3. Eigenvalues: Theory and Algorithms
   - Introduction
   - Eigenvalues and Eigendecompositon
   - Eigenvalue Algorithms: Solving the eigenvalue problem
   - Example: The PageRank Algorithm from Google

Recommended reading:

- Lectures 24, 25, 27 in [4]
- Sections I.6 in [3]
- Sections 6.1, 6.2, 6.4 in [2]
- Kapitel 7 in [1]

Literature:

[1] R. Rannacher.
Numerik 0 - Einführung in die Numerische Mathematik.
Heidelberg University Publishing, 2017.

[2] G. Strang.
Introduction to Linear Algebra.
Wellesley-Cambridge Press, 2003.

[3] G. Strang.
Linear Algebra and Learning from Data.
Wellesley-Cambridge Press, 2019.

[4] L.N. Trefethen and D. Bau.
Numerical linear algebra.
SIAM, Soc. for Industrial and Applied Math., Philadelphia, 1997.

3 Eigenvalues: Theory and Algorithms

3.1 Introduction

3.2 Eigenvalues and Eigendecompositon

Definition 3.2 (*Eigenvalues and -vectors*) *Let $A \in \mathbb{F}^{n\times n}$ be a matrix. A number $\lambda \in \mathbb{C}$ is called an **eigenvalue** of A, if*

$$\exists v \in \mathbb{F}^n, v \neq 0\colon \ Av = \lambda v.$$

*In that case, v is called an **eigenvector** and (λ, v) an **eigenpair**. The set of all eigenvalues is denoted by*

$$\sigma(A) := \{\lambda \in \mathbb{C}\colon \lambda \text{ is eigenvalue of } A\}$$

*and called the **spectrum of** A.*

1) Assume we knew an eigenvalue λ:

2) Assume we had an eigenvector v:

The determinant and eigenvalues

Let $A \in \mathbb{F}^{n \times n}$. Then:

Lemma 3.4 (*Matrix and Eigenvalue Properties*)

i) *Power of a matrix:* $A \in \mathbb{F}^{n\times n}$, $\lambda \in \sigma(A) \;\Rightarrow\; \lambda^k \in \sigma(A^k)$ *for any* $k \in \mathbb{N}$

ii) *Inverse matrix:* $A \in GL_n(\mathbb{F})$, $\lambda \in \sigma(A) \;\Rightarrow\; \lambda \neq 0$, $\frac{1}{\lambda} \in \sigma(A^{-1})$

iii) *Scaling:* $A \in \mathbb{F}^{n\times n}$, $\lambda \in \sigma(A) \;\Rightarrow\; \alpha\lambda \in \sigma(\alpha A)$ *for any* $\alpha \in \mathbb{F}$

iv) $A \in \mathbb{F}^{n\times n}$ *hermitian* $(A = A^H) \;\Rightarrow\; \sigma(A) \subset \mathbb{R}$.

v) $Q \in \mathbb{F}^{n\times n}$ *unitary* $(Q^HQ = I)$, $\lambda \in \sigma(Q) \;\Rightarrow\; |\lambda| = 1$

vi) $A \in \mathbb{F}^{n\times n}$ *positive definite (semi-definite)* $(x^HAx > 0\ (\geq 0)) \;\Leftrightarrow\; \forall\lambda \in \sigma(A)\colon\ \lambda > 0\ (\lambda \geq 0)$

vii) *The eigenvalues of an upper (lower) triangular matrix are sitting on the diagonal.*

viii) *Similarity transformation:* $A \in \mathbb{F}^{n\times n}$, $T \in GL_n(\mathbb{F}) \;\Rightarrow\; \sigma(A) = \sigma(T^{-1}AT)$

ix) *Shifts:* $A \in \mathbb{F}^{n\times n}$, (λ, v) *eigenpair of* $A \;\Rightarrow\; \forall s \in \mathbb{F}\colon\ (\lambda + s, v)$ *eigenpair of* $A + sI$

Attention: The following rules do not hold in general:

- $\lambda \in \sigma(A), \mu \in \sigma(B) \;\not\Rightarrow\; (\lambda + \mu) \in \sigma(A + B)$
- $\lambda \in \sigma(A), \mu \in \sigma(B) \;\not\Rightarrow\; (\lambda \cdot \mu) \in \sigma(A \cdot B)$

Proof. Exercise.

Diagonalizing a matrix

Let us first revisit the example from above (see Examples ?? and ??)

In the previous Example **??** the matrix V of eigenvectors turned out to be orthogonal. The next theorem states, that this is true for any real symmetric matrix.

Theorem 3.6 (*Eigendecomposition of real symmetric matrices*) *For any symmetric matrix $A \in \mathbb{R}^{n\times n}$, there is an orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ (i.e., $Q^\top Q = I$) such that*

$$Q^\top A Q = \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix} =: \mathit{diag}(\lambda_1, \ldots, \lambda_n) \quad (= \textit{diagonal matrix})$$

and $\lambda_i \in \mathbb{R}, i \in \{1, \ldots, n\}$, are the eigenvalues of A. The columns of Q are the normalized eigenvectors.

