# NUMERICAL LINEAR ALGEBRA

Prof. Paola Antonietti

MOX - Dipartimento di Matematica

Politecnico di Milano

**https://antonietti.faculty.polimi.it**

TA: Dr. Michele Botti

## P3: Solving large scale eigenvalue problems

# Eigenvalue problems

The algebraic eigenvalue problem reads as follows:

Given a matrix $A \in \mathbb{C}^{n\times n}$, find $(\lambda, \mathbf{v}) \in \mathbb{C} \times \mathbb{C}^n \backslash \{\mathbf{0}\}$ such that

$$A\mathbf{v} = \lambda\mathbf{v}$$

where

$\lambda$ is an eigenvalue of $A$

$\mathbf{v}$ (non- zero) is the corresponding eigenvector

- The set of all the eigenvalues of a matrix A is called the spectrum of A ($\sigma(A)$)
- The maximum modulus of all the eigenvalues is called the spectral radius of A:

$$\rho(A) = \max\{\,|\lambda| : \lambda \in \lambda(A)\}$$

Exercise. Take $A = \begin{pmatrix} 1 & 2 \\ 8 & 1 \end{pmatrix}$ and show that $\lambda = 5$ is an eigenvalue and $\mathbf{v} = (1,2)^T$ is the corresponding eigenvector.

# Some applications

Eigenvalue problems occur in many areas of science and engineering.

1. Communication systems: Eigenvalues were used by C. Shannon to determine the theoretical limit to how much information can be transmitted through a communication medium like your telephone line or through the air.
2. Designing of bridges:The natural frequency of the bridge is the eigenvalue of smallest magnitude of a system that models the bridge. The engineers exploit this knowledge to ensure the stability of their constructions.
3. Designing car stereo system: Eigenvalue analysis is also used in the design of the car stereo systems, where it helps to reproduce the vibration of the car due to the music.
4. Geophysics: Oil companies use eigenvalue analysis for oil extraction. Oil other substances all give rise to linear systems which have different eigenvalues, so eigenvalue analysis can give a good indication of where oil reserves are located.

# Geometric interpretation

Eigenvalues and eigenvectors provide a means of understanding the complicated behavior of a general linear transformation by decomposing it into simpler actions

An eigenvector (corresponding to a real nonzero eigenvalue) points in a direction in which it is stretched by the linear transformation; the associated eigenvalue is the factor by which it is “stretched/contracted”.

# Mathematical background

1. The problem $A\mathbf{v} = \lambda\mathbf{v}$ is equivalent to $(A - \lambda I)\mathbf{v} = 0$.
2. This homogeneous equation has a nonzero solution $\mathbf{v}$ if and only if its matrix is singular , that is the eigenvalues of $A$ are the values $\lambda$ such that $\det(A - \lambda I) = 0$
3. $\det(A - \lambda I) = 0$ is a polynomial of degree $n$ in $\lambda$: it is called the characteristic polynomial of $A$ and its roots are the eigenvalues of $A$

# Some useful remarks

- From the Fundamental Theorem of Algebra, an $n \times n$ matrix $A$ always has $n$ eigenvalues $\lambda_i, i = 1, \ldots, n$
- Each $\lambda_i$ may be real but in general is a complex number
- The eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ may not all have distinct values
- Rayleigh quotient: Let ($\lambda_i$, $\mathbf{v}_i$) be an eigenpair of $A$, then

$$\lambda_i = \frac{\mathbf{v}_i^H A \mathbf{v}_i}{\mathbf{v}_i^H \mathbf{v}_i}$$

# Similarity transformations

We first need to identify:
- what types of transformations preserve eigenvalues
- for what types of matrices the eigenvalues are easily determined

Definition. The matrix $B$ is similar to the matrix $A$ if there exists a nonsingular matrix $T$ such that $B = T^{-1}AT$.

With the above definition, it is trivial to show that

$$B\mathbf{y} = \lambda\mathbf{y} \rightarrow T^{-1}1ATy = \lambda\mathbf{y} \rightarrow A(T\mathbf{y}) = \lambda(T\mathbf{y})$$

so that $A$ and $B$ have the same eigenvalues, and if $\mathbf{y}$ is an eigenvector of $B$, then $\mathbf{v} = T\mathbf{y}$ is an eigenvector of $A$

# Similarity transformations

Similarity transformations:
- Preserve eigenvalues
- Do not preserve eigenvectors (but the eigenvectors can be easily recovered)

Note that the converse is not true: two matrices that have the same eigenvalues are not necessarily similar!

