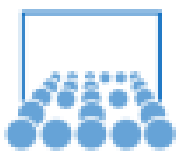
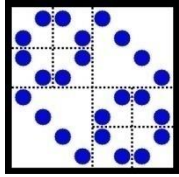


8. Computing Eigenvalues in Parallel



8.1 Introduction

x eigenvector with eigenvalue λ, iff: $Ax = \lambda x, x \neq 0$

A spd, there exists an orthogonal basis of eigenvectors $Au_i = \lambda_i u_i$, i=1,2,…,n

$$A = U \Lambda U^T \qquad or \qquad AU = U\Lambda$$

$$U = (u_1, ..., u_n), \qquad \Lambda = diag(\lambda_i, ..., \lambda_n)$$

The eigenvalues of A are the zeros of the characteristic polynomial:
$p_n(A) = 0 \leftrightarrow p_n(\lambda_i)=0$ for all eigenvalues λ_i, i=1,2,…,n

$\lambda_{min}(A) <= r(A) = x^H Ax / x^H x =< \lambda_{max}$: Rayleigh quotient

In general U may be complex unitary and Λ an upper triangular complex matrix (Schur decomposition)

Allowed operations that do not change the eigenpair: **$Q \cdot A \cdot Q^H$ with unitary Q**

8.2 Parallel Methods for computing all eigenvalues: Jacobi Method

$$off(A)^2 := \|A\|_F^2 - \sum_{i=1}^{n} a_{ii}^2 = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} a_{ij}^2$$

$A = A^T$

Describes the magnitude of the nondiagonal part of A (should → 0)

We use Givens rotations to eliminate a_{pq} for an index pair p,q

$$J(p,q,\theta) := \begin{pmatrix} \ddots & & p & & q & & 0 \\ & 1 & & & & & \\ & & c & & s & & p \\ & & & 1 & & & \\ & & & & \ddots & & \\ & & & & & 1 & \\ & & -s & & c & & q \\ & & & & & 1 & \\ 0 & & & & & & \ddots \end{pmatrix}$$

c = cos(θ),

s = sin(θ)

Jacobi Method

Effect of application of J on A on the nondiagonal entries off(A).

Consider p,q – part of $J^T A J = B$:

$$\begin{pmatrix} b_{pp} & 0 \\ 0 & b_{qq} \end{pmatrix} \overset{!}{=} \begin{pmatrix} b_{pp} & b_{pq} \\ b_{qp} & b_{qq} \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix}^T \begin{pmatrix} a_{pp} & a_{pq} \\ a_{qp} & a_{qq} \end{pmatrix} \begin{pmatrix} c & s \\ -s & c \end{pmatrix}$$

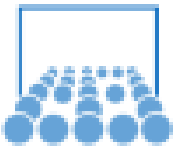
$$b_{pq} = b_{qp} = a_{pq}(c^2 - s^2) + (a_{pp} - a_{qq})cs \overset{!}{=} 0 \Rightarrow \theta, \cos(\theta), \sin(\theta)$$

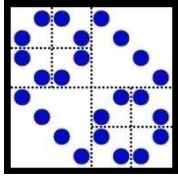
J orthogonal → Frobeniusnorm of A and B are the same:

$$a_{pp}^2 + a_{qq}^2 + 2a_{pq}^2 = b_{pp}^2 + b_{qq}^2$$

$$off^2(B) = \|B\|_F^2 - \sum b_{ii}^2 = \|A\|_F^2 - \sum_{p \neq i \neq q} b_{ii}^2 - (a_{pp}^2 + a_{qq}^2 + 2a_{pq}^2) =$$

$$= \|A\|_F^2 - \sum_{p \neq i \neq q} a_{ii}^2 - (a_{pp}^2 + a_{qq}^2 + 2a_{pq}^2) == \|A\|_F^2 - \sum a_{ii}^2 - 2a_{pq}^2 = off^2(A) - 2a_{pq}^2 < off^2(A)$$





Elimination Sequence

Choose p and q such that a_{pq} is very large (maximum).

