmonic Ritz projection. ➠ $\tilde{\lambda}_i$ = Ritz harmonic value, $\tilde{u}_i = V_m y_i$ = Ritz Harmonic vector.

Harmonic values and interior eigenvalues

Let $\tilde{z} = A^{-1}\tilde{u}$ and rewrite the condition $[A - \tilde{\lambda}I]\tilde{u} \perp AK$ as:

$$[\tilde{\lambda}^{-1}I - A^{-1}]\tilde{z} \perp L \qquad \tilde{z} \in L$$

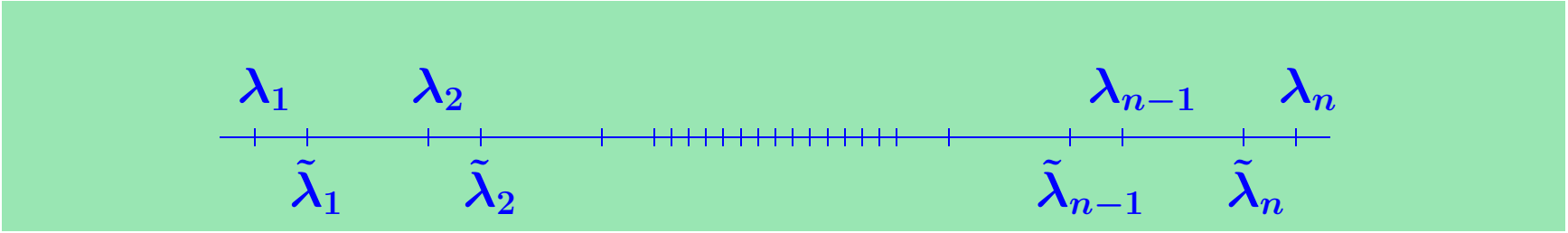
- **Orthogonal projection method for A^{-1}.**
- **Note: This is NOT shift-and-invert in disguise.**
- **Space of approximants is the same as for standard projection.**
- **Interesting consequence for Hermitian case.**

Harmonic Ritz projection in the Hermitian Case

Order eigenvalues increasingly:

$$\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$$

- **Recall: Ritz values approximate eigenvalues of A inside out:**



- **De£ne: $K^{(i)} = \{x \in K \mid x \perp \tilde{u}_1, \tilde{u}_2, \ldots \tilde{u}_{i-1}\}$ ($K^{(1)} \equiv K$).**

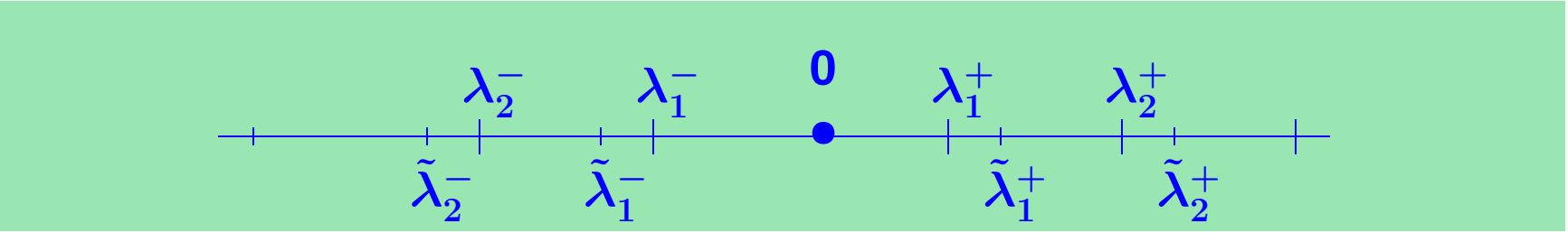
Then

$$\tilde{\lambda}_i = \min_{x \in K_i} \frac{(Ax,x)}{(x,x)}$$

➤ **Apply principle to Harmonic Ritz values** ➤

$$\tilde{\lambda}_1^{-1} \leq \lambda_1^{-1}; \quad \tilde{\lambda}_n^{-1} \geq \lambda_n^{-1} \qquad \longrightarrow \qquad \tilde{\lambda}_1 \geq \lambda_1; \quad \tilde{\lambda}_n \leq \lambda_n$$

➤ **Careful: treat positive and negative eigenvalues separately. Result: [Paige, Parlett, Van der Vorst '95]**



Assume for simplicity that A is SPD.