# Similarity transformations

- The eigenvalues of a diagonal matrix are its diagonal entries
- The eigenvalues of a triangular matrix are also the diagonal entries

Note that:

- Diagonal form simplifies eigenvalue problems for general matrices by similarity transformations
- Some matrices cannot be transformed into diagonal form by a similarity transformation

A square matrix $A$ is called diagonalisable (or non-defective) if it is similar to a diagonal matrix

# The general idea from the numerical viewpoint

Some of numerical methods for computing eigenvalues and eigenvectors are based on reducing the original matrix to a simpler form, whose eigenvalues and eigenvectors are can be easily determined.

Ideally we would like to transform the underlying system of equations into a special set of coordinate axes in which the matrix is diagonal. The eigenvalues are therefore entries of the diagonal matrix and the eigenvectors are the new set of coordinate axes

# Computing eigenvalues and eigenvectors

There are several methods designed to compute all of the eigenvalues of a matrix (and some of them require a great deal of work)

In practice, one may need only one or a few eigenvalues and corresponding eigenvectors (take advantage of this in designing the numerical scheme)

The simplest method for computing a single eigenvalue and eigenvector of a matrix is the so called power method.

# Power method

# The power method

Assume that the matrix $A$ has a unique eigenvalue $\lambda_1$ of maximum modulus, i.e.

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \ldots |\lambda_n|$$

with corresponding eigenvector $\mathbf{v}_1$

Starting from a given nonzero vector $\mathbf{x}^{(0)}$, such that $\|\mathbf{x}^{(0)}\| = 1$, let us consider the following iteration scheme, for $k \geq 0$:

$$\mathbf{y}^{(k+1)} \leftarrow A\mathbf{x}^{(k)}$$

$$\mathbf{x}^{(k+1)} \leftarrow \frac{\mathbf{y}^{(k+1)}}{\|\mathbf{y}^{(k+1)}\|}$$

$$\nu^{(k+1)} \leftarrow [\mathbf{x}^{(k+1)}]^H A\mathbf{x}^{(k+1)}$$

It can be shown that the above iteration scheme converges to a multiple of $\mathbf{v}_1$, the eigenvector corresponding to the dominant eigenvalue $\lambda_1$

# Proof (hints)

- Observe that since $A$ is diagonalisable, its eigenvector $\mathbf{v}_i$ forms a basis for $\mathbb{C}^n$
- Express the starting vector $\mathbf{x}^{(0)}$ as a linear combination of the eigenvectors
- Do some calculations to obtain

$$Ax^{(k)} = \alpha_1 \lambda_1^k \left( \mathbf{v_1} + \sum_{i=2}^{n} \frac{\alpha_i}{\alpha_1} \left[ \frac{\lambda_i}{\lambda_1} \right]^k \mathbf{v_i} \right)$$

- Use that $\lambda_1$ is a dominant eigenvalue

# Convergence rate of the power method

The convergence rate of the power method depends on the ratio $|\lambda_2|/|\lambda_1|$, where $\lambda_2$ is the eigenvalue having the second - largest modulus

The smaller $|\lambda_2|/|\lambda_1|$ is, the faster the convergence is

Hence the power method will converge

- Quickly if $|\lambda_2|/|\lambda_1|$ is small
- Slowly if $|\lambda_2|/|\lambda_1|$ is close to 1

# Deflation methods

Suppose that an eigenvalue $\lambda_1$ and corresponding eigenvector $\mathbf{v}_1$ for a matrix $A$ have been computed

We can compute additional eigenvalues $\lambda_2, \ldots, \lambda_n$ of $A$, by a process called deflation, which removes the known eigenvalue

**Idea**: constructs a new matrix B with eigenvalues $\lambda_2, \ldots, \lambda_n$, that is deflate the matrix A, removing $\lambda_1$

Then $\lambda_2$ can be obtained by the power method.

