8.4 Algorithms for computing a few eigenpairs:

Vector iteration:

\[ x^{(k)} = A^k x^{(0)} / \| A^k x^{(0)} \| \rightarrow v \]

eigenvector to eigenvalue with maximum absolute value

Easy to parallelize (only Ax), but slow convergence! Only \( \lambda_{\text{max}} \)!

Inverse iteration: Apply vector iteration on shifted problem \( (A - \sigma I)^{-1} \)

for computing the eigenvector nearest to \( \sigma \).

Expensive! Ill-conditioned linear system!
Rayleigh Quotient Iteration

Rayleigh Quotient iteration: Start with vector \( y \) and real \( \rho = y^T A y / y^T y \) and repeat:

\[
v = \frac{y}{\|y\|}; \quad y = (A - \rho I)^{-1} v; \quad \rho = \rho + \frac{y^T v}{\|y\|^2};
\]

Inverse Iteration with replacing the shift \( \sigma \) by the newest eigenvalue estimate.

\[\downarrow\]

\( y \): new eigenvector estimate \( \rightarrow \) leads to new eigenvalue estimate:

\[
\rho_{\text{new}} = \frac{y^T A y}{y^T y} = \frac{y^T (A - \rho I + \rho I) y}{y^T y} = \frac{y^T (A - \rho I) y}{y^T y} + \rho = \frac{y^T v}{\|y\|^2} + \rho
\]

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 \((b, Ab, A^2b, \ldots)\) we get the relation

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

\[
A \cdot U_m = A(u_1 \ldots u_m) = (u_1 \ldots u_{m+1}) \cdot \tilde{H}_{m+1,m} = U_m H_{m,m} + h_{m+1,m} u_{m+1}
\]

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

The same approach can be applied on: \(f(A)b\)
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 $t_m + t$

$$A(u_m + u) = (t_m + t)(u_m + u), \quad u \perp u_m, \ u_m^H u_m = 1$$
Jacobi-Davidson II

\[ A(u_m + u) = (t_m + t)(u_m + u), \quad u \perp u_m \]

\[ (A - t_m I)u = tu_m - (A - t_m I)u_m + tu_m \]

ignore correction $tu_m$ of second order

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

This leads to

\[
\begin{align*}
(I - u_m u_m^H)(A - t_m I)(I - u_m u_m^H)u &= -(I - u_m u_m^H)(A - t_m I)u_m \\
(I - u_m u_m^H)(A - t_m I)(I - u_m u_m^H)u &= -(A - t_m I)u_m + u_m(u_m^H A u_m) - t_m u_m (u_m^H u_m)
\end{align*}
\]

= 0
For new approximation we have to solve

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

\[
P(A - t_m I)Pu = -r_m \quad \text{or} \quad \widetilde{A}u = -r_m
\]

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

\(\widetilde{A}\) 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 \( u_{m+1} \rightarrow v_{m+1} \) also leads to new eigenvalue estimate \( t_{m+1} \rightarrow \frac{(v_{m+1}^H A v_{m+1})}{(v_{m+1}^H v_{m+1})} \).

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 \( \sigma \).

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

\[ P(A - t_m I)Pu = -r_m \quad \text{or} \quad P(A - t_m I)\tilde{u} = -r_m \]

approximately.

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

Preconditioner: M^{-1} preconditioner for A \rightarrow 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
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:

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

\[
p_0(\lambda) = 1 \\
p_1(\lambda) = \delta_1 - \lambda \\
p_2(\lambda) = (\delta_2 - \lambda)p_1(\lambda) - \gamma_2^2 p_0(\lambda) \\
p_i(\lambda) = (\delta_i - \lambda)p_{i-1}(\lambda) - \gamma_i^2 p_{i-2}(\lambda), \quad i = 3,4,\ldots,n \\
p(\lambda) = p_n(\lambda)
\]
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 its sign

Proof by induction.

At all zeros of \( p_i \), the neighbors \( p_{i-1} \) and \( p_{i+1} \) must have different sign.
Consider eigenvalues ordered $\lambda_1 < \lambda_2 < \ldots < \lambda_{n-1} < \lambda_n$. We want to find $\lambda_i$, the $i$-th zero of $p_n(x)$.

Define $w(a) := \# \text{ sign changes in } p_i(a), i=1,\ldots,n$.

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

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

It holds: $\lambda_i < a \rightarrow w(a) = \lceil \# \text{ zeros left of } a \rceil \geq i$. 
Bisection Algorithm:

Choose an interval $I = [a_0, b_0]$ which contains $\lambda_i$. Therefore: $w(b_0) \geq i$ 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 $w(a) \geq i$: 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 $\lambda_i$ certainly.

Advantages:
- can be easily parallelized
- can be used with high or low accuracy
8.8 MR³ for tridiagonal matrices

Idea: Use inverse iteration for computing the eigenvectors of a tridiagonal matrix. In prestep the eigenvalues have to be computed!

Observations:
Inverse iteration is cheap, because of tridiagonal form

Parallel and independent Inverse Iteration for different eigenvalues. High accuracy inspite of (near) singular linear system!

Find a good starting vector such that we need only small number of iterations!
Outline of the algorithm:

Compute eigenvalue approximation $\lambda$ with high relative accuracy (e.g. Bisection)

Find the column number $r$ of $(T - \lambda I)^{-1}$ with largest norm
Use bidiagonal factorizations $T = LDL^T$.