➤ **De£ne: $K^{(i)} = \{x \in K \mid Ax \perp A\tilde{u}_1, A\tilde{u}_2, \ldots A\tilde{u}_{i-1}\}$ ($K^{(i)} \equiv K$). Then**

$$\tilde{\lambda}_i^{-1} = \max_{x \in K_i} \frac{(Ax,x)}{(Ax,Ax)}$$

Alternative Projections

- Eigenvalue problems are really non-linear systems of equations..

Idea: £nd μ such $(A - \mu I)V$ is nearly rank-de£cient

Leads to

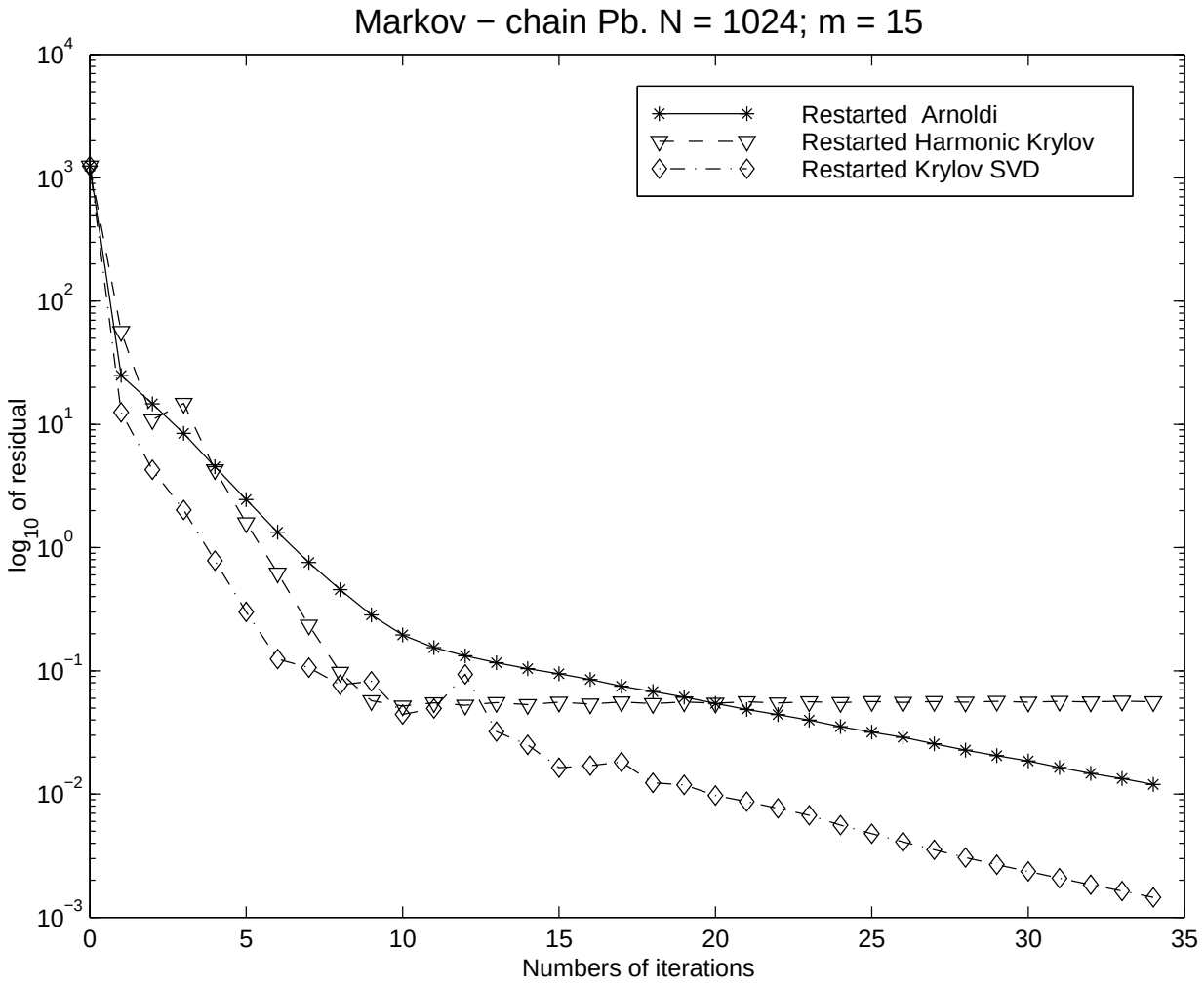
$$det[V^H(A - \mu I)^H(A - \mu I)V] = 0$$

