Scientific Computing / Calcolo Scientifico*

Lecture notes
academic year 2023/24

These lecture notes cover the topics of the **Scientific Computing** course taught in the academic year 2023/24, for the Bachelor and Master degrees in Mathematics of the University of Pisa. The reference book for the course is "Applied Numerical Linear Algebra" by Demmel [1].

1 Introduction

The aim of this course is developing effective numerical strategies for the solution of two classes of problems, often encountered in applications:

Linear systems of the form $Ax = b$, where A is either square and invertibile, or given as the least square problem $\min \|Ax - b\|_2$, with A rectangular and not necessarily full rank.

Eigenvalue problems of the form $Av = \lambda v$ for some $v \neq 0$. Sometimes, all the eigenvalues and eigenvectors are sought. In other cases, only some of them are relevant. Examples include the ones with largest or smallest modulus, or enclosed in some region $\Omega \subseteq \mathbb{C}$.

These two problems may look different at first sight, but they share several common features. In both cases we need to differentiate our approach for problems that are "small" or "large". In the former case, so-called **direct methods** will be applicable. In the latter, when the matrix A is so large that it is impossible to store unless it has some particular structure, we will need to employ projection techniques to reduce the dimensionality of the problem. The methods in the latter class are known as **iterative methods**.

This classification may look as an over-simplification of the classes of problems that can be encountered in applications: what about optimization, or the solution of partial differential equations? And nonlinear problems?

A closer look reveals that linear systems and eigenvalue problems are really at the foundation of computational mathematics: optimization leads to KKT conditions (and therefore solving a linear system, or a sequence of them), PDEs are often treated with finite element methods, and requires solving linear systems or computing extremal eigenvalues. Dealing with nonlinear problems with Newton-Rhapson requires solving sequence of linear systems with the Jacobian matrix.

*Revised on May 21, 2024. Mistakes and typos may be reported by writing to stefano.massei@unipi.it or leonardo.robol@unipi.it.

2 Nonsymmetric eigenvalue problems

The (standard) eigenvalue problem can be stated as finding all scalars λ such that $Av = \lambda v$, for some $v \neq 0$; quite often, we are interested in the right or left eigenvectors as well. Since the first lectures in linear algebra, we know that the problem can be recast as computing the roots of the characteristic polynomial:

$$p(\lambda) := \det(\lambda I - A).$$

This characterization may lead to a first tentative algorithm for computing the eigenvalues of a matrix A:

1. Determine the polynomial $p(\lambda)$ by computing the determinant (this is doable via a variant of the LU factorization).
2. Use some functional iteration for computing all the roots.
3. If the eigenvectors are needed as well, compute them by finding a basis for the kernel of $A - \lambda I$.

This approach, while theoretically sound, has several "numerical" shortcomings.

We can implement this "poor man's eigenvalue solver" in a few lines of MATLAB, taking advantage of the functions `poly` and `roots`, which compute the characteristic polynomial of a matrix A and its roots, respectively.

```
function [e] = poor_mans_eig(A)
  % Coefficients of the characteristic polynomial
  p = poly(A);
  % Compute the roots
  e = roots(p);
end
```

For now, we ignore the details on how MATLAB computes the roots of $p(\lambda)$: it is enough to know that the computation is carried out as stably and accurately as possible (we'll get back to this later). Let us test our implementation on a simple example.

```
>> A = diag(1 : 20); % The eigenvalues are the integers from 1 to 20
>> e = poor_mans_eig(A)

e =

   20.0000
   18.9994
   18.0033
   16.9882
   16.0262

   [...]
```

Arguably, this is the simplest possible eigenvalue problem: we are computing the eigenvalues of a diagonal matrix, which can just be read off the diagonal. How can our method be so inaccurate?

The answer to this matter is subtle yet fundamental for the development of stable numerical methods. What we are doing is transforming one problem into another (an eigenvalue problem into a polynomial rootfinding one), through some map Γ:

$$\begin{array}{rccc} \Gamma: & \mathbb{C}^{n\times n} & \to & \mathbb{C}[\lambda] \\ & \cup & & \cup \\ & A & \mapsto & \det(\lambda I - A) \end{array}.$$

We cannot guarantee that small perturbations in the inputs data of one problem will correspond to small perturbations in the input data of the other: small variations in the coefficients of $p(\lambda)$ may cause large changes in the entries of the original matrix A.

Since we work with floating point arithmetic, introducing roundoff errors is inevitable: we need to make sure that any algorithm that we develop is stable under perturbations, and therefore construct a meaningful perturbation theory to analyze them.

2.1 Perturbation theory for eigenvalue problems

We shall now study the effect of perturbations on the spectra of matrices. This topic is closely related with the condition number.

Definition 2.1.1. Let A be an $n \times n$ matrix, and λ an eigenvalue in $\Lambda(A)$; then, the *condition number of* λ, denoted by $\kappa(A, \lambda)$, is defined by

$$\kappa(A,\lambda) := \lim_{h\to 0} \frac{\sup_{\|\delta A\|\leq h} \min\Big\{|\mu - \lambda| \mid \mu \in \Lambda(A+\delta A)\Big\}.}{h}$$

In general, the condition number can be finite or infinite. Note that the definition of condition number depends on the choice of norm. Often this will be the spectral norm, for which we use the notation $\kappa_2(A, \lambda)$.

Whenever the dependency of λ on the entries of A is sufficiently regular, it can be expressed as the norm of the gradient of such map. Unfortunately, such global regularity of the "eigenvalue functions" cannot be guaranteed, and we can only obtain the continuity.

Theorem 2.1.2. *Let A be an $n \times n$ complex matrix. Then, there exists n continuous functions $\lambda_i : \mathbb{C}^{n\times n} \to \mathbb{C}$ such that*

$$\Lambda(A+\delta A) = \Big\{\lambda_1(A+\delta A), \ldots, \lambda_n(A+\delta A)\Big\}$$

Proof. We start by noting that $p(\lambda) := \det(\lambda I - A)$ has coefficients which are continuous functions of the eigenvalues of A. Hence, it suffices to prove that the roots of $p(\lambda)$ are continuous functions of its coefficients.

Let $\lambda_1, \dots, \lambda_r$ be the eigenvalues of A, with their multiplicities m_i. Select some $\epsilon > 0$ small enough for the sets $B(\lambda_i, \epsilon)$ to be disjoint; in particular this implies that $p(\lambda)$ does not vanish on the boundary of $\partial B(\lambda_i, \epsilon)$. Thanks to the residue theorem we have

$$m_i := \frac{1}{2\pi i} \int_{\partial B(\lambda_i, \epsilon)} \frac{p'(z)}{p(z)}\, dz, \tag{2.1}$$

and the function p'/p is a continuous and bounded function of z and the coefficients of $p(z)$ over the compact set $\mathcal{S}_\epsilon := \cup_{i=1}^r \partial B(\lambda_i, \epsilon)$. Therefore, we can select δ such that for any perturbation δp with norm of the coefficients vector bounded by δ, it holds

$$\max_{z \in \mathcal{S}_\epsilon} \left| \frac{p'(z) + \delta p'(z)}{p(z) + \delta p(z)} \right| \leq \frac{1}{2\epsilon}$$

If we compute the integral formula (2.1) for the perturbed polynomial $p(z) + \delta p(z)$ we have that the number of roots counted with multiplicities inside each $B(\lambda_i, \epsilon)$ cannot change of more than $\frac{1}{2}$. Being an integer, this implies that the number does not change, and therefore the roots cannot escape the balls $B(\lambda_i, \epsilon)$, which concludes the proof. □

Much more can be said on the regularity of the eigenvalue functions. For simple eigenvalues, these are analytic, and only lose regularity when the eigenvalue coalesce. A classical reference on this subject is the book by Kato [5].

We now characterize the condition number for simple eigenvalues.

Theorem 2.1.3. *Let $A \in \mathbb{C}^{n \times n}$, and λ a simple eigenvalue. Then,*

$$\kappa_2(A, \lambda) = \frac{\|v\|_2 \|w\|_2}{|w^* v|},$$

where w and v are the left and right eigenvectors relative to λ, respectively.

Proof. Since the eigenvalue is simple, we can do a first order expansion; assuming that $Av = \lambda v$ we may write

$$(A + \delta A)(v + \delta v) = (\lambda + \delta\lambda)(v + \delta v).$$

Rearranging the addends ignoring second order terms yields

$$\delta A v + A \delta v - \lambda \delta v = \delta\lambda v + \mathcal{O}(\|\delta A\|_2^2).$$

We may now left-multiply by w^*, to obtain

$$\frac{w^* \delta A v}{|w^* v|} = \delta\lambda + \mathcal{O}(\|\delta A\|_2)^2.$$

Taking norms, we have the upper bound

$$|\delta\lambda| \leq \frac{\|v\|_2 \|w\|_2}{|w^* v|} \|\delta A\|_2,$$

and taking the limit $\|\delta A\|_2 \to 0$ yields

$$\kappa_2(A, \lambda) \leq \frac{\|v\|_2 \|w\|_2}{|w^* v|}.$$

To show that the previous inequality is sharp, we need to find an explicit δA for which the equality holds. A direct computation shows that the choice $\delta A = h \frac{wv^*}{\|v\|_2 \|w\|_2}$ works, and concludes the proof by taking the limit for $h \to 0$. □

It is worth mentioning a few examples of eigenvalue condition numbers for special classes of matrices.

- If $A = A^*$ then the left and right eigenvectors coincides, and therefore $\kappa_2(A, \lambda) = 1$.
- If A is a Jordan block, the left and right eigenvectors are orthogonal; although Theorem 2.1.3 does not cover this, a direct application of the formula yields $\frac{1}{0}$, and indeed in this case the condition number is equal to ∞.

Exercise 2.1.4. Prove that if a matrix is normal, i.e. $AA^* = A^*A$, then the condition number of its eigenvalues is equal to 1 (as in the special case of symmetric matrices mentioned above).

Theorem 2.1.5 (Hirsch)**.** *Let $A \in \mathbb{C}^{n \times n}$; then, all eigenvalues λ of A satisfy $|\lambda| \leq \|A\|$, where $\|\cdot\|$ is any subordinate norm.*

We now state a result that bounds the distance between the eigenvalues of A and $A + \delta A$.