Proof. In the exercises we will prove this statement for the special case that the matrix has n distinct eigenvalues. The general proof is rather technical and can be found in any standard textbook.

$\rightarrow$ **Thus:** "knowing the eigenpairs = knowing the matrix"

An immediate consequence of Theorem 3.6 is this:
Corollary 3.7 *A symmetric matrix $A \in \mathbb{R}^{n\times n}$ is invertible, if and only if all its eigenvalues are nonzero.*

Geometry of the eigendecomposition:

3.3 Eigenvalue Algorithms: Solving the eigenvalue problem

Aim: Solving the ***eigenvalue problem*** defined by

$$\textit{Given } A \in \mathbb{F}^{n\times n}\textit{, find eigenpairs } (\lambda_i, v_i) \textit{ so that, for all } i = 1,\dots,n,$$

$$v_i \neq 0 \quad \textit{and} \quad Av_i = \lambda_i v_i.$$

Sometimes we are only interested in a few eigenpairs (λ_i, v_i) (for example the one with largest eigenvalue in magnitude). In this case we call it a *partial* eigenvalue problem.

Overview

1. **A first naive approach: Direct method**
   $\rightarrow$ only feasible for very small matrices: $n \in \{2,3,4\}$

2. **Partial eigenvalue problem: Simple iterative methods** (here: The Power Method)
   $\rightarrow$ determine a *single* eigenpair

3. **A second advanced approach: General iterative method** (here: The QR algorithm)
   $\rightarrow$ determine *all* eigenpairs

3.3.1 A first naive approach: Direct method

Recipe:

a) Eigenvalues:
Solving **root finding problem** for the characteristic polynomial

$$\chi_A(\lambda) := \det(A - \lambda I) = 0$$

yields the eigenvalues λ_i.

b) Eigenvectors:
Solving the homogeneous **linear system**

$$(A - \lambda_i I)v_i = 0$$

for each distinct λ_i, gives the corresponding eigenvectors v_i (or more precisely, eigenspaces).

Example: $n = 2$

Consider a general (2×2)-matrix $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$.

a) **Root finding problem**:
Above, we have derived a closed formula for the determinant of a (2×2)-matrix, which applied to $A - \lambda I$ gives

$$0 = \chi_A(\lambda) = \det(A - \lambda I) = \det\left(\begin{pmatrix} a-\lambda & b \\ c & d-\lambda \end{pmatrix}\right) = (a-\lambda)(d-\lambda) - cb = \lambda^2 - (a+d)\lambda + (ad - cb)$$

$$\to \lambda_{1/2} = \frac{a+d}{2} \pm \sqrt{\left(\frac{a+d}{2}\right)^2 - (ad - cb)}.$$

b) **Linear system**:
We then have to solve

$$\begin{pmatrix} a-\lambda_i & b \\ c & d-\lambda_i \end{pmatrix} \begin{pmatrix} v_1^i \\ v_2^i \end{pmatrix} \quad \text{for} \quad i = 1, 2.$$

$$\to v^1, v^2$$

Note: For $n = 3$ we can proceed similarly by applying the rule of Sarrus in step a).

Problem:

In practice, for general, potentially very large, matrices the root finding problem is infeasible, because:

$$A \text{ with large dimension } n \quad \Rightarrow \quad \chi_A \text{ high polynomial degree} \quad \Rightarrow \quad \text{high risk of rounding errors}$$

See for example:
`https://en.wikipedia.org/wiki/Root-finding_algorithms#Roots_of_polynomials`

Abel–Ruffini theorem (see related discussion in [4, Theorem 25.1]):
There are no "closed formulas" for the roots of general polynomials with degree higher than 4.

As a consequence:

We cannot solve the eigenvalue problem in finitely many steps.
Instead, any eigenvalue algorithm has to be iterative!

3.3.2 Simple Iterative Method: The Power Iteration

$\rightarrow$ basis for PageRank algorithm from Google and the WTF algorithm from Twitter

Theorem 3.10 (*Convergence of power iteration*) *Let $A \in \mathbb{R}^{n\times n}$ be a matrix with eigenvalues λ_i for $i \in \{1,\dots,n\}$ which satisfy $|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|$ and whose eigenvectors form a basis of $\mathbb{R}^n$. Also, let the sequence of vectors $\{w^k\}_{k=0}^{\infty}$ be defined by the so-called* ***power iteration***

$$w^{k+1} := \frac{Aw^k}{\|Aw^k\|_p}, \; k \geq 0, p \geq 1, \quad \text{with} \quad w^0 \text{ such that } (v^1, w^0)_2 \neq 0,$$

where v^1 is the normalized (i.e., $\|v^1\|_p = 1$) eigenvector corresponding to λ_1. Then, for $k \to \infty$, we find $w^k \longrightarrow \pm v^1$ and also the so-called Rayleigh quotients

$$\mu_k := \frac{(w^k, Aw^k)_2}{(w^k, w^k)_2} \longrightarrow \lambda_1.$$

Proof. See, e.g., [1, Satz 7.3] or [4, Theorem 27.1].