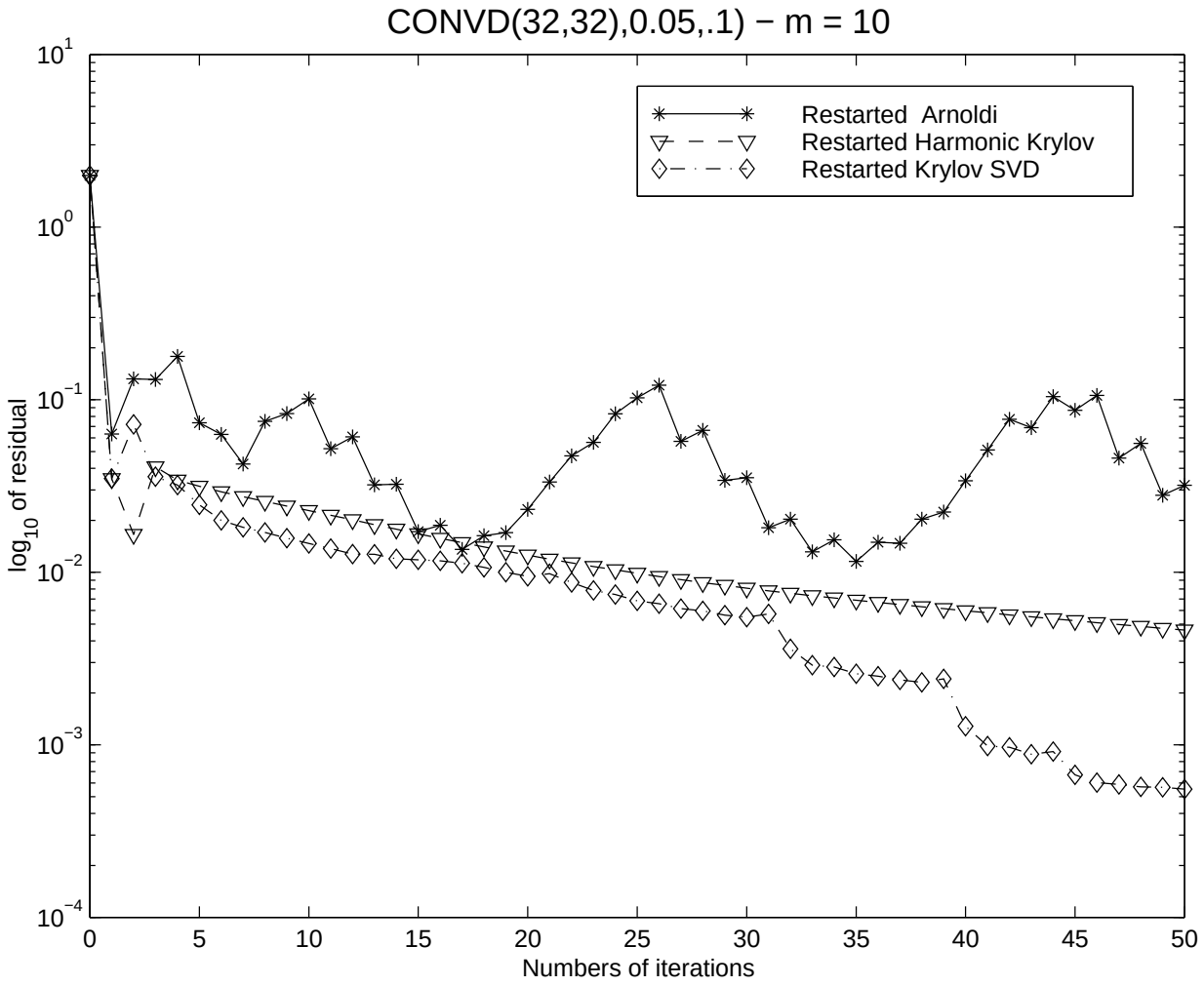
Assume $\mu =$ real. Using $AV_m = V_{m+1}\bar{H}$ $\longrightarrow$ quadratic problem

