Numerical Methods in Linear Algebra

Dénes Sexty

University of Graz

2020-2021, winter term

UNIVERSITÄT GRAZ
UNIVERSITY OF GRAZ

1. Examples: Bridge, multidim Newton's method, QM, vibrational eigenmodes
2. Linear equations
3. Gauss elimination, Pivoting
4. Numerical errors, condition number
5. LU decomposition, iterative improvement
6. Householder reduction
7. iterative solution: Gauss-Seidel, Successive overrelaxation (SOR), Conjugate gradient (CG) and others
8. Eigenvalues Subspace methods, power iteration, Lánczos method, Krylov Schur
9. **Eigenvalues transformation method: Jacobi, QR**
10. **Diagonalization with Fourier transformation**

For the moment, take real symmetric (or complex Hermitian) matrices.
Any such matrix A can be diagonalized orthogonally (unitarily)

$$D = U^+ A U, \quad U^+ = U^{-1} \implies U^+ U = 1, \tag{1}$$

where $D = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ is a diagonal matrix.
Also, we have $AU = UD$ which means that columns of U are the eigenvectors of A.

$$U = (u_1, u_2, \ldots, u_n), \qquad A(u_1, u_2, \ldots, u_n) = (\lambda_1 u_1, \lambda_2 u_2, \ldots, \lambda_n u_n) \tag{2}$$

The transformation methods find the eigenvalues by succesively applying orthogonal transformations:

$$\begin{aligned} A^{(1)} &= U_0^+ A U_0 \\ A^{(2)} &= U_1^+ A^{(1)} U_1 = U_1^+ U_0^+ A U_0 U_1 \\ A^{(k+1)} &= U_k^+ A^{(k)} U_k = U_k^+ \ldots U_1^+ U_0^+ A U_0 U_1 \ldots U_k \end{aligned} \tag{3}$$

The orthogonal transformations are choosen such that the non-diagonal elements in $A^{(k)}$ converge to zero.
In contrast to subspace methods, we get all eigenvalues and eigenvectors of the matrix

Let's define the following transformation matrix:

$$U(i,j,\varphi) = \begin{bmatrix} 1 & & 0 & & 0 & & 0 \\ & \ddots & \vdots & & \vdots & & \\ 0 & \dots & \cos\varphi & \dots & -\sin\varphi & \dots & 0 \\ & & \vdots & \ddots & \vdots & & \\ 0 & \dots & \sin\varphi & \dots & \cos\varphi & \dots & 0 \\ & & \vdots & & \vdots & \ddots & \\ 0 & & 0 & & 0 & & 1 \end{bmatrix}, \tag{4}$$

$$U(i,j,\varphi)_{mn} = \begin{cases} 1 & \text{if } m = n,\ m \neq i \text{ and } m \neq j \\ \cos\varphi & \text{if } m = n = i \text{ or } m = n = j \\ -\sin\varphi & \text{if } m = i \text{ and } n = j \\ \sin\varphi & \text{if } m = j \text{ and } n = i \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

So the $\begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}$ is "embedded" into the i and j-th row and column of an $n \times n$ unit matrix

Using $\cos^2\varphi + \sin^2\varphi = 1$ we can easily show: $U(i,j,\varphi)^T U(i,j,\varphi) = 1$.
Now we have for $A = A^T$:

$$\begin{aligned} B &= U(i,j,\varphi)^T A U(i,j,\varphi) \\ B^T &= U(i,j,\varphi)^T A^T U(i,j,\varphi) = U(i,j,\varphi)^T A U(i,j,\varphi) = B \end{aligned} \tag{6}$$

So the matrix stays symmetric.
Let's look at the transformed matrix when transforming with $U(i,j,\varphi)$:

$$A_{kl}^{(n+1)} = U_{km}^T A_{mn}^{(n)} U_{nl} = U_{mk} U_{nl} A_{mn}^{(n)} \tag{7}$$

If k and l are neither i nor j, the transformation matrices are unit matrices:

$$A_{kl}^{(n+1)} = \delta_{mk}\delta_{nl} A_{mn}^{(n)} = A_{kl}^{(n)} \text{ for } k,l \neq i,j \tag{8}$$

this means only i-th and j-th row and column are changed.