Then by $J^T A J$ the size of the off-diagonal entries is reduced by $2 (a_{pq})^2$.

Repeat this transformation for next choice of p and q: A → diagonal.

Monotonic decreasing sequence.

Different strategies for choosing a sequence of p,q:

Maximum a_{pq} optimal, but sequential and expensive!

Cyclic by row:

First use a_{11} to eleminate first row: (p,q)=(1,2),(1,3),…,(1,n)
Then a_{22} for second row: (p,q)=(2,3),….,(2,n)
a_{33} , …, $a_{n-1,n-1}$
Repeat

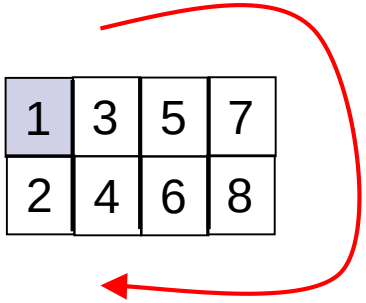
Again sequential!

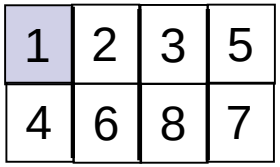
Jacobi Method in Parallel

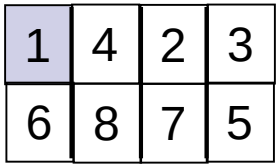
Choose sequence (p,q) such that it allows strong parallelism:

First sweep: (p,q) = (1,2), (3,4), (5,6), (7,8) (in parallel)

Second sweep: (p,q) = (1,4), (2,6), (3,8), (5,7)
Third (1,6), (4,8), (2,7), (3,5)
Fourth (1,8), (6,7), (4,5), (2,3)
.. ..
.. ..







.. ..

n-1 different positions define (n-1)n/2 deleted entries = subdiagonal

Find sequence of partitionings of (1,…n) in pairs, such that all indices appear with the same frequency.

Parallel Transformation

$J^T A$:

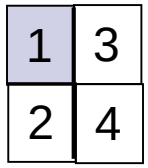
$$\begin{pmatrix} * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ \# & \# & \# & \# & \# & \# & \# & \# \\ \# & \# & \# & \# & \# & \# & \# & \# \\ + & + & + & + & + & + & + & + \\ + & + & + & + & + & + & + & + \\ \sim & \sim & \sim & \sim & \sim & \sim & \sim & \sim \\ \sim & \sim & \sim & \sim & \sim & \sim & \sim & \sim \end{pmatrix}$$

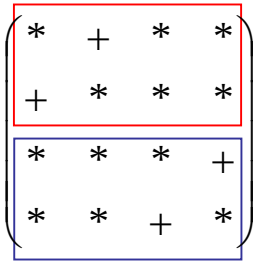
$(J^T A)\, J$:

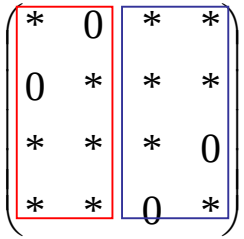
$$\begin{pmatrix} * & * & \# & \# & + & + & \sim & \sim \\ * & * & \# & \# & + & + & \sim & \sim \\ * & * & \# & \# & + & + & \sim & \sim \\ * & * & \# & \# & + & + & \sim & \sim \\ * & * & \# & \# & + & + & \sim & \sim \\ * & * & \# & \# & + & + & \sim & \sim \\ * & * & \# & \# & + & + & \sim & \sim \\ * & * & \# & \# & + & + & \sim & \sim \end{pmatrix}$$

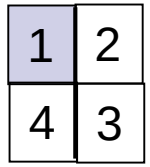
Multiplications with J^T , resp. J

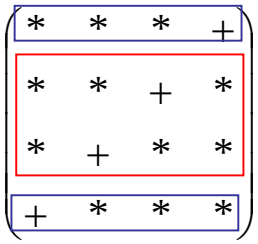
can be done in parallel.