Theorem 2.1.6 (Bauer-Fike)**.** *Let $A \in \mathbb{C}^{n \times n}$ be a diagonalizable matrix with eigenvector matrix V:*

$$V^{-1} A V = D = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}.$$

Then, for each $\delta A \in \mathbb{C}^{n \times n}$ and eigenvalue μ of $A + \delta A$, there exists an eigenvalue λ_i that satisfies $|\lambda_i - \mu| \leq \kappa(V) \|\delta A\|$., where $\|\cdot\|$ is any subordinate matrix norm induced by an absolute norm.[1]

Proof. Let $\mu \in \Lambda(A + \delta A)$; if μ is eigenvalue of A, the theorem is trivially true, otherwise we consider the singular matrix

$$V^{-1}(A + \delta A - \mu I)V = (D - \mu I) + V^{-1} \delta A V.$$

Thanks to the nonsingularity of $D - \mu I$, we automatically have that $I + (D - \mu I)^{-1} V^{-1} \delta A V$ is singular, and therefore -1 belongs to the spectrum of $(D - \mu I)^{-1} V^{-1} \delta A V$. Therefore, thanks to Hirsch's theorem we have

$$1 \leq \|(D - \mu I)^{-1} V^{-1} \delta A V\| \leq \|(D - \mu I)^{-1}\| \cdot \kappa(V) \|\delta A\|.$$

[1] An absolute norm is one for which the component-wise property $|x_i| \geq |y_i|$ implies $\|x\| \geq \|y\|$. For such norms we have $\|D\| = \max_i |d_{ii}|$ for any diagonal matrix D.

Since $\|\cdot\|$ is an absolute subordinate norm it holds that

$$\|(D-\mu I)^{-1}\| = \max_{i=1,\dots,n} \frac{1}{|\lambda_i - \mu|} = \frac{1}{\min_{i=1,\dots,n} |\lambda_i - \mu|},$$

which concludes the proof. □

Applying Bauer-Fike's theorem to a normal matrix with the spectral norm yields the upper bound

$$|\lambda_i - \mu| \leq \|\delta A\|_2,$$

since normal matrices are diagonalized by unitary or orthogonal matrices with condition number equal to 1. This result is stronger than the fact that the condition number for such matrices is equal to 1, since there is no first order approximation involved.

2.2 Backward error analysis

The classical definition of stability for an algorithm is that it approximates a function $f(\cdot)$ with a guaranteed accuracy $\|f - \tilde{f}\|_\infty \leq \epsilon$.

If f is the function that associates a matrix with its spectrum, there is little hope to construct an unconditionally stable algorithm in floating point arithmetic. There exist matrices with arbitrarily badly conditioned eigenvalues, for which round-off errors will drive the spectrum far from the true solution.

To solve this problem, it is customary in numerical analysis to look for *backward stable* algorithms. From now on, we use the notation

$$a \lesssim b \iff a \leq Cb, \qquad C \text{ being a "moderate constant".}$$

We do not give a precise definition for C: in the context of numerical linear algebra, it is often allowed to depend polynomially on the size of the problem.

Definition 2.2.1. An algorithm evaluating a function $f(z_1, \dots, z_n)$ is *backward stable* if, for any input $z_1, \dots, z_n$, the algorithm computes $\tilde{f}$ that satisfies:

$$\tilde{f}(z_1, \dots, z_n) = f(z_1 + \delta z_1, \dots, z_n + \delta z_n), \qquad \|\delta z_i\| \lesssim \|z_i\| \cdot \epsilon_m,$$

where ϵ_m is the unit-roundoff.

If we compute the eigenvalues of A using a backward stable algorithm, we obtain the exact eigenvalues of a matrix $A + \delta A$, with $\|\delta A\| \lesssim \|A\|\epsilon_m$. This implies that the computed eigenvalues $\mu_1, \dots, \mu_n$ satisfy

$$|\mu_i - \lambda_i| \lesssim \kappa(A, \lambda)\epsilon_m + \mathcal{O}(\epsilon_m^2)$$

A backward stable algorithm provides accurate eigenvalues, with the maximum accuracy attainable given the inherent difficulty of the underlying problem.

Given a candidate approximate eigenpair (λ, v), we may wish to measure the associated backward error, which can be defined as follows.

Definition 2.2.2. Let $\lambda \in \mathbb{C}$, and $v \in \mathbb{C}^n$. The *backward error of λ, v as an eigenpair of A* is defined as

$$\mathrm{BE}(A, \lambda, v) := \min\{\|\delta A\| \mid (A + \delta A)v = \lambda v\}.$$

Similarly, the *backward error of λ as an eigenvalue of A* is defined as

$$\mathrm{BE}(A, \lambda) := \min\{\|\delta A\| \mid \lambda \in \Lambda(A + \delta A)\}.$$

Clearly, we have $\mathrm{BE}(A, \lambda) \leq \mathrm{BE}(A, \lambda, v)$, for any choice of v. The backward error of the eigenpair can be easily computed a posteriori, in contrast with the forward error.

Theorem 2.2.3. *Let $A \in \mathbb{C}^{n\times n}$ be a square matrix, and λ, v a candidate eigenpair. Then, for the spectral norm $\|\cdot\|_2$,*

$$\mathrm{BE}_2(A, \lambda, v) = \frac{\|Av - \lambda v\|_2}{\|v\|_2}.$$

Proof. Let δA be any perturbation to A such that λ and v that are eigenvalue and eigenvector of $A + \delta A$. Then, we have

$$Av - \lambda v = -\delta A v \implies \|\delta A\|_2 \geq \frac{\|\delta A v\|_2}{\|v\|_2} \geq \frac{\|Av - \lambda v\|_2}{\|v\|_2}.$$

We now show that the equality hold for a specific choice of δA. Let $r := Av - \lambda v$, and set $\delta A := -\frac{rv^*}{\|v\|_2^2}$; then, we have

$$(A + \delta A - \lambda I)v = \underbrace{(A - \lambda I)v}_{r} - r\frac{v^* v}{\|v\|_2^2} = 0.$$

A direct computation shows that $\|\delta A\|_2 = \|r\|_2/\|v\|_2$, concluding the proof. □

A similar characterization can be stated for the backward error of an eigenvalue.

Theorem 2.2.4. *Let $A \in \mathbb{C}^{n\times n}$ be a square matrix, and λ a candidate eigenvalue. Then, for the spectral norm $\|\cdot\|_2$, we have*

$$\mathrm{BE}_2(A, \lambda) = \|(A - \lambda I)^{-1}\|_2^{-1}, \qquad \forall\ \lambda \notin \Lambda(A)$$

Proof. Let δA be a perturbation such that $(A + \delta A)v = \lambda v$. Then, we have

$$(A - \lambda I)v = -\delta A v \implies v = -(A - \lambda I)^{-1}\delta A v \implies \|v\|_2 \leq \|(A - \lambda I)^{-1}\|_2 \|\delta A\|_2 \|v\|_2.$$

Dividing by $\|v\|_2$ yields the upper bound $\|(A - \lambda I)^{-1}\|_2^{-1} \leq \|\delta A\|_2$. The bound holds for any δA such that $\lambda \in \Lambda(A + \delta A)$, so $\|(A - \lambda I)^{-1}\|_2^{-1} \leq \mathrm{BE}_2(A, \lambda)$.

To show the other inequality, consider v and w such that

$$(A - \lambda I)^{-1} v = w, \qquad \|v\|_2 = \|(A - \lambda I)^{-1}\|_2^{-1}, \quad \|w\|_2 = 1.$$

Then, we have $\|(A - \lambda I)w\|_2 = \|(A - \lambda I)^{-1}\|_2^{-1} = \mathrm{BE}_2(A, \lambda, w) \geq \mathrm{BE}(A, \lambda)$, which concludes the proof. □

We have emphasized how transforming an eigenvalue problem into a polynomial rootfinding one is generally a bad idea. The most natural alternative that we will soon pursue is to construct a matrix sequence

$$A_0 := A \to A_1 := F(A_0) \to \ldots \to A_{k+1} = F(A_k) \to \ldots$$

such that all matrices are similar, $\lim_k A_k$ is computable with sufficient accuracy, and the eigenvalues can be read off from the limit. For instance, we may ask for the limit to be upper triangular or diagonal.

For all this to work, we need to make sure that the transformation $A_{k+1} = F(A_k)$ does not make the condition number of the eigenvalues worse. Not all similarities are up to the task, but this is true when we use unitary or orthogonal matrices.

Exercise 2.2.5. Prove that if Q is unitary, then the condition numbers for the eigenvalues of A and QAQ^* coincide, i.e., $\mathrm{BE}_2(A, \lambda) = \mathrm{BE}_2(QAQ^*, \lambda)$ and $\mathrm{BE}_2(A, \lambda, v) = \mathrm{BE}_2(QAQ^*, \lambda, Qv)$.

2.3 The power method

We introduce the first method for computing eigenvalues: the power method. Let A be any matrix. We consider the vector sequence defined, for any choice of v_0, as follows:

$$v^{(k+1)} = \frac{Av^{(k)}}{\|Av^{(k)}\|_2}, \qquad k \geq 0, \qquad \lambda_k = (v^{(k)})^* Av^{(k)} \qquad v^{(0)} \text{ assigned.} \tag{2.2}$$

Up to the normalization factor, the vector $v^{(k)}$ satisfies $v^{(k)} = A^k v^{(0)}$.