Using $A_{kl}^{(n+1)} = U_{mk} U_{nl} A_{mn}^{(n)}$ we calculate the changed entries of the matrix

$$
\begin{aligned}
A_{ki}^{(n+1)} &= A_{ki}^{(n)} \cos\varphi + A_{kj}^{(n)} \sin\varphi \quad \text{for } k \neq i,j \qquad (9)\\
A_{kj}^{(n+1)} &= A_{kj}^{(n)} \cos\varphi - A_{ki}^{(n)} \sin\varphi \quad \text{for } k \neq i,j \\
A_{ii}^{(n+1)} &= A_{ii}^{(n)} \cos^2\varphi + 2A_{ij}^{(n)} \cos\varphi \sin\varphi + A_{jj}^{(n)} \sin^2\varphi \\
A_{jj}^{(n+1)} &= A_{jj}^{(n)} \cos^2\varphi - 2A_{ij}^{(n)} \cos\varphi \sin\varphi + A_{ii}^{(n)} \sin^2\varphi \\
A_{ij}^{(n+1)} &= A_{ij}^{(n)} (\cos^2\varphi - \sin^2\varphi) + \left(A_{jj}^{(n)} - A_{ii}^{(n)}\right) \cos\varphi \sin\varphi
\end{aligned}
$$

How should we choose i, j and φ?
Jacobi method: We should choose e.g. the largest absolute value off diagonal element $A_{ij}^{(n)}$ and make it vanish:

$$
A_{ij}^{(n+1)} = A_{ij}^{(n)} (\cos^2\varphi - \sin^2\varphi) + \left(A_{jj}^{(n)} - A_{ii}^{(n)}\right) \cos\varphi \sin\varphi = 0 \qquad (10)
$$

The angle φ is calculated from

$$
\frac{A_{ii}^{(n)} - A_{jj}^{(n)}}{A_{ij}^{(n)}} = \frac{\cos^2\varphi - \sin^2\varphi}{\sin\varphi \cos\varphi} = 2\beta \qquad (11)
$$

where we defined the shorthand $\beta > 0$ (otherwise we swap i and j)

We actually only need $\sin\varphi = s, \quad \cos\varphi = c$

$$c^2 - s^2 = 1 - 2s^2 = 2\beta sc \tag{12}$$

One more square to get rid of c:

$$1 - 4s^2 + 4s^4 = 4\beta^2 s^2 \underbrace{c^2}_{=1-s^2} = 4\beta^2 s^2 - 4\beta^2 s^4 \tag{13}$$

Solving the quadratic eq: $s^4 - s^2 + 1/(4 + 4\beta^2) = 0$

$$s^2 = \frac{1}{2}\left(1 \pm \sqrt{1 - \frac{4}{4+4\beta^2}}\right) = \frac{1}{2}\left(1 \pm \sqrt{\frac{4\beta^2}{4+4\beta^2}}\right) = \frac{1}{2}\left(1 \pm \frac{\beta}{\sqrt{1+\beta^2}}\right) \tag{14}$$

We choose the negative sign, than we get $s^2 < 1/2$ and $0 \leq \varphi \leq \pi/4$ We also get

$$sc = \sqrt{s^2 c^2} = \frac{1}{2}\sqrt{-}\sqrt{+} = \frac{1}{2}\sqrt{1 - \frac{\beta^2}{1+\beta^2}} = \frac{1}{2}\frac{1}{\sqrt{1+\beta^2}} \tag{15}$$

For nonzero A_{ij} we calculate

$$\beta = \frac{1}{2}\frac{A_{ii} - A_{jj}}{A_{ij}}, \quad c^2, s^2 = \frac{1}{2}\left(1 \pm \frac{\beta}{\sqrt{1+\beta^2}}\right), \quad sc = \frac{1}{2}\frac{1}{\sqrt{1+\beta^2}} \tag{16}$$

if $\beta < 0$ we swap i and j first.