# Deflation methods

Let $S$ be any nonsingular matrix such that $S\mathbf{v}_1 = \alpha\mathbf{e}_1$, that is $S$ is a scalar multiple of the first column $\mathbf{e}_1$ of the identity matrix $I$

Then the similarity transformation determined by S transforms A into the form

$$SAS^{-1} = \begin{pmatrix} \lambda_1 & b^T \\ 0 & B \end{pmatrix}$$

For example, good choice for $S$ can be an appropriate Householder transformation (linear transformation that describes a reflection about a plane or hyperplane containing the origin)

We use B to compute next eigenvalue $\lambda_2$ and eigenvector $\mathbf{z_2}$.

Given $\mathbf{z}_2$ eigenvector of $B$, we want to compute the second eigenvector $\mathbf{v}_2$ of the matrix $A$

# Deflation methods

We need to add an element to vector $\mathbf{z}_2$ (that consist of $n-1$ elements), that is $\mathbf{v}_2 = S^{-1}\begin{pmatrix}\alpha \\ \mathbf{z}_2\end{pmatrix}$ $\qquad \alpha = \dfrac{\mathbf{b}^H \mathbf{z}_2}{\lambda_1 - \lambda_2}$

Hence, $\mathbf{v}_2$ is an eigenvector corresponding to $\lambda_2$ for the original matrix A

The process can be repeated to find additional eigenvalues and eigenvectors

# Inverse power method

For some applications, the smallest eigenvalue of a matrix is required rather than the largest

We use the fact that the eigenvalues of $A^{-1}$ are the reciprocals of those of $A$

Hence the smallest eigenvalue of A is the reciprocal of the largest eigenvalue of $A^{-1}$

Starting from a given nonzero vector $\mathbf{q}^{(0)}$, such that $\|\mathbf{q}^{(0)}\| = 1$, let us consider the following iteration scheme, for $k \geq 0$:

$$\text{Solve } A\mathbf{z}^{(k+1)} = \mathbf{q}^{(k)}$$

$$\mathbf{q}^{(k+1)} \leftarrow \frac{\mathbf{z}^{(k+1)}}{\|\mathbf{z}^{(k+1)}\|}$$

$$\sigma^{(k+1)} \leftarrow [\mathbf{q}^{(k+1)}]^H A\mathbf{q}^{(k+1)}$$

# Inverse power method with shift

If we want to approximate the eigenvalue $\lambda$ of $A$ which is the closest to a given number $\mu \notin \sigma(A)$.

We denim $M_\mu = A - \mu I$ and observe that the eigenvalue $\lambda$ of $A$ which is the closes to $\mu$ is the minimum eigenvalue of $M_\mu$

Starting from a given nonzero vector $\mathbf{q}^{(0)}$, such that $\|\mathbf{q}^{(0)}\| = 1$, let us consider the following iteration scheme, for $k \geq 0$:

$$\text{Solve } M_\mu \mathbf{z}^{(k+1)} = \mathbf{q}^{(k)}$$

$$\mathbf{q}^{(k+1)} \leftarrow \frac{\mathbf{z}^{(k+1)}}{\|\mathbf{z}^{(k+1)}\|}$$

$$\nu^{(k+1)} \leftarrow [\mathbf{q}^{(k+1)}]^H A \mathbf{q}^{(k+1)}$$

# QR Factorization

# Projectors and complementary projectors

- A *projector* is a square matrix $P \in \mathbb{R}^{n \times n}$ that satisfies $P^2 = P$

- If $\mathbf{w} \in \text{range}(P)$, then $P\mathbf{w} = \mathbf{w}$. Indeed, since $\mathbf{w} \in \text{range}(P)$, then $\mathbf{w} = P\mathbf{z}$, for some $\mathbf{z}$. Therefore:

$$P\mathbf{w} = P(P\mathbf{z}) = P^2\mathbf{z} = P\mathbf{z} = \mathbf{w}$$

- The matrix $I - P$ is the complementary projector to $P$
- $I - P$ projects on the nullspace of $P$: if $P\mathbf{w} = 0$, then $(I - P)\mathbf{w} = \mathbf{w}$, so $\text{null}(P) \subseteq \text{range}(I - P)$
- But for any $\mathbf{w}$, $(I - P)\mathbf{w} = \mathbf{w} - P\mathbf{w} \in \text{null}(P)$, so $\text{range}(I - P) \subseteq \text{null}(P)$
- Therefore