Under suitable conditions, the terms $(\lambda_k, v^{(k)})$ converge to a dominant eigenpair of A. Let us make the simplifying assumption that A is diagonalizable, with eigenvalues

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

Then, we can rewrite the iteration as follows

$$w^{(k)} = \gamma_k D^k w^{(0)}, \qquad D := \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}, \qquad \gamma_k := \frac{1}{\|VD^k w^{(0)}\|}$$

This yields the following explicit expression for $w^{(k)}$:

$$w^{(k)} = \gamma_k \lambda_1^k \begin{bmatrix} w_1^{(0)} \\ \left(\frac{\lambda_2}{\lambda_1}\right)^k w_2^{(0)} \\ \vdots \\ \left(\frac{\lambda_n}{\lambda_1}\right)^k w_n^{(0)} \end{bmatrix}$$

Since γ_k is chosen to normalize $v^{(k)}$, we have that if $w_1^{(0)} \neq 0$ all components in $w^{(k)}$ go to zero for $k \to \infty$, and $w^{(k)}$ converges to a multiple to e_1 with rate $(\frac{\lambda_2}{\lambda_1})^k$. Since $v^{(k)} = Vw^{(k)}$, we conclude that $v^{(k)}$ converges to an eigenvector relative to λ_1, and consequently $\lambda^{(k)} = (v^{(k)})^* A v^{(k)}$ converges to λ_1 with the same linear rate. A more formal analysis of the convergence rate will be carried out in Section 2.4.

Note that the condition $w_1^{(0)} \neq 0$ is generic, in the sense that if we choose $x^{(0)}$ randomly with any absolutely continuous probability measure hold with probability 1. In theory, we make an initial choice for which $w_1^{(0)} = 0$, this condition should continue to hold throughout the iterations. However, working in floating point arithmetic will introduce perturbations that will lead us back to generic case.

Remark 2.3.1. The convergence mentioned above for the eigenvectors is to be intended in a broad sense. we can only guarantee that $w^{(k)}$ will converge to a multiple of e_1 normalized to have $\|w^{(k)}\| = 1$, so its sign could still change over $\mathbb{S}^1$; this will not alter its property of being an eigenvector, and thus the statement holds in a strict sense for the eigenvalue approximation $\lambda^{(k)}$.

To summarize, we can implement the power method for a generic matrix A following the pseudocode in Algorithm 1. In this code, the stopping criterion is checked by computing the residual $Av^{(k)} - \lambda v^{(k)}$ at each iteration.

Algorithm 1 Power method applied to a matrix A starting from a vector v. The iteration stops when $\|Av - \lambda v\|$ is smaller than a given tolerance τ.

```
1: procedure PowerMethod(A, v, τ)
2:     λ ← 0
3:     r ← Av
4:     while ‖r‖ > τ do
5:         w ← Av
6:         λ ← v*w
7:         r ← w − λv
8:         v ← w/‖w‖
9:     end while
10:    return (λ, v)
11: end procedure
```

A limitation of the power method is that it can only compute the eigenvalues of largest modulus; if we are interested in the eigenvalues close to a certain point $\sigma \in \mathbb{C}$, we can consider the shifted and inverted matrix $(A - \sigma I)^{-1}$; the largest eigenvalue of this matrix is $\frac{1}{\lambda - \sigma}$, where λ is the eigenvalue closest to the shift.

Hence, the algorithm can be modified as reported in the pseudocode of Algorithm 2. Note that we use an auxiliary vector $\hat{w}$ to check the residual and the approximation of the eigenvalue, since the iteration is now driven by $(A - \sigma I)^{-1}$ instead of A.

Algorithm 2 Shift-and-Invert Power method applied to a matrix A with shift σ, starting from a vector v. The iteration stops when $\|Av - \lambda v\|$ is smaller than a given tolerance τ.

procedure SHIFTANDINVERT(A, v, σ, τ)
    $\lambda \leftarrow 0$
    $r \leftarrow (A - \sigma I)^{-1} v$
    while $\|r\| > \tau$ **do**
        $w \leftarrow (A - \sigma I)^{-1} v$
        $\hat{w} \leftarrow Av$
        $\lambda \leftarrow v^* \hat{w}$
        $r \leftarrow \hat{w} - \lambda v$
        $v \leftarrow w / \|w\|$
    end while
    return (λ, v)
end procedure

2.4 Convergence rate of the power method

We now formally analyze the convergence of the power iteration with respect to the dominant eigenvector v_1. Note that, even in the case that λ_1 is simple, the dominant eigenvector is defined up to a constant; in particular, it has little sense to measure quantities like $\|v_1 - v^{(k)}\|$ as if, for instance, $v^{(k)} \to -v_1$ we would not detect any convergence. The key point is to quantify how collinear are the vectors v_1 and $v^{(k)}$, and this requires to introduce trigonometric functions of an angle between two vectors.

Definition 2.4.1. Given $x, y \in \mathbb{C}^n$, $x \neq 0$, we define the orthogonal projections on the $\mathrm{span}(x)$ and its complement as

$$\Pi_x(y) = \frac{xx^*}{\|x\|_2^2} y, \qquad \Pi_x^\perp(y) = \left(I - \frac{xx^*}{\|x\|_2^2} \right) y.$$

Moreover we define the $\sin, \cos$ and $\tan$ of the angle between x and y as follows:

$$\sin\theta(x,y) = \frac{\|\Pi_x^\perp(y)\|_2}{\|y\|_2} = \frac{\min_{z \in \mathrm{span}(x)} \|y - z\|_2}{\|y\|_2},$$
$$\cos\theta(x,y) = \frac{\|\Pi_x(y)\|_2}{\|y\|_2} = \frac{|x^* y|}{\|x\|_2 \|y\|_2},$$
$$\tan\theta(x,y) = \frac{\sin\theta(x,y)}{\cos\theta(x,y)} = \frac{\|\Pi_x^\perp(y)\|_2}{\|\Pi_x(y)\|_2}.$$

Remark 2.4.2. It holds $\sin\theta(x,y)^2 + \cos\theta(x,y)^2 = 1$, $\forall x, y \in \mathbb{C}^n \setminus \{0\}$.

Remark 2.4.3. Trigonometric functions for vectors are commutative with respect to the two inputs, invariant by scaling, and do not change if we apply the same unitary matrix

to both x and y. In particular, when analyzing the convergence of the power method we can consider the simplified iteration $v^{(k)} = A^k v^{(0)}$ as the normalization step has no influence on the collinearity of the iterate with respect to v_1.

Before stating the main result, let us assume that A is diagonalizable and that $V^{-1}AV = D := \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$. Then, we consider the auxiliary sequence

$$y^{(0)} = V^{-1}v^{(0)}, \qquad y^{(k)} = Dy^{(k-1)} = D^k y^{(0)} \qquad \Longrightarrow \qquad y^{(k)} = V^{-1}v^{(k)},$$

and we analyze how collinear is $y^{(k)}$ with respect to $e_1 = V^{-1}v_1$, that is the dominant eigenvector for D. Block partitioning $y^{(k)}$ and D as

$$y^{(k)} = \begin{bmatrix} y_1^{(k)} \\ y_2^{(k)} \end{bmatrix}, \qquad D = \begin{bmatrix} \lambda_1 & \\ & D_2 \end{bmatrix}, \qquad y_1^{(k)} \in \mathbb{C}, \quad y_2^{(k)} \in \mathbb{C}^{n-1},$$

we see that

$$y^{(k)} = D^k y^{(0)} = \begin{bmatrix} \lambda_1^k y_1^{(0)} \\ D_2^k y_2^{(0)} \end{bmatrix} = \lambda_1^k \begin{bmatrix} y_1^{(0)} \\ \left(\frac{D_2}{\lambda_1}\right)^k y_2^{(0)} \end{bmatrix}.$$

Moreover, we have $\left\| \left(\frac{D_2}{\lambda_1}\right)^k \right\|_2 = \left|\frac{\lambda_2}{\lambda_1}\right|^k$ (since D_2 is diagonal) and

$$\Pi_{e_1}^{\perp}(y^{(k)}) = \begin{bmatrix} 0 \\ y_2^{(k)} \end{bmatrix}, \qquad \Pi_{e_1}(y^{(k)}) = \begin{bmatrix} y_1^{(k)} \\ 0 \end{bmatrix}.$$

Putting all together, we have

$$\tan\theta(e_1, y^{(k)}) = \frac{\|y_2^{(k)}\|_2}{|y_1^{(k)}|} \leq \left|\frac{\lambda_2}{\lambda_1}\right|^k \frac{\|y_2^{(0)}\|_2}{|y_1^{(0)}|} = \left|\frac{\lambda_2}{\lambda_1}\right|^k \tan\theta(e_1, y^{(0)}). \tag{2.3}$$

We are ready to state the main result about the convergence of the power method.

Theorem 2.4.4. *Let $A \in \mathbb{C}^{n\times n}$ be diagonalizable with eigenvector matrix V, and dominant eigenvalue λ_1 such that $|\lambda_1| > |\lambda_2|$. If $v^{(0)} \in \mathbb{C}^n$ is such that $u_1^* v^{(0)} \neq 0$, for a left dominant eigenvector u_1, then the kth iterate of the power method, starting from $v^{(0)}$, verifies*

$$\sin\theta(v_1, v^{(k)}) \leq \kappa(V) \left|\frac{\lambda_2}{\lambda_1}\right|^k \frac{\sin\theta(v_1, v^{(0)})}{\cos\theta(e_1, V^{-1}v^{(0)})}.$$

Proof. Note that, $u_1^* v^{(0)} \neq 0$ implies $\cos\theta(e_1, y^{(0)}) \neq 0$, so that the right-hand-side of (2.3) is well defined. The latter inequality yields

$$\sin\theta(e_1, y^{(k)}) \leq \tan\theta(e_1, y^{(k)}) \leq \left|\frac{\lambda_2}{\lambda_1}\right|^k \tan\theta(e_1, y^{(0)}).$$

Then, we have

$$\begin{aligned}
\sin\theta(v_1, v^{(k)}) &= \sin\theta(Ve_1, Vy^{(k)}) \\
&= \frac{\min_{z\in\mathrm{span}(y^{(k)})} \|Ve_1 - Vz\|_2}{\|Ve_1\|_2} \\
&\leq \frac{\|V\|_2}{\|Ve_1\|_2} \sin\theta(e_1, y^{(k)}) \\
&\leq \frac{\|V\|_2}{\|Ve_1\|_2} \left|\frac{\lambda_2}{\lambda_1}\right|^k \frac{\sin\theta(e_1, y^{(0)})}{\cos\theta(e_1, y^{(0)})} \\
&= \frac{\|V\|_2}{\cos\theta(e_1, V^{-1}v^{(0)})} \left|\frac{\lambda_2}{\lambda_1}\right|^k \frac{\min_{z\in\mathrm{span}(y^{(0)})} \|V^{-1}(Ve_1 - Vz)\|_2}{\|Ve_1\|_2} \\
&\leq \kappa(V) \left|\frac{\lambda_2}{\lambda_1}\right|^k \frac{\sin\theta(v_1, v^{(0)})}{\cos\theta(e_1, V^{-1}v^{(0)})}.
\end{aligned}$$

□

Remark 2.4.5. A few remarks about the assumptions of the previous theorem:

- A diagonalizable can be relaxed to assuming λ_1 simple.
- The condition $u_1^* v^{(0)} \neq 0$ is necessary but also virtually true in practice when generating a random starting guess $v^{(0)}$. Indeed, the set $\{v \in \mathbb{C}^n : \ u_1^* v = 0\}$ has zero Lebesgue measure.
- Also $|\lambda_1| > |\lambda_2|$ can not be removed; take for instance the case $A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ and a starting vector that is not aligned with neither $v_1 = \left[\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right]$ nor $v_2 = \left[\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right]$.