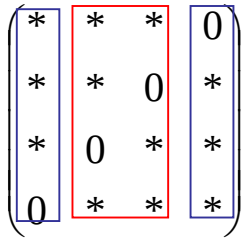


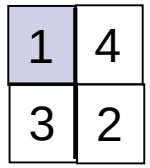


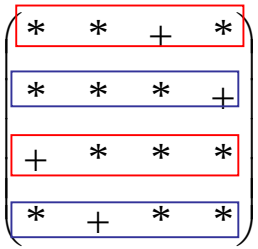


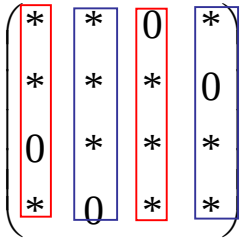












Twelve zeros after three sweeps ←→ twelve nondiagonal entries

Repeat until convergence to diagonal matrix.

Modifications

Block Jacobi sweeps.

Nonsymmetric → hyperbolic rotations

Use orthogonal J(p,q,Θ) and hyperbolic Givens rotations:

$$H(p,q,\phi) := \begin{pmatrix} \ddots & & p & & q & & 0 \\ & 1 & & & & & \\ & & c & & s & & p \\ & & & 1 & & & \\ & & & & \ddots & & \\ & & & & & 1 & \\ & & -s & & c & & q \\ & & & & & 1 & \\ 0 & & & & & & \ddots \end{pmatrix}$$

c = cosh(Φ),

s = sinh(Φ)

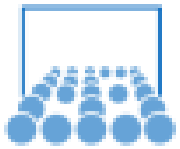
or

use complex Givens rotations to eliminate a_{pq} for an index pair p,q

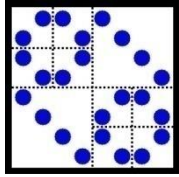
$$B(p,q,\alpha,\theta) := \begin{pmatrix} \ddots & & p & & q & & 0 \\ & 1 & & & & & \\ & & c & & s & & p \\ & & & 1 & & & \\ & & & & \ddots & & \\ & & & & & 1 & \\ & & -s & & c & & q \\ & & & & & 1 & \\ 0 & & & & & & \ddots \end{pmatrix}$$

c = exp(iα)cos(θ),

s = exp(iα)sin(θ)



8.3 Divide & Conquer for tridiagonal $A=A^T$



A divide and conquer approach for computing eigenvalues of a symmetric tridiagonal matrix T.

$$T = \begin{pmatrix} a_1 & b_1 & & \\ b_1 & a_2 & & \\ & & \ddots & b_{n-1} \\ & & b_{n-1} & a_n \end{pmatrix}$$

Idea: Split T in two tridiagonal matrices T_1 and T_2.
Compute eigenvalues of T_1 and T_2.
Recover the original eigenvalues of T as perturbations.

Repeat recursively.

Splitting of T

Set $v := (0 \quad \cdots\cdots \quad 0 \quad 1 \mid \theta \quad 0 \quad \cdots\cdots \quad 0)^T$

$$\tilde{T} := T - \rho v v^T$$ Rank-1 perturbation of T

Aim: Generate zeros at the sub/superdiagonal entries in the middle of T

$$\tilde{T}(m:m+1, m:m+1) = \begin{pmatrix} a_m & b_m \\ b_m & a_{m+1} \end{pmatrix} - \rho \begin{pmatrix} 1 & \theta \\ \theta & \theta^2 \end{pmatrix} =$$

$$= \begin{pmatrix} a_m - \rho & b_m - \rho\theta \\ b_m - \rho\theta & a_{m+1} - \rho\theta^2 \end{pmatrix} \overset{!}{=} \begin{pmatrix} * & 0 \\ 0 & * \end{pmatrix}$$

$$\rho\theta = b_m \longrightarrow T = \begin{pmatrix} T_1 & 0 \\ 0 & T_2 \end{pmatrix} + \rho v v^T$$

Relation between T and T_1, T_2

Assume, that we know the eigenvalues and eigenvectors of T_1 and T_2 .

How can we get the eigenpairs of T?

$$T_1 = U_1 \Lambda_1 U_1^T, \qquad T_2 = U_2 \Lambda_2 U_2^T, \quad \overset{?}{\Rightarrow} \quad T = U \Lambda U^T$$

Note, that T is a rank-1 perturbation of diag(T_1 , T_2).