Remark:

- A variant of this approach is given by the so-called **inverse power method**, which can estimate any eigenpair, assumed a good initial guess for the eigenvalue is available.
- The assumption on the eigenvectors is satisfied, e.g., for real symmetric matrices (see Theorem 3.6)
- From the proof idea one finds that the convergence speed is determined by the fraction $\left(\frac{\lambda_2}{\lambda_1}\right)^k$ (potentially very slow)

3.3.3 A second advanced approach: General iterative method

Recall: (Lemma 3.4)

a) **Similar matrices** have the same eigenvalues:

$$\sigma(A) = \sigma(T^{-1}AT) \quad \text{for} \quad T \in GL_n(\mathbb{F}).$$

b) **Simple matrices**: Eigenvalues of an upper triangular matrix U (e.g., a diagonal matrix) are found on its diagonal, i.e.,

$$\sigma(U) = \{u_{11}, \ldots, u_{nn}\}.$$

Recipe:

a) Iteratively applying **similarity transformations** $T_k \in GL_n(\mathbb{F})$ to $A =: A_0$ thereby producing a sequence

$$A_k = T_k^{-1} A_{k-1} T_k.$$

b) Choose T_k so that this sequence converges to a **simple matrix** (triangular or even diagonal)

$$A_\infty := \lim_{k \to \infty} A_k.$$

$\rightarrow$ **Question:** Choice of the T_k's?

Requirements on the transformations T_k:

1. easily constructed from A_{k-1}
2. easy to invert (e.g., orthogonal matrix)
3. $(A_k)_k$ converges to something simple

One Implementation:

a) The QR-**Algorithmn** defines such transformations T_k through

$$A_0 = A$$
$$\textbf{for } k = 1, \ldots, \infty :$$
$$Q_k R_k := A_{k-1}$$
$$A_k := R_k Q_k$$

b) We find: $A_k = \overline{Q}_k^T A \overline{Q}_k \xrightarrow[k\to\infty]{} U$, where U is **(quasi) upper triangular**; given as follows:

Theorem 3.11 (QR *Algorithm)* *Consider a matrix $A \in \mathbb{R}^{n\times n}$ with distinct eigenvalues λ_i for $i = 1, \ldots, n$, i.e., $|\lambda_1| > |\lambda_2| > \ldots > |\lambda_n|$. Then the iterates $A_k \in \mathbb{R}^{n\times n}$ produced by the QR algorithm converge to the diagonal matrix $\Lambda := diag(\lambda_1, \ldots, \lambda_n)$ which consists of the eigenvalues of A, i.e.,*

$$\lim_{k\to\infty} A_k = \Lambda.$$

Proof. See, e.g., [1, Satz 7.8].

Finally: **What about the eigenvectors?**

One can further show that similar to the power iteration, we find that the columns of

$$\overline{Q}_\infty := \lim_{k\to\infty} \overline{Q}_k$$

are normalized eigenvectors of A.

3.3.4 In Practice: Combined Iterative Methods

Problems:

- QR decomposition for general and very large matrices too expensive
- Exact Schur complement is not reached in finitely many steps (= many QR decompositions needed)

However:

- Any matrix can be **reduced** to a Hessenberg matrix (= simple matrix) in *finitely many* steps
- QR decomposition for this type of matrix is cheap

This leads to:

(3) A third state-of-the-art approach: Combined iterative methods

a) **Similarity transformation via reduction** (e.g., Householder, Wilkinson, Givens) to something simple such as Hessenberg or even tridiagonal
($\rightarrow$ finite steps)

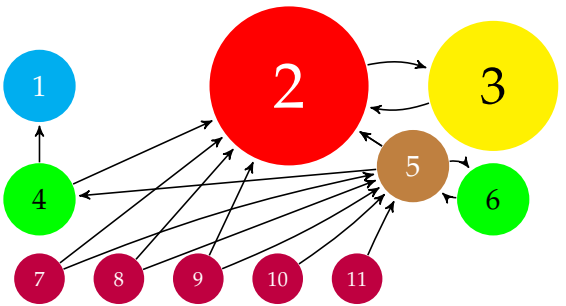
b) **Similarity transformation via iterative method** (e.g., QR or LR algorithm)
($\rightarrow$ potentially infinitely many steps)

Standard: QR Algorithm (with performance optimized modifications (shifts etc...))

c) Determine eigenvalues from the limiting **simple matrix** (and eigenvectors from the similarity transformations).

Common combination in practice: (a) Householder reflection + (b) QR algorithm
$\rightarrow$ Works pretty well for matrices up to 1 mio. columns $n \approx 10^6$
$\rightarrow$ for larger matrices one needs to develop problem-tailored structure exploiting methods

3.4 Example: The PageRank Algorithm from Google



Aim: Rank search enginge results according to the *"importance"* of the web pages.

1998: For this purpose, Larry Page and Sergei Brin develop the PageRank algorithm as the basis of the Google empire.

Assumption: *"important"* means more links from other (important) web pages.

→ More details on the sheet and in the video.