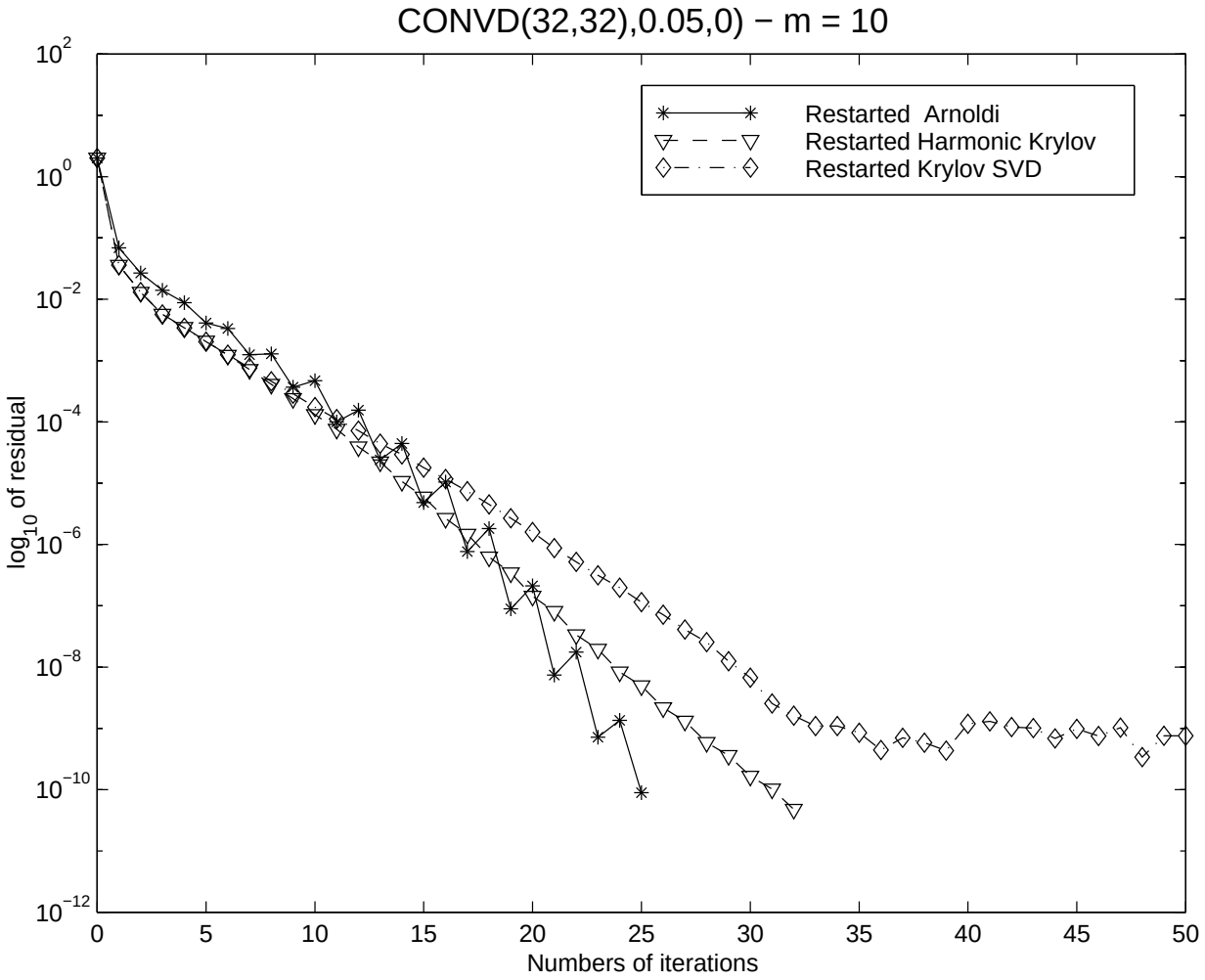
$$(\bar{H}_m^T\bar{H}_m - \mu(H_m + H_m^T) + \mu^2 I)y = 0$$

Alternative formulations

- $\det\left(V^H(A-\mu I)V\right)=0 \rightarrow$ **orthog. projection**
- $\det\left((AV)^H(A-\mu I)V\right)=0 \rightarrow$ **Harmonic projection**
- $\sigma_{min}\left((A-\mu I)V\right)=0 \rightarrow$ **SVD projection**


Markov − chain Pb. N = 1024; m = 15
Restarted Arnoldi
Restarted Harmonic Krylov
Restarted Krylov SVD
log10 of residual
Numbers of iterations
10^4
10^3
10^2
10^1
10^0
10^−1
10^−2
10^−3
0
5
10
15
20
25
30
35


CONVD(32,32),0.05,.1) − m = 10
Restarted Arnoldi
Restarted Harmonic Krylov
Restarted Krylov SVD
log10 of residual
Numbers of iterations
10^1
10^0
10^−1
10^−2
10^−3
10^−4
0
5
10
15
20
25
30
35
40
45
50