Recover the original eigenvalues as perturbations of eigenvalues of T_1 and T_2 .

$$T = \begin{pmatrix} T_1 & 0 \\ 0 & T_2 \end{pmatrix} + \rho v v^T = \begin{pmatrix} U_1 \Lambda_1 U_1^T & 0 \\ 0 & U_2 \Lambda_2 U_2^T \end{pmatrix} + \rho v v^T = \begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix} \begin{pmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{pmatrix} \begin{pmatrix} U_1^T & 0 \\ 0 & U_2^T \end{pmatrix} + \rho v v^T$$

$$\Longrightarrow \quad \begin{pmatrix} U_1^T & 0 \\ 0 & U_2^T \end{pmatrix} T \begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix} = \begin{pmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{pmatrix} + \rho \tilde{v} \tilde{v}^T$$

Computing the eigenvector

Hence, we need to compute the eigenvalues of a matrix

of the form "diagonal + rank-1": $D + \rho \tilde{v} \tilde{v}^T$

Let λ_i and u_i be an eigenpair of $D + \rho v v^T$. Then it holds

$$\left(D + \rho \tilde{v} \tilde{v}^T\right) u_i = \lambda_i u_i \Leftrightarrow \left(D - \lambda_i I\right) u_i + \rho \left(\tilde{v}^T u_i\right) \tilde{v} = 0$$

$$u_i = const \cdot \left(D - \lambda_i I\right)^{-1} \tilde{v}$$

Hence, if we know λ_i, then we easily get the eigenvector u_i .

Eigenvalues as Zeros

Furthermore, we get the equation

$$\tilde{v}^T \left(D - \lambda_i I\right)^{-1} \cdot \left[\left(D - \lambda_i I\right) u_i + \rho\left(\tilde{v}^T u_i\right) \tilde{v}\right] = 0$$

$$\tilde{v}^T u_i + \rho\left[\tilde{v}^T \left(D - \lambda_i I\right)^{-1} \tilde{v}\right] \cdot \left[\tilde{v}^T u_i\right] = 0$$

$$f(\lambda) = 1 + \rho\left[\tilde{v}^T \left(D - \lambda I\right)^{-1} \tilde{v}\right] = 1 + \rho\left(\frac{\tilde{v}_1^2}{d_1 - \lambda} + \cdots + \frac{\tilde{v}_n^2}{d_n - \lambda}\right) = 0$$