We than update the matrix (B is the shorthand for $A^{(n+1)}$):

$$\begin{aligned} B_{ki} &= A_{ki}c + A_{kj}s \quad \text{for } k \neq i,j \\ B_{kj} &= A_{kj}c - A_{ki}s \quad \text{for } k \neq i,j \\ B_{ii} &= A_{ii}c^2 + 2A_{ij}sc + A_{jj}s^2 \\ B_{jj} &= A_{jj}c^2 - 2A_{ij}sc + A_{ii}s^2 \\ B_{ij} &= 0 \end{aligned} \tag{17}$$

We made one offdiagonal element zero, but influenced other ones. Do we get closer to diagonal?
Let's use the Frobenius norm:

$$\|A\|_F = \sum_{i,j} A_{ij}^2 = \sum_{i,j} A_{ij}(A^T)_{ji} = \mathrm{Tr}(AA^T) \tag{18}$$

Now we calculate the Frobenius norm of the transformed matrix $B = A'$

$$\|B\|_F = \mathrm{Tr}(B^T B) = \mathrm{Tr}(BB) = \mathrm{Tr}(U^T A \underbrace{UU^T}_{=1} AU) = \mathrm{Tr}(AA) = \|A\|_F \tag{19}$$

So the total Frobenius norm is unchanged. We can also prove that the sum squared of the diagonal part increases in each transformation:
First note that:

$$\begin{pmatrix} B_{ii} & B_{ij} \\ B_{ji} & B_{jj} \end{pmatrix} = \begin{pmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{pmatrix} \begin{pmatrix} A_{ii} & A_{ij} \\ A_{ji} & A_{jj} \end{pmatrix} \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} \tag{20}$$

We can also use the conservation of the Frobenius norm for these 2×2 matrices

We choose the angle such that we zero the non-diagonal element in B:

$$B_{ii}^2 + B_{jj}^2 = A_{ii}^2 + A_{jj}^2 + 2A_{ij}^2 \tag{21}$$

Since the other diagonal elements are unchanged in the transformed matrix, we have:

$$\sum_k B_{kk}^2 = \sum_k A_{kk}^2 + 2A_{ij}^2 \tag{22}$$

Which means that the magnitude of the diagonal elemnts grow, the magnitude of offdiagonals decrease in every step.

Usually one does not always look for the largest offdiagonal element, but e.g. performs zeroing them in order and do many "sweeps".

Convergence criterium: $\sum_{i \neq j} a_{ij}^2 < \epsilon \sum_i a_{ii}^2$

During the transformations we keep track of the transformations:

$$A^{(0)} = A, \quad V^{(0)} = 1 \tag{23}$$
$$A^{(n+1)} = U^T A^{(n)} U$$
$$V^{(n+1)} = V^{(n)} U$$

Thus we get the eigenvalues in V
Updating the eigenvectors can be done cheaply: only i-th and j-th column changes in a step zeroing A_{ij}

$$(VU(i,j,\varphi))_{ki} = V_{ki}\cos\varphi + V_{kj}\sin\varphi \tag{24}$$
$$(VU(i,j,\varphi))_{kj} = -V_{ki}\sin\varphi + V_{kj}\cos\varphi$$

Take H Hermitian $N \times N$ matrix. H has real eigenvalues

$$H = A + iB, \quad H^{+} = A^{T} - iB^{T} = A + iB \tag{25}$$

$\Longrightarrow$ A is real symmetric, B is real and antisymmetric.
Eigenvalues are given by:

$$Hv = rv, \quad v = u + iw \tag{26}$$

with real vectors u and w

$$(A + iB)(u + iw) = r(u + iw) \tag{27}$$

This implies

$$\begin{aligned} Au - Bw &= ru \\ Bu + Aw &= rw \end{aligned} \qquad \text{which is} \qquad \begin{bmatrix} A & -B \\ B & A \end{bmatrix} \begin{bmatrix} u \\ w \end{bmatrix} = r \begin{bmatrix} u \\ w \end{bmatrix} \tag{28}$$

An eigenvalue problem of a $2N \times 2N$ real, symmetric matrix

Thus we can use e.g. Jacobi method for

$$\begin{bmatrix} A & -B \\ B & A \end{bmatrix} \tag{29}$$

$2N \times 2N \implies$ there are $2N$ eigenvalues instead of N

$$\begin{bmatrix} A & -B \\ B & A \end{bmatrix} \begin{bmatrix} u \\ w \end{bmatrix} = r \begin{bmatrix} u \\ w \end{bmatrix} \quad \implies \quad \begin{bmatrix} A & -B \\ B & A \end{bmatrix} \begin{bmatrix} -w \\ u \end{bmatrix} = r \begin{bmatrix} -w \\ u \end{bmatrix} \tag{30}$$

So we have two eigenvectors for every eigenvalue $\implies$ The eigenvalues are degenerate.
The two vectors differ by a phase factor: $v_2 = -w + iu = i(u + iw) = iv_1$
We choose either vector for each eigenvalue, than we have all eigenvalues and eigenvectors of H