CONVD(32,32),0.05,0) − m = 10
Restarted Arnoldi
Restarted Harmonic Krylov
Restarted Krylov SVD
log10 of residual
Numbers of iterations
10^2
10^0
10^−2
10^−4
10^−6
10^−8
10^−10
10^−12
0
5
10
15
20
25
30
35
40
45
50

JACOBI – DAVIDSON

Introduction via Newton's metod

Assumptions: $M = A + E$ **and** $Az \approx \mu z$

Goal: **to £nd an improved eigenpair** $(\mu + \eta, z + v)$**.**

- **Write** $A(z+v) = (\mu+\eta)(z+v)$ **and neglect second order terms + rearrange**

$$(M - \mu I)v - \eta z = -r \quad \text{with} \quad r \equiv (A - \mu I)z$$

- **Unknowns:** η **and** v**.**
- **Underdertermined system. Need one constraint.**
- **Add the condition:** $w^H v = 0$ **for some vector** w**.**

In matrix form:

$$\begin{pmatrix} M-\mu I & -z \\ w^H & 0 \end{pmatrix} \begin{pmatrix} v \\ \eta \end{pmatrix} = \begin{pmatrix} -r \\ 0 \end{pmatrix}$$

- **Eliminate v from second equation:**

$$\begin{pmatrix} M-\mu I & -z \\ 0 & w^H(M-\mu I)^{-1}z \end{pmatrix} \begin{pmatrix} v \\ \eta \end{pmatrix} = \begin{pmatrix} -r \\ w^H(M-\mu I)^{-1}r \end{pmatrix}$$

- **Solution: [Olsen's method]**

$$\eta = \frac{w^H(M-\mu I)^{-1}r}{w^H(M-\mu I)^{-1}z} \qquad v = -(M-\mu I)^{-1}(r-\eta z)$$

- **When $M = A$, corresponds to Newton's method for solving**

$$\begin{cases} (A-\lambda I)u = 0 \\ w^T u = Constant \end{cases}$$

Note: Another way to characterize the solution is:

$$v = -(M - \mu I)^{-1} r + \eta (M - \mu I)^{-1} z, \quad \eta \text{ such that } \quad w^H v = 0$$

- **Involves inverse of $(M - \lambda I)$. Jacobi-Davidson rewrites solution using projectors.**

- **Let P_z be a projector in the direction of z which leaves r invariant. It is of the form**

$$P_z = I - \frac{z s^H}{s^H z}$$

where $s \perp r$. Similarly let P_w any projector which leaves v unchanged. Then the Olsen's solution can be written as

$$[P_z (M - \mu I) P_w] v = -r \qquad w^H v = 0$$

The two solutions are mathematically equivalent.

The Jacobi-Davidson approach

- In orthogonal projection methods (e.g. Arnoldi) we have $r \perp z$
- Also it is natural to take $w \equiv z$. Assume $\|z\|_2 = 1$

With the above assumptions, Olsen's correction equation is mathematically equivalent to £nding v such that :

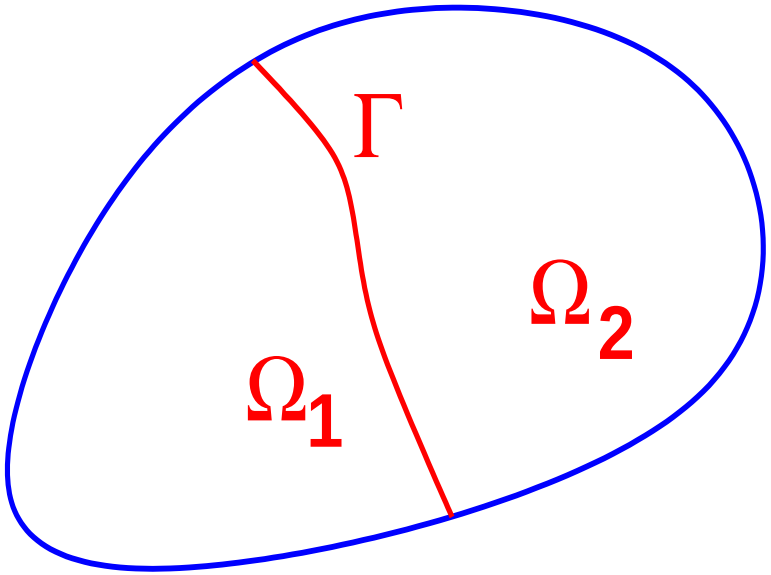
$$(I - zz^H)(M - \mu I)(I - zz^H)v = -r \qquad v \perp z$$

- Main attraction: can use iterative method for the solution of the correction equation. (M -solves not explicitly required).