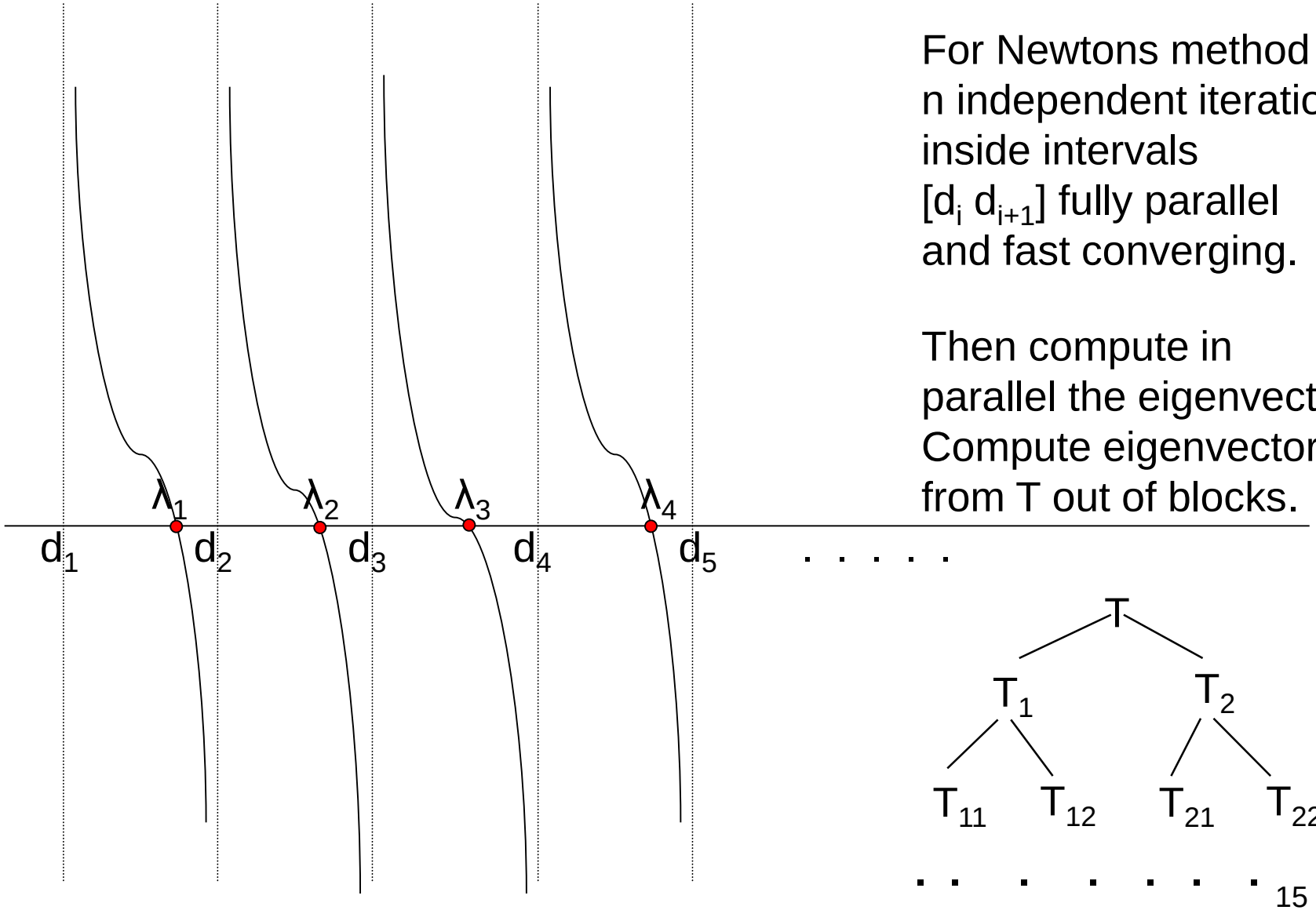
Use Newton's method, to determine the zeroes of function f(λ)

These zeroes are the eigenvalues of $D + \rho \tilde{v} \tilde{v}^T$

and therefore also of T.

Repeat recursively for T_1 and T_2 .

Zeros and poles of f(λ):



For Newtons method
n independent iterations
inside intervals
$[d_i\ d_{i+1}]$ fully parallel
and fast converging.

Then compute in
parallel the eigenvectors.
Compute eigenvectors
from T out of blocks.

8.4 Algorithms for computing a few eigenpairs:

Vector iteration:

$$x^{(k)} = A^k x^{(0)} \Big/ \left\| A^k x^{(0)} \right\| \to v$$

eigenvector to eigenvalue with maximum absolute value!

Easy to parallelize (only Ax), but slow convergence! Only λ_{max}!
Compare Krylov!

Subspace Iteration: Apply the same idea to set of vectors $U^{(0)}=(x^{(0)},\ldots,x^{(m)})$
For k=0,1,..: Replace $U^{(k)}$ by $AU^{(k)}$. Orthogonalize $A\,U^{(k)} \rightarrow U^{(k)}$.
Eigenvectors/values of $U^{(k)T} A\, U^{(k)}$.
Also possible: Increase U upto certain limit. Then restart with one vector.

Inverse iteration:

Apply vector iteration on shifted problem $(A - \sigma I)^{-1}$
for computing the eigenvector nearest to σ.

Expensive! Ill-conditioned linear system!