2.5 The Hermitian case

In the case that A is Hermitian we can show that the approximant of the dominant eigenvalue computed by the power method converges with the double decay rate with respect to the general case. To shed some lights about this phenomenon it is insightful to look at the Rayleigh quotient function $\rho_A(x) = \frac{x^* A x}{x^* x}$ that is such that $\rho_A(v_1) = \lambda_1$. If we look at the gradient (considering ρ_A as a function on real vectors) we have

$$\nabla\rho_A(x) = \frac{1}{x^* x}(Ax + A^* x - 2\rho_A(x) \cdot x).$$

In particular, when A is Hermitian v_1 (and any other dominant eigenvector) is a stationary point for ρ_A while it is not when $A \neq A^*$. Therefore, by looking at the Taylor expansion of $\rho_A(x)$ we have

$$|\rho_A(x) - \lambda_1| = |\rho_A(x) - \rho_A(v_1)| = \begin{cases} \mathcal{O}(\|x - v_1\|_2^2) & \text{if } A \text{ Hermitian} \\ \mathcal{O}(\|x - v_1\|_2) & \text{otherwise} \end{cases}.$$

More formally, we prove the following result.

Theorem 2.5.1. *Let $A \in \mathbb{C}^{n\times n}$ be Hermitian with eigenvalues $|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_2| \geq 0$, $v^{(0)} \in \mathbb{C}^n$ such that $v_1^* v^{(0)} \neq 0$. Then, the kth iterate of the power method, starting from $v^{(0)}$, verifies*

$$\tan\theta(v_1, v^{(k)}) \leq \left|\frac{\lambda_2}{\lambda_1}\right|^k \tan\theta(v_1, v^{(0)}),$$

$$|\lambda_1 - \rho_A(v^{(k)})| \leq \max_{j=1,\ldots,n} |\lambda_1 - \lambda_j| \cdot \left|\frac{\lambda_2}{\lambda_1}\right|^{2k} [\tan\theta(v_1, v^{(0)})]^2.$$

Proof. The inequality concerning the eigenvector convergence follows from (2.3) by applying a unitary eigenvector matrix V to the vectors involved in the trigonometric functions in both the left- and the right-hand-side.

To show the second inequality we assume that the normalization step in the power method is not performed and that the starting vector $v^{(0)}$ has been rescaled to obtain $\|v^{(k)}\|_2 = 1$; note that, all these assumptions cause no loss of generality as the Rayleigh quotient is invariant under (nonzero) rescaling of the argument. Let $v^{(0)} = \sum_{j=1}^n a_j v_j$, then we have

$$\rho_A(v^{(k)}) = (v^{(k)})^* A v^{(k)} = \frac{(v^{(0)})^* A^{2k+1} v^{(0)}}{(v^{(0)})^* A^{2k} v^{(0)}} = \frac{\sum_{j=1}^n a_j^2 \lambda_j^{2k+1}}{\sum_{j=1}^n a_j^2 \lambda_j^{2k}}.$$

So that

$$|\lambda_1 - \rho_A(v^{(k)})| = \left|\frac{\sum_{j=2}^n a_j^2 \lambda_j^{2k}(\lambda_j - \lambda_1)}{\sum_{j=1}^n a_j^2 \lambda_j^{2k}}\right| \leq \frac{\max_{j=1,\ldots,n} |\lambda_1 - \lambda_j|}{a_1^2} \sum_{j=2}^n a_j^2 \left|\frac{\lambda_2}{\lambda_1}\right|^{2k}$$

$$= \max_{j=1,\ldots,n} |\lambda_1 - \lambda_j| \left|\frac{\lambda_2}{\lambda_1}\right|^{2k} [\tan\theta(v_1, v^{(0)})]^2.$$

□

2.6 Subspace iteration

As a natural generalization of the power method, we may consider iterating over subspaces instead of vectors. Mathematically, we wish to select an initial subspace $\mathcal{U}_0 \subseteq \mathbb{C}^n$, and then construct a sequence of subspaces as follows:

$$\mathcal{U}_{k+1} := A\mathcal{U}_k = \{Ax \mid x \in \mathcal{U}_k\}.$$

In the case of the vector iteration, we have convergence to an eigenvector; this may be reinterpreted as converging to a basis of a one-dimensional subspace, by setting $\mathcal{U}_k := \text{span}(v_k)$. For subspaces of higher dimension, convergence to an eigenvector is replaced with convergence to an invariant subspace. Recall that, given a linear operator A, an invariant subspace is one satisfying $A\mathcal{U} \subseteq \mathcal{U}$. If U is a matrix whose columns span $\mathcal{U}$, the property of being an invariant subspace of dimension p can be rephrased as

$$AU = UR, \qquad R \in \mathbb{C}^{p\times p}. \tag{2.4}$$

Note that if $Rw = \lambda w$ then Uw is an eigenvector relative to λ for A:

$$AUw = URw = \lambda Uw \implies \lambda \in \Lambda(A).$$

Hence, finding an invariant subspace described as in (2.4) is valuable for computing selected eigenvalues.

Not all bases are numerically suitable for representing subspaces. Given a basis U, we have that any vector in $\mathcal{U}$ can be written as $v = Uw$, where w is the vector of coordinates in the chosen basis:

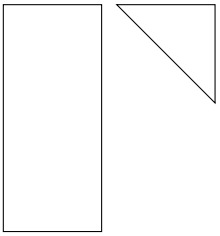
$$v = w_1 u^{(1)} + \ldots + w_k u^{(k)} = \begin{bmatrix} u^{(1)} & \cdots & u^{(k)} \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_k \end{bmatrix}.$$

We need to ensure that small perturbations in the input data for this representation (for instance, the vector w), correspond to small changes in the output (the vector v). A natural choice to achieve this goal is taking U orthogonal. This guarantees

$$\|U(w + \delta w) - Uw\|_2 = \|U\delta w\|_2 = \|\delta w\|_2,$$

thanks to the unitary invariance of the Euclidean norm.

Given any basis matrix V of size $n \times p$, we can always make it orthogonal (or unitary) by computing a *thin* (or economy size) QR factorization:

$V = QR =$  $\implies \text{colspan}(V) = \text{colspan}(Q),$

that holds because $\det(R) \neq 0$, since V is full rank. The matrix Q is computed through a sequence of $p-1$ Householder reflectors, each of them annhiliating subdiagonal entries in the ith column. In detail, we start by determining a reflector $P_1 = I - \beta uu^*$ such that $P_1(Ve_1) = r_{11}e_1$, which yields

$$P_1 V = \begin{bmatrix} r_{11} & \times & \cdots & \times \\ 0 & \times & \cdots & \times \\ \vdots & \vdots & \cdots & \vdots \\ 0 & \times & \cdots & \times \end{bmatrix}.$$

The matrix P is a rank-1 perturbation to a unitary matrix, so the cost of computing $P_1 V$ is $\mathcal{O}(np)$ flops (floating point operations). Then, the remaining columns can be reduced to upper triangular form by computing similar matrices $P_2, \ldots, P_p$, with a total computational cost of $\mathcal{O}(np^2)$ (to be precise, when $p = n$ only $p - 1$ reflectors are necessary, whereas p are needed in all other cases).

Algorithm 3 Subspace iteration for A, starting from an orthogonal matrix $U^{(0)}$

procedure SUBSPACEITERATION(A,$U^{(0)}$)
  for i = 0, 1, ... **do**
    $W^{(k+1)} \leftarrow AU^{(k)}$
    $U^{(k+1)}R^{(k+1)} \leftarrow W^{(k+1)}$ ▷ QR factorization
    $Y^{(k+1)} \leftarrow (U^{(k+1)})^* AU^{(k+1)}$
  end for
end procedure

We now have all the tools to describe the subspace iteration, starting from a generic $n \times k$ basis $U^{(0)}$. The corresponding pseudocode is described in Algorithm 2.6. The latter introduces the quantity $Y^{(k+1)} = (U^{(k+1)})^* AU^{(k+1)}$, which takes the role of the term $(v^{(k)})^* Av^{(k)}$ that we had in the power iteration. Observe that if $U^{(k)}$ is a basis for an invariant subspace, this yields

$$AU^{(k)} = U^{(k)}Y^{(k)} \implies \Lambda(Y^{(k)}) \subseteq \Lambda(A),$$

with $U^{(k)}w$ being the eigenvectors, if $Y^{(k)}w = \lambda w$. Hence, when A is large, we can use the eigenvalues of the (small) matrix $Y^{(k)}$ as approximation for its (largest) eigenvalues. Approximation to the eigenvectors are obtained as well.

A convergence theorem for the subspace iteration would require the angle between subspaces, which is nicely described by the singular value decomposition (SVD), a tool which we have not yet introduced. Hence, we will limit ourselves to understanding the convergence of the eigenvalues of $Y^{(k)}$ to the ones of A, which depends on λ_{p+1}/λ_p.