Automatic Multi-Level Substructuring

Origin: **Extention of substructuring for eigenvalue problems.**

Background: **Domain decomposition. Let $A \in \mathbb{C}^{n\times n}$, Hermitian**



$$\rightarrow A = \begin{pmatrix} B & E \\ E^* & C \end{pmatrix} \quad B \in \mathbb{C}^{(n-p)\times(n-p)}$$

Note: B is block-diagonal

- **B= block-diagonal - represents local matrices -**
- **E represent coupling - C operates on interface variables.**

The problem $Au = \lambda u$, can be written as:

$$\begin{pmatrix} B & E \\ E^* & C \end{pmatrix} \begin{pmatrix} \tilde{u}^B \\ \tilde{u}^S \end{pmatrix} = \lambda \begin{pmatrix} \tilde{u}^B \\ \tilde{u}^S \end{pmatrix}$$

Basic idea of the method for two levels

First step: **eliminate the blocks E, E^*.**

$$U = \begin{pmatrix} I & -B^{-1}E \\ 0 & I \end{pmatrix} \rightarrow U^*AU = \begin{pmatrix} B & 0 \\ 0 & S \end{pmatrix} ; \quad S = C - E^*B^{-1}E.$$

Original problem is equivalent to $U^*AUu = \lambda U^*Uu \rightarrow$

$$\begin{pmatrix} B & 0 \\ 0 & S \end{pmatrix} u = \lambda \begin{pmatrix} I & -B^{-1}E \\ -E^*B^{-1} & M_S \end{pmatrix} u ;$$

with $M_S = I + E^*B^{-2}E$

Second step: neglect the coupling in right-hand side matrix:

$$\begin{pmatrix} B & 0 \\ 0 & S \end{pmatrix} u = \lambda \begin{pmatrix} I & 0 \\ 0 & M_S \end{pmatrix} u \rightarrow \begin{cases} Bv = \mu\, v \\ Sw = \eta\, M_S w \end{cases}$$

➨ Compute a few of the smallest eigenvalues of above problem.

Third step: Build a 'good' subspace to approximate to eigenfunctions of original problem. For projection, use basis the form

$$\left\{ \hat{v}_i = \begin{pmatrix} v_i \\ 0 \end{pmatrix} \quad i = 1, \ldots, m_B; \qquad \hat{w}_j = \begin{pmatrix} 0 \\ w_j \end{pmatrix} \quad j = 1, \ldots, m_S \right\},$$

where $m_B < (n-p)$ and $m_S < p$.

Then use this subspace for a Rayleigh-Ritz projection applied to

$$\begin{pmatrix} B & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} u^B \\ u^S \end{pmatrix} = \lambda \begin{pmatrix} I & -B^{-1}E \\ -E^*B^{-1} & M_S \end{pmatrix} \begin{pmatrix} u^B \\ u^S \end{pmatrix}$$

(Note: not the original problem.)

Final step: **exploit recursion –**

NOTE: algorithm does only one shot of descent - ascent (no iterative improvement).

References:

[1] J. K. BENNIGHOF AND R. B. LEHOUCQ, ***An automated multilevel substructuring method for eigenspace computation in linear elastodynamics*, To appear in SIAM. J. Sci. Comput., (2003).**

[2] R. R. GRAIG, JR. AND M. C. C. BAMPTON, ***Coupling of substructures for dynamic analysis*, AIAA Journal, 6 (1968), pp. 1313–1319.**

[3] W. C. HURTY, ***Vibrations of structural systems by component-mode synthesis*, Journal of the Engineering Mechanics Division, ASCE, 86 (1960), pp. 51–69.**

[4] K. BEKAS AND Y. SAAD, ***Computation of Smallest Eigenvalues using Spectral Schur Complements*, MSI technical report, Jan. 2004 - to appear.**

Spectral Schur complements

- Can interpret AMLS in terms of Schur complements. Start with

$$\begin{pmatrix} B & E \\ E^* & C \end{pmatrix} \begin{pmatrix} u^B \\ u^S \end{pmatrix} = \lambda \begin{pmatrix} u^B \\ u^S \end{pmatrix}$$

- For $\lambda \notin \Lambda(B)$ de£ne $S(\lambda) = C - E^*(B - \lambda I)^{-1}E$

When $\lambda \notin \Lambda(B)$ then $\lambda \in \Lambda(A) \leftrightarrow \lambda \in \Lambda(S(\lambda))$, i.e., iff

$$S(\lambda)u^S = \lambda u^S$$

Observation: The Schur complement problem solved by AMLS can be viewed as the problem resulting from £rst order approximation of $S(\lambda)$ around $\lambda = 0$.