We define the following iterative procedure starting from a matrix A

$$A_k = Q_k R_k, \quad A_{k+1} = R_k Q_k, \tag{31}$$

We do the decomposition to a Q orthogonal matrix and an R upper triangular matrix (using the Householder reduction)
A_{k+1} is similar to A_k and thus to $A_1 = A$

$$A_{k+1} = R_k Q_k = Q_k^T Q_k R_k Q_k = Q_k^T A_k Q_k = Q_k^T Q_{k-1}^T \dots Q_1^T A Q_1 Q_2 \dots Q_k \tag{32}$$

A_{k+1} has the same eigenvalues as A

Using the notation $Q^{(k)} = Q_1 Q_2 \dots Q_k$ we have

$$Q^{(k)} R_k = Q_1 \dots Q_{k-1} A_k = Q_1 \dots Q_{k-1} R_{k-1} Q_{k-1} = \tag{33}$$
$$Q_1 \dots Q_{k-2} A_{k-1} Q_{k-1} = \dots = A Q_1 \dots Q_{k-1} = A Q^{(k-1)}$$

So we have proven

$$A Q^{(k-1)} = Q^{(k)} R_k \tag{34}$$

We also define $R^{(k)} = R_k \dots R_1$. Then we write

$$Q^{(k)} R^{(k)} = Q^{(k)} R_k R^{(k-1)} = A Q^{(k-1)} R^{(k-1)} \tag{35}$$

Doing this $k-1$ more times we have a decomposition for the k-th power of AKA^k

$$A^k = Q^{(k)} R^{(k)} \tag{36}$$

Let's denote the columns of $Q^{(k)}$ as $u_i^{(k)}$
Now let's look at the first column of $AQ^{(k)} = Q^{(k+1)} R_{k+1}$:

$$A u_1^{(k)} = u_1^{(k+1)} (R_{k+1})_{11} \tag{37}$$

The orthogonality condition of $Q^{(k)}$ means that the $u_i^{(k)}$ vectors are orthonormal, so this equation is just the definition of the power iteration.

$\implies$ $(R_k)_{11}$ converges to the largest eigenvalue and $u_1^{(k)}$ to its eigenvector.

We start again from: $AQ^{(k)} = Q^{(k+1)}R_{k+1}$, which implies $A = Q^{(k+1)}R_{k+1}Q^{(k)T}$, which we invert and transpose

$$A^{-1} = Q^{(k)}R_{k+1}^{-1}Q^{(k+1)T} \tag{38}$$
$$(A^{-1})^T = Q^{(k+1)}(R_{k+1}^{-1})^T Q^{(k)T}$$
$$(A^{-1})^T Q^{(k)} = Q^{(k+1)}(R_{k+1}^{-1})^T$$

The inverse of an upper (lower) triangular matrix is again upper (lower triangular), so $(R_{k+1}^{-1})^T$ is lower triangular.
We look at the last column of the above equation:

$$(A^{-1})^T u_n^{(k)} = u_n^{(k+1)}(R_{k+1}^{-1})_{nn}^T \tag{39}$$

Since columnvectors in $Q^{(k)}$ are normalized, This is just the inverse iteration for the transpose of the matrix (or equivalently, inverse iteration for left eigenvectors)
$\Longrightarrow$ The last column of $Q^{(k)}$ converges to an eigenvector and $(R_{k+1}^{-1})_{nn}$ to an eigenvalue

So first and last column of $Q^{(k)}$ converges to eigenvectors, what about the rest?

QR decomposition can be understood as **orthonormalization procedure**:

$$M = QR \tag{40}$$

M is a collection of column vectors m_i
Q is a collection of orthonormal column vectors q_i (as $Q^T Q = 1$).
q_1 is the same as m_1 normalized: $m_1 = q_1 r_{11}$
m_2 is calculated from the first 2 vectors of Q: $m_2 = q_1 r_{12} + q_2 r_{22}$
looking at it "backwards":

$$q_2 = \frac{1}{r_{22}}(m_2 - q_1 r_{12}) = \frac{1}{r_{22}}(m_2 - q_1(q_1^+ m_2)) \tag{41}$$

Which means q_2 is just the (normalized) part of m_2 which is orthogonal to q_1.
and so on with $q_3, q_4, \ldots$