Theorem 2.6.1. *Let A be an $n \times n$ diagonalizable matrix, with $V^{-1}AV = D$, and $D = \operatorname{diag}(\lambda_1, \dots, \lambda_n)$. Let $U^{(0)} \in \mathbb{C}^{n \times p}$ be a matrix with orthogonal columns. If the eigenvalues, ordered by magnitude, satisfy*

$$|\lambda_1| \geq \dots \geq |\lambda_p| > |\lambda_{p+1}| \geq \dots \geq |\lambda_n|,$$

and $V^{-1}U^{(0)}$ has an invertible minor in the top p rows, then the subspace iteration defined in Algorithm 2.6 produces a sequence of matrices $Y^{(k)}$ whose spectrum converges to $\{\lambda_1, \dots, \lambda_p\}$ with rate λ_{p+1}/λ_p.

Proof. The definition of subspace iteration implies that the iterate $U^{(k)}$ is an orthogonal basis of $A^k U^{(0)}$. If the latter is a full-rank matrix, this completely determines the column span of $U^{(k)}$.

We may write $A^k U^{(0)}$ leveraging the eigendecomposition of A, using the hypothesis on the invertibility of the top $p \times p$ submatrix of $V^{-1}U^{(0)}$:

$$A^k U^{(0)} = VD^k V^{-1}U^{(0)} =: VD^k \begin{bmatrix} X_0 \\ X_1 \end{bmatrix}, \qquad \det X_0 \neq 0.$$

Notice that this implies that $A^k U^{(0)}$ is full rank for all k. Partitioning D as $D_1 \oplus D_2$, with D_1 containing the eigenvalues $\lambda_1, \dots, \lambda_p$, we obtain that

$$\operatorname{colspan}(U^{(k)}) = \operatorname{colspan}\left(V \begin{bmatrix} D_1^k X_0 \\ D_2^k X_1 \end{bmatrix}\right) = \operatorname{colspan}\left(V \begin{bmatrix} I_p \\ D_2^k X_1 X_0^{-1} D_1^{-k} \end{bmatrix}\right),$$

where we have used that $\operatorname{colspan}(AB) = \operatorname{colspan}(A)$ for any invertible matrix B, and tall and thin matrix A. Since $\|D_2^k X_1 X_0^{-1} D_1^{-k}\|$ converges to zero with rate λ_{p+1}/λ_p, the intuition suggests that

$$\operatorname{colspan}(U^{(k)}) \to \operatorname{colspan}\left(V \begin{bmatrix} I_p \\ 0 \end{bmatrix}\right),$$

which are the eigenvectors relative to $\lambda_1, \dots, \lambda_p$. Formalizing this claim would require angles between subspaces; we now prove the claim about the eigenvalues of $Y^{(k)}$. Let v_j be the eigenvector for λ_j in A. Then, by defining $w_j^{(k)} := (U^{(k)})^* v_j$ we have

$$\begin{aligned} Y^{(k)} w_j^{(k)} &= (U^{(k)})^* A U^{(k)} (U^{(k)})^* v_j = (U^{(k)})^* \lambda_j v_j - (U^{(k)})^* A (I - U^{(k)} (U^{(k)})^*) v_j \\ &= \lambda_j w_j^{(k)} - (U^{(k)})^* A (I - U^{(k)} (U^{(k)})^*) v_j. \end{aligned}$$

By taking spectral norms, we can bound the residual for the eigenpair $\lambda_j, w_j^{(k)}$ as follows:

$$\|Y^{(k)} w_j^{(k)} - \lambda_j w_j^{(k)}\|_2 \leq \|A\|_2 \|(I - U^{(k)} (U^{(k)})^*) v_j\|_2 = \|A\|_2 \min_{z \in \operatorname{colspan} U^{(k)}} \|v_j - z\|_2,$$

where in the last step we have used the characterization of the orthogonal projection as the minimization of the Euclidean norm of the difference. We can make an explicit choice for z, by setting

$$z = V \begin{bmatrix} I_p \\ D_2^k X_1 X_0^{-1} D_1^{-k} \end{bmatrix} e_j \implies z - v_j = V \begin{bmatrix} 0_p \\ D_2^k X_1 X_0^{-1} D_1^{-k} \end{bmatrix} e_j.$$

Taking norms, yields the upper bound

$$\|Y^{(k)} w_j^{(k)} - \lambda_j w_j^{(k)}\|_2 \leq \|A\|_2 \|V\|_2 \|X_1 X_0^{-1}\|_2 \|D_2^k\|_2 \|D_1^{-k}\|_2 \sim \mathcal{O}\left(\left|\frac{\lambda_{p+1}}{\lambda_p}\right|^k\right).$$

Hence, λ_j is an approximate eigenvalue of $Y^{(k)}$ with backward error bounded as above, thanks to Theorem 2.2.3. The conclusion follows by a continuity argument of the spectrum, combined with the fact that $Y^{(k)}$ is diagonalizable for large enough k, and therefore the dependence is at least C^1 (for large enough k). □

2.7 Simultaneous iteration

A key advantage of subspace iteration is that, while running the algorithm with subspaces of dimension p, we are actually simultaneously running all iterations for $p' = 1, \dots, p$.

Note that, if W is tall and thin, its economy-size QR factorization has emebedded all the economy-size QR factorizations for W' that include the first p' columns of W:

$$W = QR \implies W \begin{bmatrix} I_{p'} \\ 0 \end{bmatrix} = QR \begin{bmatrix} I_{p'} \\ 0 \end{bmatrix} = \left(Q \begin{bmatrix} I_{p'} \\ 0 \end{bmatrix} \right) \left(\begin{bmatrix} I_{p'} & 0 \end{bmatrix} R \begin{bmatrix} I_{p'} \\ 0 \end{bmatrix} \right).$$

Hence, if we restrict the matrices $U^{(k)}$ and $Y^{(k)}$ generated by the subspace iteration by considering only the first p' columns of $U^{(k)}$ and the top $p' \times p'$ minor of $Y^{(k)}$, we get the subspace iteration of dimension p' started from the first p' columns of $U^{(0)}$.

An immediate consequence of this observation is the following result.

Theorem 2.7.1. *Let A be a diagonalizable matrix with eigenvalues ordered as $|\lambda_1| > \dots > |\lambda_n|$, and consider the matrices $U^{(k)}$ generated by the subspace iteration started from $U^{(0)} = I_n$. Then, if the leading $p \times p$ minors of the inverse eigenvector matrix V^{-1} are all invertible, the sequence $Y^{(k)}$ converges, up to scaling by diagonal unitary matrices, to a Schur form of A.*

Proof. It suffices to combine all observations that we have made up to now. The hypothesis on V^{-1} guarantees the convergence of all the simultaneous subspace iterations for $p = 1, \dots, n$. As a consequence, the unitary matrices $U^{(k)}$ converge to an orthogonal basis spanned by the eigenvectors relative to $\lambda_1, \dots, \lambda_n$, which in turn implies the convergence of $Y^{(k)}$ to a Schur form.

The eigenvector basis is uniquely determined up to a scaling of the columns by a complex number of modulus 1, from which the thesis follows. □

Exercise 2.7.2 (Real Schur form). Show that the assumptions of Theorem 2.7.1 fail for real matrices with complex eigenvalues, but nevertheless the proof can be modified to guarantee convergence to the real Schur form, with 2×2 blocks on the diagonal.

2.8 The QR iteration

We now rephrase the simultaneous subspace iteration in a way that will be much more amenable to efficient computation. On one hand, the simultaneous subspace iteration delivers an approximation of the Schur form, as originally desired. On the other hand, it does so at a large cost: the convergence speed is slow (governed by the minimal ratio between two consecutive eigenvalues), and the cost per iteration is cubic.

Recall that our aim is to design a matrix iteration that produces a sequence of matrices that are similar, through unitary or orthogonal matrices. In fact, the simultaneous iteration started with $U^{(0)} = I_n$ constructs such sequence:

$$Y^{(k+1)} = (U^{(k+1)})^* A U^{(k+1)} = (U^{(k+1)})^* U^{(k)} \underbrace{(U^{(k)})^* A U^{(k)}}_{Y^{(k)}} (U^{(k)})^* U^{(k+1)} = (Z^{(k)})^* Y^{(k)} Z^{(k)},$$

where we have set $Z^{(k)} := (U^{(k)})^* U^{(k+1)}$. Moreover, looking at line 4 in Algorithm 2.6, we see that

$$Z^{(k)} R^{(k+1)} = Y^{(k)},$$

meaning that $Z^{(k)}$ is the Q factor of a QR factorization of the matrix $Y^{(k)}$. Finally, observe that to get the conjugate of a square matrix with respect to its Q factor it is sufficient to compute product RQ of its QR factorization; in our context this reads as

$$Y^{(k+1)} = (Z^{(k)})^* Y^{(k)} Z^{(k)} = R^{(k+1)} Z^{(k)}.$$

Therefore, if we find a way to construct the matrices $Z^{(k)}$ directly, we can rephrase the iteration in a more convenient way. To achieve this, we need to first recall a few relevant facts regarding the QR factorization of a matrix A.

Theorem 2.8.1. *Let $A \in \mathbb{C}^{m \times n}$ be a full rank matrix with $m \geq n$, and $Q_1 R_1 = Q_2 R_2 = A$ two economy-size QR factorizations. Then, there exists a unitary diagonal matrix D such that $Q_1 = Q_2 D$.*

Proof. Since R_1 and R_2 need to be invertible $n \times n$ matrices, We may rearrange the two factorization by writing:

$$D := Q_2^* Q_1 = R_2 R_1^{-1}$$

From the above equation we conclude that D is upper triangular. Moreover, by using that the column span of Q_1 is included in the one of Q_2 (and viceversa), we also have that D is a unitary (or orthogonal) square matrix. A unitary upper triangular matrix has to be diagonal with diagonal entries of modulus 1. To conclude, we use again that the column span of Q_1 is included in the one of Q_2 to get:

$$Q_1 = Q_2 Q_2^* Q_1 = Q_2 D.$$

□

Diagonal unitary matrices are often called *phase matrices*, since the diagonal contain elements of modulus 1, which can be interpreted as angles. The previous result shows that the QR factorization is *essentially unique*.

Exercise 2.8.2. Show that the hypothesis of being full rank is (almost) necessary, and if $\operatorname{rank}(A) \leq n - 2$ the essential uniqueness is lost.