$$\text{range}(I - P) = \text{null}(P)$$

and

$$\text{null}(I - P) = \text{range}(P)$$

# Orthogonal Projectors

- A projector $P$ is orthogonal i f $P = P^2 = P^T$

# The QR factorisation - Main idea

- Find orthonormal vectors $[\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n]$ that span the successive spaces spanned by the columns of $A = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n]$:

$$< \mathbf{a}_1 > \subseteq < \mathbf{a}_1, \mathbf{a}_2 > \ldots \subseteq < \mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n >$$

- This means that (for full rank $A$)

$$< \mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_j > = < \mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_j > \qquad \forall j = 1, \ldots, n$$

# The QR Factorization - matrix form

In matrix form $< \mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_j > = < \mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_j > \qquad \forall j = 1, \ldots, n$

becomes

$$\left[\mathbf{a}_1 \,\middle|\, \mathbf{a}_2 \,|\cdots|\, \mathbf{a}_n\right] = \left[\mathbf{q}_1 \,\middle|\, \mathbf{q}_2 \,|\cdots|\, \mathbf{q}_n\right] \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & \vdots \\ 0 & 0 & \ddots & r_{nn} \end{bmatrix}$$

that is

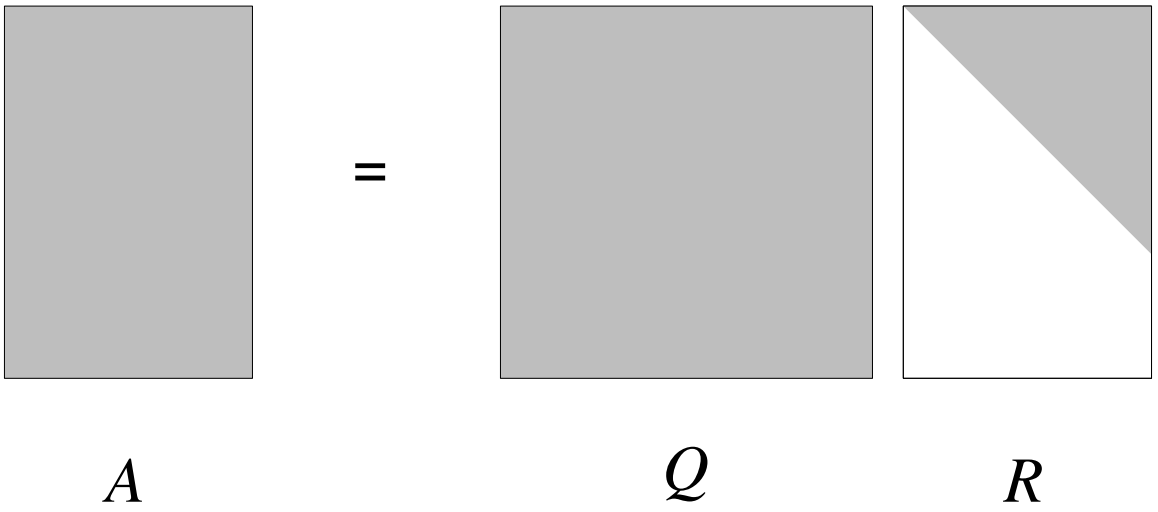
$$A = \hat{Q}\hat{R}$$

This is called the reduced QR factorization

# The Full QR Factorization

Let $A$ be an $m \times n$ matrix. The full QR factorization of $A$ is the factorization $A = QR$, where