Rayleigh Quotient Iteration

Rayleigh Quotient iteration: Start with vector y and real $\rho = y^T Ay / y^T y$ and repeat:

$$v = y / \|y\|; \qquad y_{new} = (A - \rho I)^{-1} v; \qquad \rho = \rho + y_{new}^T v / \|y_{new}\|^2;$$

Inverse Iteration with replacing the shift σ by the newest eigenvalue estimate.

y: new eigenvector estimate → leads to new eigenvalue estimate:

$$\rho_{new} = \frac{y_{new}^T A y_{new}}{y_{new}^T y_{new}} = \frac{y_{new}^T (A - \rho I + \rho I) y_{new}}{y_{new}^T y_{new}} = \frac{y_{new}^T (A - \rho I) y_{new}}{y_{new}^T y_{new}} + \rho = \frac{y_{new}^T v}{\|y_{new}\|^2} + \rho$$

Fast convergence, but uncertain to which eigenvalue we will converge.
Expensive! Ill-conditioned!

8.5 Arnoldi (Lanczos) for sparse A

Use the transformation on Hessenberg (tridiagonal) form described for GMRES.
Compute the eigenvalues of the small Hessenberg matrix and use them as approximations for the eigenvalues of the original matrix.

By Arnoldi Orthogonalization of the Krylov subspace (b,Ab,A^2b,…) we get the relation

$$Au_{j-1} = \sum_{k=1}^{j-1} h_{k,j-1} u_k + \tilde{u}_j = \sum_{k=1}^{j} h_{k,j-1} u_k$$

$$A \cdot U_m = A\begin{pmatrix} u_1 & \dots & u_m \end{pmatrix} = \begin{pmatrix} u_1 & \dots & u_{m+1} \end{pmatrix} \cdot \tilde{H}_{m+1,m} = U_m H_{m,m} + h_{m+1,m} u_{m+1}$$

Eigenvalues of $H_{m,m}$ as approximations for A.
(Small $h_{m+1,m}$ → good approximation).
Good approximation for extreme eigenvalues.
For symmetric A, H is tridiagonal.

The same approach can be applied on: f(A)b ~ Uf(H)U^Tb

8.6 Jacobi-Davidson for sparse A

Idea: - No Krylov subspace, more related to Rayleigh quotient and subspace it.
- choose subspace relative to eigenvalue we are looking for
- include preconditioning;

Starting point:
Consider eigenvalue approximations derived by $V_m^H A V_m$
for subspace relative to V_m.
The eigenpairs of $V_m^H A V_m$ are used as approximations to eigenvalues of A

How to choose new subspace V_{m+1} with additional vector u such that the new approximation for special eigenvalues is strongly improved?

For first eigenpair approximation u_m and $t_m=(u_m^H A u_m)/(u_m^H u_m)$, we try to improve these approximations by small corrections u and t to get better estimates u_m+u and t_m+t

$$A(u_m+u)=(t_m+t)(u_m+u), \qquad u \perp u_m,\ u_m^H u_m = 1$$

Jacobi-Davidson II

$$A(u_m + u) = (t_m + t)(u_m + u), \qquad u \perp u_m$$

$$\Downarrow$$

$$(A - t_m I)u = tu_m - (A - t_m I)u_m + \cancel{tu}$$

ignore correction t u of second order

Use orthogonal projection with $I - u_m u_m^H$ from the left.
This leads to

$$(I - u_m u_m^H)(A - t_m I)(I - u_m u_m^H)\, u = -(I - u_m u_m^H)(A - t_m I)u_m$$

$$(I - u_m u_m^H)(A - t_m I)(I - u_m u_m^H)\, u = -(A - t_m I)u_m + u_m \cancel{\left(u_m^H A u_m\right) - t_m u_m \left(u_m^H u_m\right)}$$

$= 0$

Jacobi-Davidson III

For new approximation we have to solve

$$\left(I - u_m u_m^H\right)\left(A - t_m I\right)\left(I - u_m u_m^H\right) u = -\left(A - t_m I\right) u_m$$

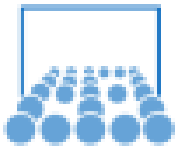
$$P\left(A - t_m I\right) P u = -r_m \qquad or \qquad \tilde{A} u = -r_m$$

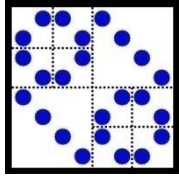
$A - t_m I$ gets ill-conditioned for t_m near eigenvalue, but P is a projection orthogonal to the near singular vector!