The observations that we used to define $Z^{(k)}$ allow to construct the QR iteration, described in Algorithm 4. The pseudocode can be implemented quickly, since the kernel of each iteration is just one QR factorization and one matrix-matrix multiplication. However, Algorithm 4 in its current form is far from practical for the following reasons:

- The convergence depends on the eigenvalues being of different moduli, and can be very slow for clustered eigenvalues.
- Each iteration has a cubic cost (both the QR factorizations and the matrix-matrix multiplication contribute to this), and even in the optimistic scenario where $\mathcal{O}(n)$ iterations are enough, this would still yields a $\mathcal{O}(n^4)$ algorithm.

Algorithm 4 Explicit QR iteration

```
procedure QR(A)
    Y^(0) ← A
    for i = 0, 1, ... do
        Z^(k), R^(k) ← QR(Y^(k))
        Y^(k+1) ← R^(k) Z^(k)
    end for
end procedure
```

- Several hypotheses that we have made are often not satisfied. For instance, all real matrices with complex conjugate eigenvalues have eigenvalues $|\lambda_p| = |\lambda_{p+1}|$.

The next section will be dedicated to tweak the algorithm to make it a practical one, which is indeed the state-of-the art algorithm for computing eigenvalues of general unstructured matrices.

2.9 Shifting and deflation

Let us consider the following model problem: we have a matrix A with eigenvalues satisfying the following inequalities:

$$|\lambda_1| \geq \ldots \geq |\lambda_{n-1}| > |\lambda_n|, \qquad \frac{|\lambda_n|}{|\lambda_{n-1}|} = \epsilon \ll 1.$$

In view of the previous analysis, we expect that after k iterations of the QR method we get a matrix $Y^{(k)}$ of the form

$$Y^{(k)} = \left[\begin{array}{c|c} \hat{Y}^{(k)} & \begin{matrix}\times\\ \vdots\\ \times\end{matrix} \\ \hline (w^{(k)})^T & \lambda_n^{(k)} \end{array}\right], \qquad \begin{cases} |\lambda_n^{(k)} - \lambda_n| \sim \mathcal{O}(\epsilon^k) \\ \|w^{(k)}\| \sim \mathcal{O}(\epsilon^k) \end{cases}.$$

If ϵ is sufficiently small, after a few iterations we will have that $\|w^{(k)}\|$ will be of the size of the machine precision, and therefore we may consider the slightly perturbed matrix

$$Y^{(k)} + \delta Y^{(k)} = \left[\begin{array}{c|c} \hat{Y}^{(k)} & \begin{matrix}\times\\ \vdots\\ \times\end{matrix} \\ \hline 0^T & \lambda_n^{(k)} \end{array}\right],$$

which has exactly $\lambda_n^{(k)}$ as eigenvalue. Since this matrix is unitarily similar to A, this correspond to the exact QR iteration with the matrix $A + \delta A$ with $\delta A = U^{(k)} \delta Y^{(k)} (U^{(k)})^*$, which has spectral norm equal to $\|w^{(k)}\|$. Hence, we can decide that $\lambda_n^{(k)}$ is an approximate eigenvalue of A with a small backward error, and continue the iteration on the smaller $(n-1) \times (n-1)$ matrix $\hat{Y}^{(k)}$. This procedures is called *deflation*.

There is, in general, no reason to assume that λ_n is much smaller than the rest of spectrum, and therefore to be in this favorable situation. It turns out that we can always slightly modify the eigenvalue problem to make it happen.

Assume to be given a certain shift $\sigma \in \mathbb{C}$ such that $\sigma \approx \lambda_n$; then, the shifted matrix $A - \sigma I$ has $\lambda_n - \sigma$ as eigenvalue of smaller module (if we assume that σ is closer to λ_n than to any other eigenvalue). Applying one step of QR iteration to the shifted matrix will yield

$$\begin{aligned} Y_\sigma^{(0)} &= A - \sigma I & Z_\sigma^{(0)} R_\sigma^{(0)} &= Y_\sigma^{(0)} \\ Y_\sigma^{(1)} &= (Z_\sigma^{(0)})^* Y_\sigma^{(0)} Z_\sigma^{(0)} = (Z_\sigma^{(0)})^* A Z_\sigma^{(0)} - \sigma I, \end{aligned}$$

where we have denoted by $Y_\sigma^{(k)}$ the iteration obtained by starting from $A - \sigma I$. This observation may be generalized to an arbitrary number of steps through the following result.

Lemma 2.9.1. *Let $Y_\sigma^{(k)}$ the matrix sequence generated by the QR iteration started with $A - \sigma I$. Then, if we denote by $Z_\sigma^{(k)}$ the orthogonal matrix of the QR factorization at step k,*

$$(Z_\sigma^{(0)} \dots Z_\sigma^{(k-1)})^* A (Z_\sigma^{(0)} \dots Z_\sigma^{(k-1)}) = Y_\sigma^{(k)} + \sigma I, \qquad \forall k \geq 0.$$

Proof. The definition of the QR iteration yields

$$(Z_\sigma^{(0)} \dots Z_\sigma^{(k-1)})^* (A - \sigma I)(Z_\sigma^{(0)} \dots Z_\sigma^{(k-1)}) = Y_\sigma^{(k)}.$$

The claim follows by moving σI to the right hand side, and recalling that the matrices $Z^{(i)}$ are unitary. □

We conclude that, if we have a good approximation $\sigma \approx \lambda_n$ at our disposal, we can make the (shifted) QR iteration converge in a few steps to a form where λ_n can be "deflated". Then, we proceed by restricting our focus on the top-left $(n-1) \times (n-1)$ minor, and continue the process. The two more common strategies for choosing the shift σ are the following (note that, in practice, the shift σ is renewed at each iteration):

Rayleigh shift works by selecting $\sigma = Y_{nn}^{(k)}$; this choice may block convergence, for instance for real matrices wiht complex eigenvalues, where the iteration cannot "escape" real matrices, or for particular symmetric configurations.

Wilkinson shift selects as shift one of the eigenvalues of the bottom-right 2×2 matrix, taking the one closest to $Y_{nn}^{(k)}$; this choice guarantees convergence for symmetric matrices, and works very well in practice for unsymmetric matrices as well.

Shifting has the purpose of guaranteeing convergence of QR in about $\mathcal{O}(n)$ iterations, most often about 2 per eigenvalue. This implies that the cost of the QR method is $\mathcal{O}(n \cdot f(n))$, where $f(n)$ is the cost of a single iteration. In our setting, $f(n) = n^3$ because of the QR factorization and the matrix-matrix multiplication, and this leads to a total cost of $\mathcal{O}(n^4)$ flops.

For dense linear algebra methods, a cubic cost is the usual target, and for eigenvalue problems can be achieved by using the Hessenberg reduction, described in the next section.

2.10 Hessenberg reduction

A key observation for reducing the cost of the iteration is preprocessing the matrix to make it "as upper triangular as possible". Clearly, the preprocessing step needs to work with unitary matrices, and to be a similarity.

Definition 2.10.1. A matrix H is in *Hessenberg form* if it has zero elements below the first subdiagonal, i.e., if $H_{ij} = 0$ for all $i > j + 1$.

The reduction of the matrix to Hessenberg form can be computed with $\mathcal{O}(n^3)$ flops using Householder reflectors.

Lemma 2.10.2. *Let A be any $n \times n$ complex matrix, with $n \geq 2$. Then, there exist an upper Hessenberg matrix H and $n-2$ Householder reflectors P_j for $j = 1, \ldots, n-2$, such that*

$$P_{n-2} \ldots P_1 A P_1^* \ldots P_{n-2}^* = H,$$

The matrices H and $P := P_{n-2} \ldots P_1$ can be computed from A with $\mathcal{O}(n^3)$ flops.

Proof. The proof presents an algorithm for computing H and P_j with the required asymptotic complexity. A more formal proof can be obtained by using induction. Let $\hat{P}_1$ be a $(n-1) \times (n-1)$ Householder reflector such that

$$\hat{P}_1 A_{2:n,1} = \begin{bmatrix} \times \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

where $\times$ is used to denote a generic non-zero element. Then, if we define $P_1 := 1 \oplus \hat{P}_1$ the matrix $P_1 A P_1^*$ has the following sparsity pattern (by a direct computation):

$$A^{(1)} := P_1 A P_1^* = \begin{bmatrix} \times & \times & \times & \ldots & \times \\ \times & \times & \times & \ldots & \times \\ 0 & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & \times & \times & \ldots & \times \end{bmatrix},$$

and can be compute in $\mathcal{O}(n^2)$ flops exploiting the identity-plus-rank-1 structure of the Householder reflector. Following the same idea, P_2 can be defined to have

$$P_2 = I_2 \oplus \hat{P}_2, \qquad \hat{P}_2 A^{(1)}_{3:n,2} = \begin{bmatrix} \times \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Computing $P_2 A^{(1)} P_2^*$ will put the second column in "Hessenberg form", and will not deteriorate the structure of the first column, thanks to the presence of I_2 on top. Continuing the process yields the required upper Hessenberg matrix $H = A^{(n-2)}$. □

Preprocessing the matrix A to be in upper Hessenberg form brings two key advantages to the QR iteration:

- For an upper Hessenberg matrix, the QR factorization $Z^{(k)} R^{(k)} = Y^{(k)}$ and the next iterate $Y^{(k+1)} := R^{(k)} Z^{(k)}$ can be computed with $\mathcal{O}(n^2)$ flops.
- The Hessenberg structure is preserved by the QR iterations, and therefore the above benefit is not limited to the first step.

We now introduce *Givens' rotations*, which are unitary matrices with a similar aim of Householder reflectors, but that act element-by-element, making it easier to preserve sparsity.

Definition 2.10.3. A Givens rotation acting on rows (k, l) is a matrix of the form G such that, for some $c, s \in \mathbb{C}$ with $|c|^2 + |s|^2 = 1$,

$$G = \begin{bmatrix} I_{k_1} & & & & \\ & c & & s & \\ & & I_{k_2} & & \\ & -\overline{s} & & \overline{c} & \\ & & & & I_{k_3} \end{bmatrix},$$

and such that the entries $c, s, -\overline{s}, \overline{c}$ are found on rows and columns k or l. These transformations are unitary with $\det(G) = 1$.