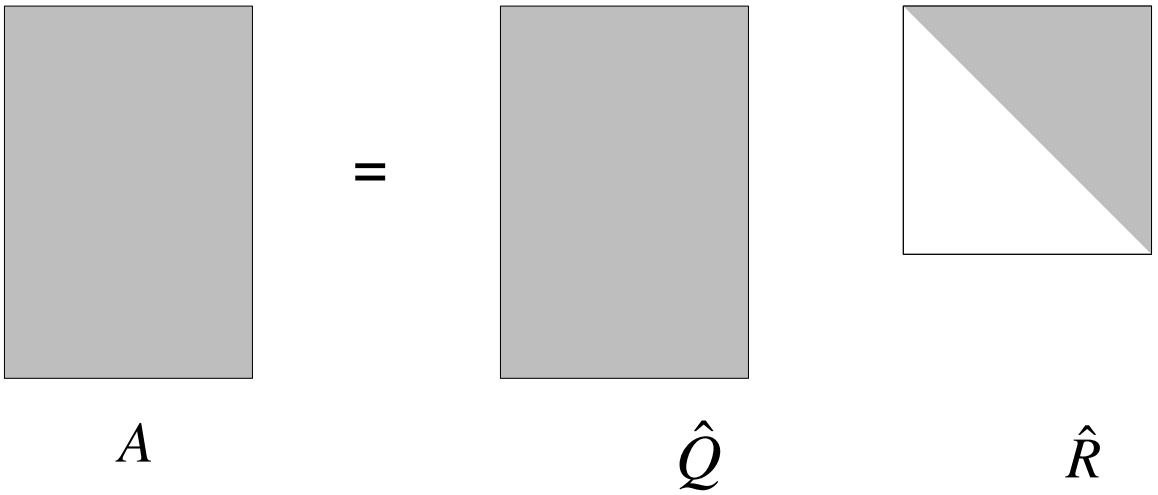
$Q$ is $m \times m$ orthogonal ($QQ^T = I$)
$R$ is $m \times n$ upper-trapeziodal



# The reduced QR factorization

Let $A$ be an $m \times n$ matrix. The reduced QR factorization of $A$ is the factorization $A = \hat{Q}\hat{R}$, where

$Q$ is $m \times n$
$R$ is $n \times n$ upper-triangular



# Gram-Schmidt orthogonalisation

- Given $\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_n$ (the columns of $A$)

- Find new $\mathbf{q}_j$ (the $j$-th column of $\hat{Q}$) orthogonal to $\mathbf{q}_1, \ldots, \mathbf{q}_{j-1}$ by subtracting components along previous vectors

$$\mathbf{w}_j = \mathbf{a}_j - \sum_{k=1}^{j-1} (\overline{\mathbf{q}}_k^T \mathbf{a}_j)\mathbf{q}_k$$

- Normalize to get $\mathbf{q}_j = \dfrac{\mathbf{w}_j}{\|\mathbf{w}_j\|}$

- We then obtain a reduced QR factorization $A = \hat{Q}\hat{R}$, with

$$r_{ij} = \overline{\mathbf{q}}_i^T \mathbf{a}_j \qquad (i \neq j)$$

and

$$r_{jj} = \|\mathbf{a}_j - \sum_{i=1}^{j-1} r_{ij}\mathbf{q}_i\|$$

- *Numerically unstable (see later Modified Gram-Schmidt)*

# Existence and uniqueness

- Every $A \in \mathbb{C}^{m \times n} (m \geq n)$ has a full QR factorization and a reduced QR factorisation
- Proof. For full rank A, Gram-Schmidt proves existence of $A = \hat{Q}\hat{R}$. Otherwise, when $\mathbf{w}_j = 0$ choose arbitrary vector orthogonal to previous $\mathbf{q}_i$. For full QR, add orthogonal extension to Q and zero rows to R.
- Each $A \in \mathbb{C}^{m \times n} (m \geq n)$ of full rank has *unique* $A = \hat{Q}\hat{R}$ with $r_{jj} > 0$, $j = 1, \ldots, n$. Proof. Again Gram-Schmidt ($r_{jj}$ determines the sign).