$\tilde{A}$ is singular, but linear system is still solvable.

Compared to Inverse Iteration/RQI:

Replace ill-conditioned by singular system.





Jacobi-Davidson IV

New eigenvector estimate $u_m+u \rightarrow v_{m+1}$ also leads to new eigenvalue estimate $t_{m+1} \rightarrow (v_{m+1}^H A v_{m+1})/(v_{m+1}^H v_{m+1})$.

Choose the new estimate v_{m+1} to enlarge the subspace V_m by the new vector u to V_{m+1}.

Compute eigenpairs of $V_{m+1}^H A V_{m+1}$ and choose next eigenvector approximation u_{m+1} appropriately, e.g. maximum, minimum, close to σ.

Repeat this step a few times.

Restart the whole process with last best approximation as starting vector u_1 , resp. 1-dim subspace V_1.

Advantages: Allows to compute also inner eigenvalues without solving more and more ill-conditioned problems like Rayleigh QI.

Jacobi-Davidson V

Main step: Solve linear system

$$P\left(A - t_m I\right)Pu = -r_m \qquad or \qquad P(A - t_m I)\tilde{u} = -r_m$$

approximately.

Therefore, we use a few steps of preconditioned cg or GMRES.

Preconditioner: M^{-1} preconditioner for A → $PM^{-1}P$ preconditioner for PAP

In each iteration step we have to multiply with A, with P, and solve in M.

Simple preconditioner: M = diag(A)
Better preconditioner: SPAI or MSPAI, ILU

8.7 Bisection for computing eigenvalues of a tridiagonal matrix

Observation: The characteristic polynomial of a tridiagonal matrix can be evaluated via the matrix entries in form of a sequence of polynomials with increasing degree:

$$p(\lambda) = \det(T - \lambda I) = \det \begin{pmatrix} \delta_1 - \lambda & \gamma_2 & 0 & \cdots & 0 \\ \gamma_2 & \delta_2 - \lambda & \gamma_3 & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & \gamma_{n-1} & \delta_{n-1} - \lambda & \gamma_n \\ 0 & \cdots & 0 & \gamma_n & \delta_n - \lambda \end{pmatrix}$$