The property $|c|^2 + |s|^2$ allows to interpret c and s as (complex) cosines and sines, and this is the reason for naming these transformation "rotations". Often, we will consider $l = k + 1$, which allows to look for the simplified form

$$G = I_{k_1} \oplus \hat{G} \oplus I_{k_2}, \qquad \hat{G} := \begin{bmatrix} c & s \\ -\overline{s} & \overline{c} \end{bmatrix}.$$

Lemma 2.10.4. *Given any vector $v \in \mathbb{C}^n$ and $i > 1$, it exists a Givens' rotation G such that $(Gv)_i = 0$, and G only changes the rows $i - 1$ and i of v. In particular, G can be applied with $\mathcal{O}(1)$ flops.*

Proof. We consider the problem of finding the 2×2 rotation $\hat{G}$ such that

$$\hat{G} \begin{bmatrix} v_{i-1} \\ v_i \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix},$$

where $\times$ is a placeholder for any complex number. This is equivalent to imposing $\overline{c} v_i - \overline{s} v_{i-1} = 0$, which is a linear equation in $\overline{c}, \overline{s}$; choosing any norm 1 solution yields the sought result by setting $G = I_{i-2} \oplus \hat{G} \oplus I_{n-i}$. □

We now use Givens' rotations for computing a QR factorization of an upper Hessenberg matrix in quadratic time.

Lemma 2.10.5. *Let H be an $n \times n$ upper Hessenberg matrix. Then, there exists $n-1$ Givens' rotations $G_1, \dots, G_{n-1}$ with G_i acting on rows i and $i+1$, such that*

$$H = G_1^* \dots G_{n-1}^* R = QR,$$

with R upper triangular. The matrices Q and R can be computed with $\mathcal{O}(n^2)$ flops.

Proof. We prove the result by induction, showing that the construction requires at most $8n^2$ floating point operations. The result is trivially true for $n = 1$, since H is already upper triangular, and we can just set $Q = 1$ as the empty product of 0 rotations.

Assume that the result holds true for $n-1$, and consider a rotation G_1 operating on rows 1 and 2 such that:

$$G_1 \begin{bmatrix} H_{11} \\ H_{21} \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} \times \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

We then have

$$G_1 H = \left[\begin{array}{c|ccc} \times & \times & \dots & \times \\ \hline & & & \\ & & \hat{H} & \\ & & & \end{array}\right],$$

with $\hat{H}$ an $(n-1) \times (n-1)$ upper Hessenberg matrix. By induction, we have $\hat{H} = \hat{G}_1^* \dots \hat{G}_{n-2}^*$, and by setting $G_i := 1 \oplus \hat{G}_{i-1}$ for $i = 2, \dots, n-1$, we get

$$H = G_1^* G_2^* \dots G_{n-1}^* \underbrace{\left[\begin{array}{c|ccc} \times & \times & \dots & \times \\ \hline & & & \\ & & \hat{R} & \\ & & & \end{array}\right]}_{=:R} = G_1^* G_2^* \dots G_{n-1}^* R = QR.$$

A direct computation shows that multiplying a rotation by a matrix requires $4n$ floating point operations[2], obtaining R and Q from $\hat{R}$ and $\hat{Q} := \hat{G}_1^* \dots \hat{G}_{n-2}^*$ requires 2 products by G_1. In addition, we have 4 more floating point operations for finding G_1 and computing $G_1 H e_1$, which yields the total cost

$$8n + 4 + 8(n-1)^2 = 8n^2 - 8n + 12 \sim \mathcal{O}(n^2). \qquad \square$$

We now focus on the structure preservation statement. We briefly recall the definitions of lower and upper banded matrices.

[2]Here we are considering for simplicity a CPU capable of a *fused-multiply-add*, that can compute $ab+c$ in a single floating point operation; any modern CPU supports this instruction.

Definition 2.10.6. A matrix has *lower (resp. upper) bandwidth* k if all its entries below the k-th subdiagonal (resp. superdiagonal) are zero.

According to this definition, a matrix with bandwidth k has bandwidth k' for any $k' > k$; sometimes, it is helpful to also require at least one nonzero diagonal entry to make sure that the bandwidth can be precisely defined. For our applications, this will not be of particular importance.

Lemma 2.10.7. *Let A, B be matrices with lower (resp. upper) bandwidth k_A, k_B, respectively. Then, their product AB has lower (resp. upper) bandwidth $k_A + k_B$.*

Proof. Note that having bandwidth k_A can be rephrased as the following inclusion:

$$Ae_j \in \operatorname{colspan}\{e_1, \dots, e_{\min\{j+k_A,n\}}\},$$

and similar for B. Hence, the claim is equivalent to verify that

$$ABe_j = Av = A\left[\sum_{i \le \min\{j+k_B,n\}} \gamma_i e_i\right] = \sum_{i \le \min\{j+k_B,n\}} \gamma_i Ae_i \in \mathcal{S}_j,$$

where

$$\mathcal{S}_j := \bigcup_{i=1}^{\min\{j+k_B,n\}} \operatorname{colspan}\{e_1, \dots, e_{\min\{i+k_A,n\}}\} = \operatorname{colspan}\{e_1, \dots, e_{\min\{j+k_A+k_B,n\}}\}. \quad \square$$

We can now make use of Lemma 2.10.7 to ensure that, under the full-rank assumption of the iterates in the QR method, which ensure essential uniqueness of the QR factorizations, we have structure preservation for Hessenberg matrices.

Lemma 2.10.8. *Let H be a full rank upper Hessenberg matrix, and $H = QR$ a QR factorization. Then, the matrix Q is upper Hessenberg.*

Proof. The full-rank assumption guarantees the invertibility of R, and therefore we have $Q = HR^{-1}$. The result follows from Lemma 2.10.7. $\square$

Lemma 2.10.9. *Let $Y^{(k+1)} = R^{(k)}Z^{(k)} + \sigma_k I$ be the $(k+1)$th iterate of the (shifted) QR method started with an upper Hessenberg matrix $Y^{(0)}$ of full rank. Then, $Y^{(k+1)}$ is upper Hessenberg as well.*

Proof. Lemma 2.10.9 guarantees that $Z^{(k)}$ is upper Hessenberg, and therefore we have that $Y^{(k+1)} = R^{(k)}Z^{(k)} + \sigma_k I$ is upper Hessenberg as well by means of Lemma 2.10.7, since the shift $\sigma_k I$ does not introduce any perturbation in the Hessenberg structure. $\square$

2.11 Computing eigenvectors and invariant subspaces

The QR iteration as discussed in the previous sections allows to build a sequence of similar matrices $Y^{(k)}$ that, under suitable assumptions, converge to a Schur form of $Y^{(0)} = A$. The final Schur form T is enough for determining the eigenvalues (we just need to read the diagonal entries) and in case of multiple eigenvalues the corresponding Jordan blocks as well.

The eigenvector recovery is more involved, and is performed in two steps:

- First, we determine the eigenvectors w of the upper triangular matrix T, corresponding to the eigenvalues $\lambda_i := T_{ii}$ for $i = 1, \dots, n$.
- Then, we recover the eigenvectors of the original problem by using the relation $Q^*AQ = T$ and setting $v = Qw$.

If $Tw = \lambda w$ then $AQ = QT$ implies $Av = AQw = QTw = \lambda Qw = \lambda v$, and therefore the second step completely characterizes the eigenvectors of A starting from the ones of T. The analogue construction works to map Jordan chains into Jordan chains as well.

To compute the eigenvectors of the upper triangular matrix, we make the simplifying assumption that T_{ii} is simple, and rely on the following observation:

$$T - T_{ii}I = \begin{bmatrix} T_1 & x & \\ & 0 & \\ & & T_2 \end{bmatrix},$$

with T_1 nonsingular and upper triangular. We need to determine a vector in the right kernel of the above matrix, which can be done by imposing:

$$(T - T_{ii}I)w = 0, \qquad w = \begin{bmatrix} y \\ 1 \\ 0 \end{bmatrix},$$

where w is partitioned to match the block structure identified in T. Then, we solve the equation by setting $T_1 y = -x$. Hence, y (and therefore w) is determined solving an upper triangular linear system, which costs at most $\mathcal{O}(n^2)$ flops. This needs to be repeated for all eigenvalues, yielding a total cost of $\mathcal{O}(n^3)$.

Exercise 2.11.1. Work out what happens in the case of multiple but semisimple eigenvalues, and in the case of Jordan chains. Note that, although theoretically possible to extract Jordan chains from the Schur form, matrices with a nontrivial Jordan structure form a zero-measure set within all square matrices, and need therefore to be treated carefully when working in floating point arithmetic.

A similar technique can be used to find orthogonal basis for invariant subspaces corresponding to a subset $\{\lambda_1, \dots, \lambda_k\} \subseteq \Lambda(A)$ of all the eigenvalues of A. Assume that we are particularly lucky, and that the Schur form computed by the QR iteration satisfies

$$Q^* A Q = \begin{bmatrix} T_{11} & T_{12} \\ & T_{22} \end{bmatrix}, \qquad T_{11} = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_k \end{bmatrix},$$

with T_{22} containing all the other eigenvalues. Then, the invariant subspace under consideration is spanned by the first k columns of Q, which form an orthogonal basis for it. Our problem is easily solved.

However, there is no particular reason why this should happen: the QR iteration can compute the eigenvalue in any order, and we have little control over the process. If the eigenvalues end up in the "wrong" place, we can simply reorder them, to push the ones of interest at the top of the matrix.

The problem can be reduced to the 2×2 case which is solved by the following Lemma.

Lemma 2.11.2. *Let T be an upper triangular matrix with two distinct eigenvalues $t_{11} = \lambda_1 \neq \lambda_2 = t_{22}$; let G be a Givens' rotation such that, for some $\alpha \in \mathbb{C}$,*

$$G \begin{bmatrix} t_{12} \\ \lambda_2 - \lambda_1 \end{bmatrix} = \begin{bmatrix} \alpha \\ 0 \end{bmatrix},$$

Then, the matrix GTG^* is upper triangular with eigenvalues listed in the opposite order.