# Classical vs. Modified Gram-Schmidt

- Small modification of classical G-S gives modified G-S (but see next slide)
- Modified G-S is numerically stable (less sensitive to rounding errors)

for $j = 1, \ldots, n$

$\mathbf{w}_j = \mathbf{a}_j$

for $i = 1, \ldots, j-1$

$$\begin{cases} r_{ij} = \overline{\mathbf{q}}_i^T \mathbf{a}_j & \text{(GS)} \\ r_{ij} = \overline{\mathbf{q}}_i^T \mathbf{w}_j & \text{(MGS)} \end{cases}$$

$\mathbf{w}_j = \mathbf{w}_j - r_{ij}\mathbf{q}_i$

$r_{jj} = \|\mathbf{w}_j\|$

$$\mathbf{q}_j = \frac{\mathbf{w}_j}{\|\mathbf{w}_j\|}$$

end

end

# FLOP Counts - MGS

- No distinction between real and complex
- No consideration of memory accesses or other performance aspects
- Flops count $\sim 2mn^2$

# The QR algorithm

- Basic QR algorithm
- Hessenberg QR algorithm
- QR algorithm with shifts
- Double step QR algorithm for real matrices

# Schur decomposition

If $A \in \mathbb{C}^{n\times n}$ then there is a unitary matrix $U \in \mathbb{C}^{n\times n}$ such that

$$U^H A U = T$$

is upper triangular. The diagonal elements of $T$ are the eigenvalues of $A$.

$U = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n]$ are called Schur vectors. They are in general **not** eigenvectors.

# Schur vectors

The $k$-th column of $U^H A U = T$ read

$$A\mathbf{u}_k = \lambda_k \mathbf{u}_k + \sum_{i=1}^{k-1} t_{ik}\mathbf{u}_i$$

that is

$$A\mathbf{u}_k \in \text{span}\left\{\mathbf{u}_1, \ldots, \mathbf{u}_k\right\} \qquad \forall k$$

- The first Schur vector $\mathbf{u}_1$ is an eigenvector of $A$.
- The first $k$ Schur vectors $\mathbf{u}_1, \ldots, \mathbf{u}_k$ form an invariant subspace for A.
- The Schur decomposition is not unique.

# Basic QR algorithm

Let $A \in \mathbb{C}^{n \times n}$. The QR algorithm computes an upper triangular matrix $T$ and a unitary matrix $U$such that $A = UTU^H$ is the Schur decomposition of $A$.

1. Set $A^{(0)} = A, U^{(0)} = I$
2. while(STOPPING CRITERIA)
3. $A^{(k-1)} = Q^{(k)}R^{(k)}$ (QR factorisation of $A^{(k-1)}$)
4. $A^{(k)} = R^{(k)}Q^{(k)}$
5. $U^{(k)} = U^{(k-1)}Q^{(k)}$ (Update transformation matrix)
6. end for
7. Return $T = A^{(k)}, U = U^{(k)}$.

# Basic QR algorithm: some remarks

1. Notice that $A^{(k)} = R^{(k)}Q^{(k)} = [Q^{(k)}]^H A^{(k-1)}Q^{(k)}$, and therefore $A^{(k)}$ and $A^{(k-1)}$ are similar
2. Moreover, from the above observation, we have

$$\begin{aligned} A^{(k)} &= [Q^{(k)}]^H A^{(k-1)}Q^{(k)} \\ &= [Q^{(k)}]^H [Q^{(k-1)}]^H A^{(k-2)}Q^{(k-1)}Q^{(k)} \\ &= \dots \\ &= [Q^{(k)}]^H \dots [Q^{(1)}]^H A^{(0)}Q^{(1)} \dots Q^{(k)} \end{aligned}$$

# Basic QR algorithm: convergence

Let us assume that all the eigenvalues are isolated, i.e.,

$$|\lambda_1| > |\lambda_2| > \ldots > |\lambda_n|.$$

Then the elements of $A^{(k)}$ below the diagonal converge to zero, i.e.

$$\lim_{k\to\infty} a_{ij}^{(k)} = 0 \quad \forall i > j$$

Moreover, it can be shown that

$$a_{ij}^{(k)} = O\left(\left|\frac{\lambda_i}{\lambda_j}\right|^k\right) \qquad i > j$$

Remarks:

Convergence is low if the eigenvalues are close.

Computational costs: $O(n^3)$

Stopping criteria:

# Basic QR algorithm: convergence

Let us assume that all the eigenvalues are isolated, i.e.,

$$|\lambda_1| > |\lambda_2| > \ldots > |\lambda_n|.$$

Then the elements of $A^{(k)}$ below the diagonal converge to zero, i.e.

$$\lim_{k\to\infty} a_{ij}^{(k)} = 0 \quad \forall i > j$$

Moreover, it can be shown that

$$a_{ij}^{(k)} = O\left(\left|\frac{\lambda_i}{\lambda_j}\right|^k\right) \qquad i > j$$

Remarks:

Convergence is low if the eigenvalues are close.