$$p_0(\lambda) = 1$$

$$p_1(\lambda) = \delta_1 - \lambda$$

$$p_2(\lambda) = (\delta_2 - \lambda)p_1(\lambda) - \gamma_2^2 p_0(\lambda)$$

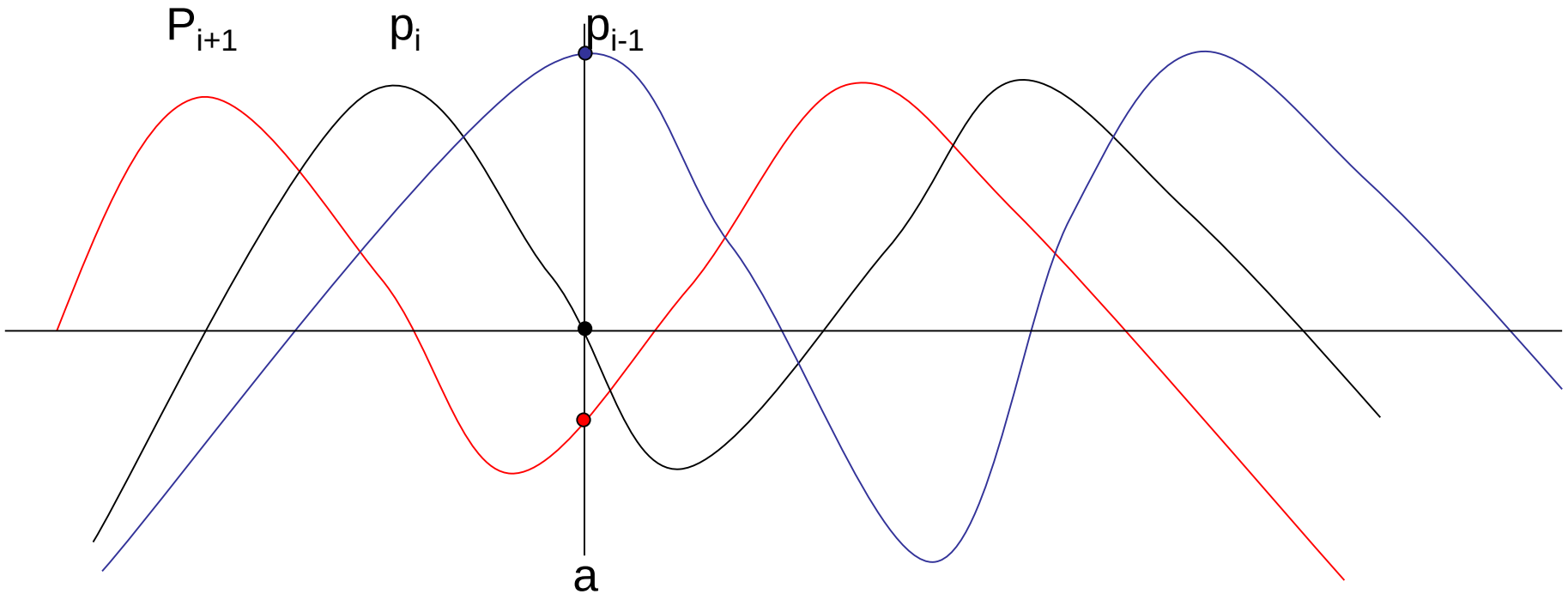
$$p_i(\lambda) = (\delta_i - \lambda)p_{i-1}(\lambda) - \gamma_i^2 p_{i-2}(\lambda), \qquad i = 3,4,\ldots,n$$

$$p(\lambda) = p_n(\lambda)$$

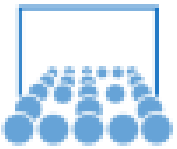
The sequence of polynomials is a Sturm chain:

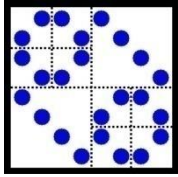
1. All p_i have only single zeros
2. $\text{sign}(p_{n-1}(a)) = -\text{sign}(p'_n(a))$ for all real zeros of $p_n(x)$
3. For i=1,2,…,n-1: $p_{i+1}(a)p_{i-1}(a) < 0$ for all real zeros of $p_i(x)$
4. The polynomial $p_0(x)$ does not change ist sign

Proof by induction.



At all zeros of p_i the neighbors p_{i-1} and p_{i+1} must have different sign.



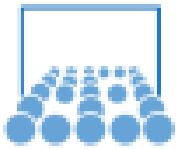


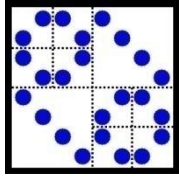
Define w(a):= # sign changes in $p_i(a)$, i=1,…,n .

It holds: w(a) = # zeros of $p_n(x)$ for x<a.

Consider eigenvalues ordered $\lambda_1 < \lambda_2 < \ldots < \lambda_{n-1} < \lambda_n$.
We want to find λ_i, the i-th zero of $p_n(x)$.

It holds: $\lambda_i < a \rightarrow$ w(a) = [# zeros left of a] >= i





Bisection Algorithm:

Choose an interval $I=[a_0, b_0]$ which contains λ_i.
Therefore: $w(b_0) >= i$ and $w(a_0) < i$.
Evaluate the polynomial sequence for $a=(a_0+b_0)/2$ and count the sign changes in the sequence $p_i(a) \rightarrow w(a)$.

If $w(a) >= i$: Replace in interval I b_0 by a
Otherwise: Replace in interval I a_0 by a.

Generates converging sequence of smaller and smaller intervals that contain the eigenvalue λ_i certainly.

Advantages:

- can be easily parallelized on top level
- can be used with high or low accuracy

Allows also Newton method for zeros of p_n