The standard expansion of the resolvent

$$(B-\lambda I)^{-1} = B^{-1}\sum_{k=0}^{\infty}(\lambda B^{-1})^k = \sum_{k=0}^{\infty}\lambda^k B^{-k-1},$$

around $\lambda = 0$, leads to the series

$$S(\lambda) = C - E^*\left(B^{-1} + \lambda B^{-2} + \lambda^2 B^{-3} + \ldots\right)E = S - \sum_{k=1}^{\infty}\lambda^k E^* B^{-k-1} E$$

- **Zeroth order approximation [$\approx$ shift-and-invert with zero shift]**

$$Su^S = \lambda u^S$$

- **First order approximation [AMLS]**

$$Su^S = \lambda(I + E^*B^{-2}E)u^S$$

- **Second order approximation [See Bekas and YS '04]**

$$Su^S = \lambda(I + E^*B^{-2}E + \lambda E^*B^{-3}E)u^S$$

Approximating the eigenvectors

Let λ, u^S be an eigenpair of the nonlinear eigenvalue problem i.e., such that: $S(\lambda)u^S = \lambda u^S$ Then, λ is an eigenvalue of A with associated eigenvector:

$$\begin{pmatrix} -(B-\lambda I)^{-1}Eu^S \\ u^S \end{pmatrix} = \underbrace{\begin{pmatrix} I & -(B-\lambda I)^{-1}E \\ 0 & I \end{pmatrix}}_{U(\lambda)} \begin{pmatrix} 0 \\ u^S \end{pmatrix}$$

- **AMLS approximates the exact prolongator $U(\lambda)$ by $U(0) \equiv U$;**
- **It then adds approximate eigenvectors from B to construct a subspace of approximants to perform a projection process.**

- the space of approximants is spanned by the family of vectors:

$$\left\{\begin{pmatrix} v_i^B \\ 0 \end{pmatrix}\right\}, \qquad \left\{\begin{pmatrix} -B^{-1}Eu_j^S \\ u_j^S \end{pmatrix} = U(0)\begin{pmatrix} 0 \\ u^S \end{pmatrix}\right\},$$

in which v_i^B are eigenvectors of B associated with the smallest eigenvalues

- When λ is small, then $U(\lambda) \approx U(0) \rightarrow$ some simple bounds can obtained for the distance between this space of approximants and exact eigenvectors of A.

AN APPLICATION

Electronic structures and Shrödinger's equation

- **Determining matter's electronic structure can be a major challenge: [a macroscopic amount contains $\approx 10^{23}$ electrons and nuclei]**

- **Solution via the many-body Shrödinger equation:**

$$H\Psi = E\Psi$$

- **The Hamiltonian H is very complex:**

$$H = -\sum_i \frac{\hbar^2 \nabla_i^2}{2M_i} - \sum_j \frac{\hbar^2 \nabla_j^2}{2m} + \frac{1}{2}\sum_{i,j} \frac{Z_i Z_j e^2}{|\vec{R}_i - \vec{R}_j|} - \sum_{i,j} \frac{Z_i e^2}{|\vec{R}_i - \vec{r}_j|} + \frac{1}{2}\sum_{i,j} \frac{e^2}{|\vec{r}_i - \vec{r}_j|}$$

- **Involves sums over all electrons / nuclei and their pairs in terms involving Laplaceans, distances betweens electrons /nuclei.**

Several approximations/theories used

➨ **Born-Oppenheimer approximation: Neglects motion of nuclei [heavier than electrons]**

➨ **Many electrons $\longrightarrow$ one electron systems: each electron sees only the average potential from other electrons/ nuclei.**

Density Functional Theory: observable quantities uniquely determined by ground state charge density. Consequence: Kohn-Sham equations

Kohn-Sham:

$$\left[-\frac{h^2}{2m}\nabla^2 + v_0(r) + \int \frac{\rho(r')}{|r-r'|}dr' + \frac{\delta E_{xc}}{\delta \rho}\right]\Psi(r) = E\Psi(r)$$