Computational costs: $O(n^3)$

Stopping criteria:

# QR algorithm - remarks

The basic QR algorithm can be used to compute eigenvalues, but

1. it is computationally expensive (requiring $O(n^3)$ operations per iteration)
2. it can have a very slow convergence depending on the eigenvalues of $A$

There are approaches to improve the situation:

- Reduce the matrix $A$ to a similar matrix that is upper Hessenberg. Notice that Hessenberg structure is preserved by the QR algorithm (see later). This reduces the cost per iteration to $O(n^2)$ operations.
- Once an eigenvalue has been computed deflate away this matrix. This greatly speeds up later eigenvalue computations.
- Use "shifts" in the QR algorithm.

# QR iteration on Hessenberg matrices

A matrix $H \in \mathbb{C}^{n\times n}$ is called a Hessenberg matrix if its elements below the lower off-diagonal are zero

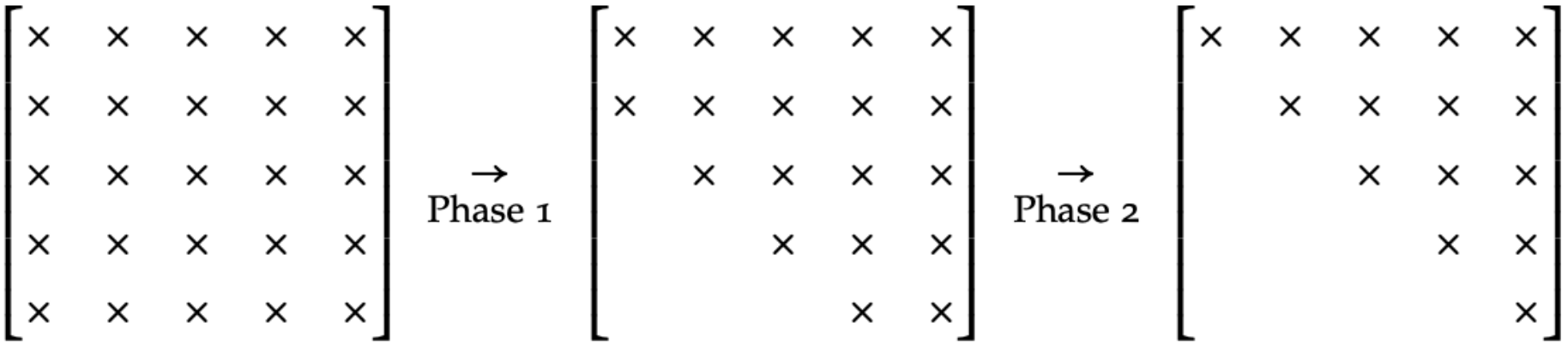
$$h_{ij} = 0, \quad i > j + 1.$$

$$H = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix}$$

A QR iteration on a Hessenberg matrix $H$ costs only $O(n^2)$ flops and the resulting matrix is again a Hessenberg matrix.

# Hessenberg QR-method

To improve the QR-method we make use of an algorithm consisting of two phases:




- **Phase 1**. Compute a Hessenberg matrix $H$ (and an orthogonal matrix $U$) such that $A = UHU^H$. Such a reduction can be done with a finite number of operations.
- **Phase 2**. Apply the basic QR-method to the matrix $H$. It turns out when applying the basic QR-method to a Hessenberg matrix such that the complexity of one step is$O(n^2)$, instead of $O(n^3)$ in the basic version.

# The Lanzos algorithm

# The Lanzos algorithm

The Lanczos algorithm can be used to efficiently find the extremal eigenvalues (maximum and minimum) of a symmetric matrix $A$ of size $n \times n$.