One can actually use this orthogonalization procedure (Gram-Schmidt ortogonalization) to calculate the QR reduction, however the Householder reduction we looked at earlier is more stable numerically

Let's look at the iteration eq. again:

$$AQ^{(k)} = Q^{(k+1)}R_{k+1} \tag{42}$$

:
As $Q^{(k+1)}$ is orthogonal and R_{k+1} is upper triagonal, this is a QR reduction or an orthonormalization of $AQ^{(k)}$.
Thus we can understand the QR iteration like this:

1. Start with n column vectors spanning n dimension space: $Q^{(0)} = 1$
2. Loop: Apply the matrix A to each column vector
3. Make the column vectors orthogonal to each other
4. goto 2

This is what we call a simultaneous power iteration. The ortogonalization step is essential, otherwise all vectors will converge to the eigenvector of the largest eigenvalue.
Thus QR iteration is equivalent to simultaneous power iteration $\rightarrow$ it will find eigenvalues and eigenvectors.

Theorem: Assume A is a nonsingular real matrix with nondegenerate eigenvalues. Let P denote the matrix of eigenvectors:
$A = P\mathrm{diag}(\lambda_1, \ldots, \lambda_n)P^{-1}$, we assume that P^{-1} can be LU decomposed.
Then the sequence A_k generated by the QR algorithm converges to an upper triangular matrix with the eigenvalues on the diagonal.

If A is symmetric than A_k is also symmetric and thus converges to a diagonal matrix and $Q^{(k)}$ converges to the eigenvectors.

Costs: naively it costs $O(n^3)$ operations to calculate the QR reduction in each step of the algorithm.
It makes sense to first transform the matrix to upper Hessenberg form (where everything below the first subdiagonal is zero). For a symmetric matrix this means transforming to a tridiagonal form (nonzero elements only on the diagonal, first sub and superdiagonal)
Upper Hessenberg matrices remain upper Hessenberg after a QR iteration

For any $n \times n$ matrix A there exists a unitary matrix P which is a product of $n-2$ Householder matrices such that $P^{+}AP$ is upper Hessenberg.

When we calculate the QR algorithm on the upper Hessenberg form, we check whether $(A_k)_{n,n-1}$ is small, as it signals the convergence of the eigenvalue in $(A_k)_{n,n}$. (If it converges we can leave out the last row and last line of the matrix and continue with the rest = deflation algorithm)

It can be shown that the convergence of the smallest eigenvalue goes with

$$(A_k)_{n,n-1} = O\left(\left|\frac{\lambda_1}{\lambda_2}\right|^k\right) \tag{43}$$

where λ_1 is the smallest eigenvalue λ_2 is the second smallest.

If we can arrange it such that the smallest eigenvalue is very small, convergence will be faster $\Longrightarrow$ shifted QR
If we know σ to be a good approximation to one of the eigenvalues, $A - \sigma$ will have a small eigenvalue
We proceed with

$$\begin{aligned} A_k - \sigma &= Q_k R_k \\ A_{k+1} &= R_k Q_k + \sigma = Q_k^T (Q_k R_k + \sigma) Q_k = Q_k^T A_k Q_k \end{aligned} \tag{44}$$

This is called QR algorithm with an *explicit shift*.

We can also change the shift at every step:
Let σ_k be a good approximation of the smallest eigenvalue (e.g. $(A_k)_{nn}$) after step k

$$A_k - \sigma_k = Q_k R_k, \quad A_{k+1} = R_k Q_k + \sigma_k \tag{45}$$

Shifted QR is related to the inverse power iteration:
If we know there is an eigenvalue close to μ:

$$|\lambda_i - \mu| < |\lambda_j - \mu| \text{ for } j \neq i \tag{46}$$

We can use power iteration of $(A - \mu \mathbf{1})^{-1}$:

$$\begin{aligned} \mathbf{v}_0 &= \mathbf{v} / \|\mathbf{v}\| \\ \text{solve} \quad (A - \mu \mathbf{1})\mathbf{y_k} &= \mathbf{v}_{k-1} \text{ for } \mathbf{y}_k \\ \mathbf{v}_{k+1} &= \mathbf{y}_k / \|\mathbf{y}_k\| \end{aligned} \tag{47}$$