- v_0 = external potential, E_{xc} = exchange-correlation enery
- Local Density Approximation: exchange-correlation energy E_{xc} is a simple known funtion
- Pseudopotentials: replace effect of core (inner shell) electrons of the system by an effective potential

➠ **In the end:**

$$\left[-\frac{h^2}{2m}\nabla^2 + V_{tot}[\rho(r), r]\right]\Psi(r) = E\Psi(r)$$

With

$$V_{tot} = V_{ion} + V_H + V_{xc}$$

- V_H = Hartree potential
- V_{xc} = Exchange & Correlation potential
- V_{ion} = Ionic potential

➠ **Local**

➠ **Non-Local**

➠ **Electron Density:**

$$\rho(r) = \Sigma_i^{occup} |\Psi_i(r)|^2$$

➠ **Above problem can be viewed as a nonlinear eigenvalue problem.**

The three potential terms

- Hartree Potential V_H is solution of the Poisson equation:

$$\nabla^2 V_H = -4\pi\rho(r)$$

- Solve using Conjugate Gradient method once ρ is known.

- Potential V_{xc} (exchange & correlation) is approximated by a potential induced by a local density. [Local Density Approximation]. Valid for slowly varying $\rho(r)$.

- Potential V_{ion} is more complex: In matrix terms: a small-rank matrix localized around each atom.

Self Consistency

$$\begin{cases} 1. & \left[-\frac{h^2}{2m}\nabla^2 + V_{tot}[\rho(r), r]\right]\Psi_i(r) = E_i\Psi_i(r), i = 1, ..., i^{occup} \\ 2. & \rho(r) = \Sigma_i^{occup}\,|\Psi_i(r)|^2 \\ 3. & \nabla^2 V_H = -4\pi\rho(r) \quad \rightarrow \quad V_{tot} = V_H + V_{xc} + V_{ion} \end{cases}$$

- Both V_{xc} and V_H, depend on ρ.
- The potentials and charge densities must be self-consistent: Can be viewed as a nonlinear eigenvalue problem
- Preferred approach: Broyden-type quasi-Newton technique
- Typically, a small number of iterations are required
- Not represented above: time stepping.

Resources

- ARPACK:

http://www.caam.rice.edu/software/ARPACK/indexold.html

- "Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide", Zhaojun Bai, James Demmel, Jack Dongarra, Axel Ruhe, and Henk van der Vorst, SIAM, 2000.

- Matrix Algorithms, Vol 2, G. W. Stewart, SIAM, 2001

- Numerical Methods for Large Eigenvalue Problems, Y. Saad, available from

http://www.cs.umn.edu/∼saad/books.html

The end

- The slides for this talk can be downloaded from my web site:

URL: http://www.cs.umn.edu/∼saad

- Follow the "Teaching" icon [▶▶ Calais lecture notes]
- Will include all matlab scripts used for demonstrations
- My e-mail address:

e-mail: saad@cs.umn.edu

MERCI DE VOTRE ATTENTION!

The Test – Durée: 20mn

1. Quel nombre de couleurs trouveriez-vous si vous appliquiez l'algorithme gourmand de coloriage a une matrice a 5 points [disc. du Laplacien a 2-D - grille rectangulaire - point de depart = 1]

2. Montrez les tableaux AA, JA, IA utilisés pour stocker la matrice creuse suivante en format CSR (compressed sparse row)?

$$\begin{pmatrix} 1 & 2 & 0 & 0 \\ 3 & 4 & 0 & 5 \\ 0 & 0 & 6 & 7 \\ 8 & 9 & 0 & 0 \end{pmatrix}$$

3. L'algorithme "steepest descent" converge-t-il quand $A \neq A^T$? Quel (s) algorithmes de projection a une dimension convergent toujours quand A est telle que $A + A^T$ est SDP?

4. A quelle methode de projection correspond l'algorithme du gradient conjugué?

5. Quelles relations d'orthogonalite sont satisfaites par les residus preconditiones z_j de l'algorithme du gradient conjugue preconditioné?

6. Quelle est la matrice de preconditionnement SSOR ($\omega = 1$) pour une matrice A quelconque?

7. A quelle methode de projection correspond l'algorithme des valeurs de Ritz harmoniques?

8. Quelle methode utiliseriez-vous pour calculer toutes les valeurs propres d'une matrice symmetrique dense ?