It is based on computing the following decomposition of $A$:

$$A = QTQ^T$$

where $Q$ is an orthonormal basis of vectors $\mathbf{q}_1, \ldots, \mathbf{q}_n$ and $T$ is tri-diagonal:

$$Q = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n] \qquad T = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & \ldots & 0 \\ \beta_1 & \alpha_2 & \beta_2 & \ldots & 0 \\ 0 & \ddots & \ddots & \vdots & 0 \\ 0 & \ddots & \ddots & \vdots & \beta_{n-1} \\ 0 & \ldots & 0 & \beta_{n-1} & \alpha_n \end{bmatrix}$$

The decomposition always exists and is unique provided that $\mathbf{q}_1$ has been specified.

# Lanczos Iterations

We know that $T = Q^T A Q$ which gives

$$\alpha_k = \mathbf{q}_k^T A \mathbf{q}_k \qquad \beta_k = \mathbf{q}_{k+1}^T A \mathbf{q}_k$$

The full decomposition is obtained by imposing $AQ = QT$:

$$[A\mathbf{q}_1, A\mathbf{q}_2, \ldots, A\mathbf{q}_n] = \left[\alpha_1\mathbf{q}_1 + \beta_1\mathbf{q}_2, \beta_1\mathbf{q}_1 + \alpha_2\mathbf{q}_2 + \beta_2\mathbf{q}_3, \ldots, \ldots, \beta_{n-1}\mathbf{q}_{n-1} + \alpha_n\mathbf{q}_n\right]$$

# Lanczos algorithm

At iteration $k$ the algorithm generates intermediate matrices $Q_k$ and $T_k$ that satisfy $T_k = Q_k^T A Q_k$

$$\mathbf{Q}_k = [\ \mathbf{q}_1 \quad \mathbf{q}_2 \quad \cdots \quad \mathbf{q}_k\ ],$$

$$\mathbf{T}_k = \begin{bmatrix} \alpha_1 & \beta_1 & 0 & \cdots & 0 \\ \beta_1 & \alpha_2 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & \beta_k \\ 0 & \cdots & 0 & \beta_k & \alpha_k \end{bmatrix}$$

**Lanczos algorithm**

$\mathbf{r}_0 = \mathbf{q}_1$; $\mathbf{q}_0 = \mathbf{0}$; $\beta_0 = 1$;

for $(k = 1, \ldots, n)$

- if $(\beta_{k-1} = 0)$
  - break;
- end
- $\mathbf{q}_k = \mathbf{r}_{k-1}/\beta_{k-1}$;
- $\alpha_k = \mathbf{q}_k^T \mathbf{A} \mathbf{q}_k$;
- $\mathbf{r}_k = (\mathbf{A} - \alpha_k)\mathbf{q}_k - \beta_{k-1}\mathbf{q}_{k-1}$;
- $\beta_k = |\mathbf{r}_k|$;

end

Remark 1: $\mathbf{q}_1$ is set randomly.

Remark 2: the (orthonormal) vectors $\mathbf{q}_k$ are called the Lanczos vectors.

# Properties of $\mathbf{q}_k$ and $T_k$

At iteration $k$, the $k-$th Lanczos vector $\mathbf{q}_k$ is proven to maximise the l.h.s. of

$$\max_{\mathbf{y}\neq\mathbf{0}} \frac{\mathbf{y}^T(Q_k^T A Q_k)\mathbf{y}}{\mathbf{y}^T\mathbf{y}} = \lambda_1(T_k) \leq \lambda_1(A) = \lambda_1(T)$$

and to simultaneously minimize the l.h.s. of

$$\min_{\mathbf{y}\neq\mathbf{0}} \frac{\mathbf{y}^T(Q_k^T A Q_k)\mathbf{y}}{\mathbf{y}^T\mathbf{y}} = \lambda_n(T_k) \geq \lambda_n(A) = \lambda_n(T)$$

where $\lambda_1(A)$ and $\lambda_n(A)$ are the maximum and the minimum eigenvalue of A, respectively.

The extremal eigenvalues of $T_k$ progressively become more similar to the ones of A.

# Summary

- The Lanczos algorithm can be used to compute the extremal eigenvalues of a symmetric matrix A.
- The Lanczos algorithm only requires matrix-vector multiplications with respect to A (matrix free, very useful if A has a sparse form).
- The algorithm is very sensitive to round-off. The Lanczos vectors $\mathbf{q}_k$ loose orthogonality.
- The Lanczos algorithm can be used to efficiently find a low-rank approximation of A.