$\implies$ This converges to $\mathbf{x}_i$ $\implies$ λ_i is found

This is the formalization of the idea "let's read off the eigenvalue from the first component"
Let's take a linear functional on the vector space, which can be represented by a row vector using the scalar product:

$$l(\mathbf{v}) = \mathbf{l}^T \mathbf{v} \tag{48}$$

(e.g. take the i-th component with $\mathbf{l} = e_i$)
We can use this to keep our vector "normalized":

$$\begin{aligned} \mathbf{v}_0 &= \mathbf{u} \\ \mathbf{w}_{k+1} = A\mathbf{v}_k, \quad n_{k+1} &= l(\mathbf{w}_{k+1}) \\ \mathbf{v}_{k+1} &= \mathbf{w}_{k+1}/n_{k+1} \end{aligned} \tag{49}$$

Similarly to the reasoning to before, One can show that n_{k+1} converges to an eigenvalue and $\mathbf{v}_k$ to an eigenvector, if neither the initial guess nor $\mathbf{l}$ is not orthogonal to the eigenvector
(e.g. with e_1 one keeps the first component of $\mathbf{v}_k$ equal to 1, than one reads off the eigenvalue after the application of the matrix)

In shifted inverse iteration, we should update the shift after every iteration, that should improved convergence.

$$\begin{aligned} \mathbf{y}_{k+1} &= (A - \sigma_k \mathbf{1})^{-1} \mathbf{v}_k \\ n_{k+1} = l(\mathbf{y}_{k+1}), \qquad \mathbf{v}_{k+1} &= \mathbf{y}_{k+1}/n_{k+1} \\ \sigma_{k+1} &= \sigma_k + \frac{1}{n_{k+1}} \end{aligned} \tag{50}$$

σ update comes from:

$$\frac{1}{\lambda - \sigma_k} = n_{k+1} \qquad \Longrightarrow \qquad \lambda = \sigma_k + \frac{1}{n_{k+1}} \tag{51}$$

This is equivalent to the Newton iteration on the system

$$F\left(\begin{bmatrix} \mathbf{u} \\ \lambda \end{bmatrix}\right) = \begin{bmatrix} A\mathbf{u} - \lambda \mathbf{u} \\ \mathbf{l}^{+}\mathbf{u} - 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{52}$$

$\Longrightarrow$ Algorithm is quadratically convergent

We define the Rayleigh quotient for vectors $\mathbf{u}$ and $\mathbf{v}$ which are not orthogonal as $R = \mathbf{v}^+ A\mathbf{u}/\mathbf{v}^+\mathbf{u}$
if $\mathbf{u}$ is an eigenvector of A than R gives the eigenvalue.

We can use the Rayleigh quotient as an approximation to the eigenvalue for the next shift:

$$\begin{aligned} \mathbf{y}_{k+1} &= (A - \sigma_k \mathbf{1})^{-1} \mathbf{v}_k \\ \mathbf{v}_{k+1} &= \mathbf{y}_{k+1} / \left\| \mathbf{y}_{k+1} \right\| \\ \sigma_{k+1} &= \mathbf{v}_{k+1}^+ A \mathbf{v}_{k+1} \end{aligned} \tag{53}$$

If A is symmetric than this algorithm is cubically convergent. (quadratic for non-symmetric matrices)
In the QR iteration we chose $(A_k)_{nn}$ as the next shift value:

$$(A_k)_{nn} = e_n^T A_k e_n = e_n^T Q^{(k)T} A Q^{(k)} e_n = (Q^{(k)} e_n)^T A (Q^{(k)} e_n) = u_n^T A u_n \tag{54}$$

which is the Rayleigh quotient based on the last column vector of $Q^{(k)}$.
$\Longrightarrow$ Shifted QR converges fast.

Summary: QR with explicit shifts for an $n \times n$ matrix A

1. Transform the matrix to upper-Hessenberg form
2. $m = n$ eigenvalues still missing
3. if $\|A_{m,m-1}\| < \epsilon$:
   eigenvalue found: $A_{m,m}$
   delete last row and last column of A
   $m = m - 1$
   stop if all e.v. are found
4. Take as shift the bottom left corner: $\sigma = A_{nn}$
   $A - \sigma = QR, \quad A = RQ + \sigma$
5. goto 3

QR algorithm is the most often used algorithm for dense eigenvalue problems. In practice one uses a slightly more complicated algorithm. (with keywords: implicit (double-) shifts, bulge chasing, agressive early deflation, blocked versions, ...)