Perform one step of inverse iteration $(T - \lambda I) z = e_r$

MR$^3$ allows the computation of eigenvectors with high accuracy (also for small or close together eigenvalues) using factorizations: $L_+ D_+ L_+^T = LDL^T - \sigma I$. 

Multiple Relatively Robust Representations = MRRR
8.9 Sequential QR Algorithm for computing all Eigenvalues:

Standard algorithm for computing eigenpairs: QR-algorithm

Prestep: Transform A by Givens or Householder matrices to tridiagonal form.

\[
G_{2,3} \begin{pmatrix}
 a_{11} & a_{12} & a_{13} & * & * \\
 a_{21} & a_{22} & a_{23} & * & * \\
     & a_{32} & a_{33} & * & * \\
     &     &     & * & * \\
     &     &     & * & *
\end{pmatrix} G_{2,3}^H
\]

to eliminate \(a_{31}\) and \(a_{13}\)

Main difference to QR-factorization:
- Use subdiagonal entry for eliminating elements
- Apply Q from both sides
- Gives tridiagonal matrix (or upper Hessenberg for nonsymmetric A).

For better parallelism use block Householder like in the QR-decomposition.
QR-Algorithm

First step:
By Householder matrices transform \( A \) by equivalence transformations on tridiagonal (upper Hessenberg) form: \( A \rightarrow H \cdot A \cdot H^T = T \)

For the following we assume \( A \) already tridiagonal (upper Hessenberg)

Second step: Compute QR-decomposition of \( A \), \( A = QR \) and

replace \( A = A_{\text{old}} \) by \( A_{\text{new}} = RQ \)

\[
A_{\text{new}} = RQ = (Q^T A)Q = Q^T AQ
\]

Therefore \( A \) and \( A_{\text{new}} \) have the same eigenvalues

Repeat these QR-steps until convergence against diagonal (upper triangular) matrix.
Use last diagonal entry \( r \) as shift \( A-rI \), apply QR step on shifted matrix.
Reduce full matrix to tridiagonal (upper Hessenberg) Sequential!

For allowing better parallelism reduce matrix $A$ to block-banded form, and then in a second step to tridiagonal form.

Advantage:
First step allows block/BLAS3 operations and is good in parallel. second step is cheap; can be implemented e.g. by $\text{MR}^3$. 
Bothsided Householder for Tridiagonalization

Compute Householder vector $u$ in order to eliminate subtridiagonal entries in the first column/row.

Apply

\[ A \rightarrow (I-2uu^H)A(I-2uu^H) = A - 2u(u^HA) - 2(Au)u^H + 4 uu^H(u^HAu) = \]
\[ = A - 2u(u^HA+ru^H) - 2(Au+ru)u^H = \]
\[ = A - uy^H -yu^H \]

To reduce BLAS2 operations work blockwise,

\[ A \rightarrow A - UY^H -YU^H \quad \text{(BLAS3)} \]

but still first $Au$ is needed (BLAS2).
Block-Band reduction

In the first step find QR decomposition of subblock \( A(1 + b : n, 1 : n_b) = A_1 \) where \( b \) is the bandwidth and \( n_b \) is a block size.

Compute QR decomposition of black part \( A_1 \):

Applying \((I, Q^H)\) from the left leads to triangular form of black part.

Applying from both sides: Band structure.

Store Householder vectors on positions of new generated zeros.

Use Cholesky QR.
2D-Cyclic Data Distribution

4 x 4 – Matrix on 2 x 2 processor array

Advantage: better load balancing because matrices and Householder vectors are getting smaller.
Use integration over closed curve $C$ in complex plane in order to derive an approximation to the subspace built by the eigenvectors related to the eigenvalues inclosed by the curve.

Closed curve contains 2 eigenvalues with 2 (orthogonal) eigenvectors $\rightarrow$ 2-dim subspace
For rank 2 matrix $Y$, the computed matrix $U$ contains the span of the 2 eigenvectors in $C$.

With $U$ computed build the small matrices

$$A_U = U^H A U, \quad B_U = U^H U$$

and solve the small eigenvalue problem

$$A_U W = B_U W \cdot \Lambda$$

Repeat with $Y = X = U^* W$ until convergence.
FEAST c’t

Main work: Use quadrature rule with discretization points $z_j$, $j=1,...,p$, in C to compute the integral.

Therefore, we need to solve

$$\left(z_j I - A\right) U = Y$$

for different $z_j$ and blocks $Y=(y_1,...,y_m)$

$$U = \frac{1}{2\pi i} \sum_{j=1}^{p} \omega_j \varphi(t_j)\left(\varphi(t_j)I - A\right)^{-1} Y =$$

$$= \frac{1}{2\pi i} \sum_{j=1}^{p} \omega_j z_j \left(z_j I - A\right)^{-1} Y$$
Advantages:

First level of parallelismus:
Choose different curves containing all wanted eigenvalues.

Second level of parallelismus:
Solve linear equations for different $z_j$ and for one $z_j$ for different columns of $Y$.

Third level of parallelismus:
Parallelize iterative solver.
Problem

Linear equations are extremely ill-conditioned if eigenvalues are close to curve and therefore $z_j - A$ very ill-conditioned.

Slow convergence of iterative solver!

Preconditioning?