Proof. Note that, by construction, we have

$$G(T - \lambda_1 I)G^* = G \begin{bmatrix} 0 & t_{12} \\ 0 & \lambda_2 - \lambda_1 \end{bmatrix} G^* = \begin{bmatrix} 0 & \alpha \\ 0 & 0 \end{bmatrix} G^* = \begin{bmatrix} \times & \times \\ 0 & 0 \end{bmatrix},$$

where as usual we have used $\times$ to denote the sparsity pattern in the matrix. Applying the same transformation to T yields

$$GTG^* = \begin{bmatrix} \times & \times \\ 0 & 0 \end{bmatrix} + \lambda_1 I = \begin{bmatrix} \lambda_2 & \times \\ 0 & \lambda_1 \end{bmatrix},$$

where the entry in position $(1, 1)$ is determined to be exactly λ_2 because the eigenvalues of T cannot change under similarity. □

Lemma 2.11.2 can be employed to swap two eigenvalues λ_i, λ_{i+1} of a larger $n \times n$ upper triangular matrix by considering a rotation on two consecutive rows. Using the fact that transpositions generate all the permutations, we conclude that any permutation of the eigenvalues is possible, and is easily obtained by repeated applications of Lemma 2.11.2.

Exercise 2.11.3. Show that any invariant subspace (i.e., corresponding to an arbitrary choice of eigenvalues) can be computed with $\mathcal{O}(n^3)$ flops using the QR iteration and eigenvalue reordering[3].

[3]For this task, assume convergence of the QR iteration in at most $\mathcal{O}(n)$ steps.

2.12 Double shifting and the real Schur form

If the matrix A is real, computing the Schur form with complex shifts may be undesirable, because of the additional cost of complex arithmetic. Clearly, there is no hope of finding the Schur form with real arithmetic if the matrix has complex eigenvalues.

We can, however, restrict our attention to the real Schur form:

Definition 2.12.1. A matrix T is in *real Schur form* if it is block upper triangular with diagonal block T_{ii} such that either T_{ii} is a 1×1 real matrix, or a 2×2 matrix of the form

$$T_{ii} = \begin{bmatrix} a & b \\ -b & a \end{bmatrix},$$

which has $a \pm ib$ as eigenvalues.

Even though a matrix in real Schur form is not (strictly speaking) upper triangular, its eigenvalues are immediately readable from the diagonal blocks without any computation. In addition, the arguments used to find the eigenvectors and invariant subspaces from the Schur form can be easily adapted.

The real structure can be maintained throughout the iterations by the following trick; if a shift σ is determined (for instance by the Wilkinson shifting strategy), we proceed as follows:

- If $\sigma \in \mathbb{R}$, we proceed with the standard QR iteration.
- If $\sigma \in \mathbb{C} \setminus \mathbb{R}$, we consider the polynomial with real coefficients

  $$p(z) = (z - \sigma)(z - \overline{\sigma}) = z^2 - 2\Re(\sigma)z + |\sigma|^2,$$

  and compute $p(Y^{(k)})e_1$.

In the second case, we consider the two rotations needed to transform $p(Y^{(k)})e_1$ into a multiple of e_1, and apply these rotations to $Y^{(k)}$. We then restore the upper Hessenberg structure by chasing the 2×2 bulge analogously to the single shift case.

An iteration of this form costs about twice as much as a single shift iteration. However, the convergence can be linked to the subspace iteration applied to $p(Y^{(k)})$, and therefore we can expect the eigenvalues close to σ and $\overline{\sigma}$ to be well-approximated together.

In particular, we will have soon that the last two rows of $Y^{(k)}$, and we will be able to deflate a 2×2 block at once. This block will have complex conjugate eigenvalues, and can be put in the standard real Schur form.

Remark 2.12.2. We can take the idea even further; if we have a good estimate for a group of eigenvalues $\sigma_1, \ldots, \sigma_m$, we may choose the shift polynomial as

$$p(z) = \prod_{j=1}^{m} (z - \sigma_j).$$

According to the previous analysis, we may establish a connection with the subspace iteration shifted with all those shifts at once. Although this idea sounds attractive on

paper, as m grows large a phenomenon known as shift blurring [7] comes into play, and the effect of the shifts "fades". However, intermediate values of m allow for better cache usage on modern processors, and is known as the multishift QR algorithm.

3 Symmetric eigenvalue problems and the SVD

Symmetric eigenvalue problems are inherently easier than nonsymmetric ones, and allow to prove much stronger results and characterizations. In this section, we discuss the tridiagonal QR iteration, bounds in the eigenvalues through variational characterizations, the divide-and-conquer scheme.

We will then see that there is a close relation between the symmetric eigenproblem and the singular value decomposition (SVD), a powerful theoretical and algorithmic factorization.

3.1 Tridiagonal QR iteration

If we apply the QR iteration to a symmetric a few observations can be made, which are summarized by the next Lemma.

Lemma 3.1.1. *Let $A = A^*$ a symmetric or Hermitian matrix, and $Y^{(k)}$ the QR iterates applied after the Hessenberg reduction $Y^{(0)} = Q^*AQ$. Then, all the matrices $Y^{(k)}$ are tridiagonal.*

Proof. Note that $Y^{(k)}$ are unitarily similar to A, therefore there exists Q_k orthogonal (or unitary) such that $Q_k^*AQ_k = Y^{(k)}$. Hence, $Y^{(k)} = (Y^{(k)})^*$ are all symmetric (or Hermitian).

All the $Y^{(k)}$ are Hessenberg in view of Lemma 2.10.9, and Hessenberg matrices have only one subdiagonal different from zero. The symmetry implies that all $Y^{(k)}$ have only one non-zero superdiagonal, and are therefore tridiagonal. □

Reducing a symmetric matrix A to tridiagonal form is not cheaper than reducing a general matrix to upper Hessenberg form, we still need to apply the rotations on the full matrix, for a total cost of $\mathcal{O}(n^3)$ flops. If A is tridiagonal, however, this structure is easily exploited in the QR iteration if only the eigenvalues are desired. Indeed, computing $Y^{(k+1)}$ from $Y^{(k)}$ requires the following steps:

- Find an appropriate shift σ (cost: $\mathcal{O}(1)$ flops).
- Determine a rotation such that $Y^{(k)}e_1 - \sigma e_1$ is a multiple of e_1 (cost: $\mathcal{O}(1)$ flops).
- Apply the rotation to the matrix, and chase it to the bottom (cost: applying $\mathcal{O}(n)$ rotations).

The last item is the expensive part, and in the unstructured case each rotation costs $\mathcal{O}(n)$ flops. In the tridiagonal case, the tridiagonal-plus-bulge structure is preserved throughout the chasing process, and therefore a rotation can be applied at $\mathcal{O}(1)$ cost.

Summarizing, we can run the tridiagonal QR iteration with $\mathcal{O}(n)$ flops per iteration, for a total cost of $\mathcal{O}(n^2)$ flops.

Remark 3.1.2. Computing the eigenvector in the tridiagonal case is however much more expensive: the rotations need to be applied to the Q matrices that represent the change of basis, and this requires $\mathcal{O}(n)$ flops per iteration. The total cost of the method is again $\mathcal{O}(n^3)$ flops.

3.2 Courant-Fischer's theorem and interlacing properties

As seen when analyzing the power method, in the Hermitian case, there is an intimate relation between eigenvalues, eigenvectors, and the Rayleigh quotient. Here, we provide a powerful theoretical tool, known as *Courant-Fischer min-max theorem*, that characterizes the eigenvalues as optimal values of the Rayleigh quotient over subspaces.

Theorem 3.2.1. *Let $A \in \mathbb{C}^{n\times n}$ be a Hermitian matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Then:*

$$\max_{\substack{U \subset \mathbb{C}^n \\ dim(U) = k}} \min_{\substack{x \in U \\ x \neq 0}} \frac{x^*Ax}{x^*x} = \lambda_k,$$

$$\min_{\substack{U \subset \mathbb{C}^n \\ dim(U) = k}} \max_{\substack{x \in U \\ x \neq 0}} \frac{x^*Ax}{x^*x} = \lambda_{n-k+1},$$

for all $k = 1, 2, \ldots, n$.

Proof. We prove only the max-min part as the min-max is completely analogous.

Let $v_1, \ldots, v_n$ be an orthonormal basis of $\mathbb{C}^n$ made of eigenvector of A and let $S := \text{colspan}(v_k, \ldots, v_n)$. Then, for every $U \subset \mathbb{C}^n$ of dimension k we have that $S \cap U \neq \{0\}$; more specifically, there exists $x \in S \cap U$, so that $x = \sum_{j=k}^n c_j v_j$ and $x \neq 0$. This implies

$$\frac{x^*Ax}{x^*x} = \frac{\sum_{j=k}^n |c_j|^2 \lambda_j}{\sum_{j=k}^n |c_j|^2} \leq \lambda_k.$$

This proves that, for each U, the minimum of the Rayleigh quotient is less or equal than λ_k which implies that the maximum over all possible U, of the minimum Rayleigh quotient is also bounded from above by λ_k. To get the claim, it is sufficient to show that for at least one choice of U, the value λ_k corresponds to the minimum of the Rayleigh quotient. This happens when considering $U = \text{colspan}\,(v_1, \ldots, v_k)$, and this concludes the proof. □

Courant-Fischer's result offers an interesting perspective that allows us to relate the eigenvalues of a Hermitian matrix with those of smaller matrices obtained by orthogonal projection.

Corollary 3.2.2. *Let $A \in \mathbb{C}^{n\times n}$ be a Hermitian matrix with eigenvalues $\alpha_1 \geq \cdots \geq \alpha_n$, $Q \in \mathbb{C}^{n\times(n-1)}$ such that $Q^*Q = I_{n-1}$, and $B = Q^*AQ \in \mathbb{C}^{(n-1)\times(n-1)}$ with eigenvalues $\beta_1 \geq \cdots \geq \beta_{n-1}$. Then,*

$$\alpha_1 \geq \beta_1 \geq \alpha_2 \geq \beta_2 \geq \cdots \geq \beta_{n-1} \geq \alpha_n,$$

and we say that the eigenvalues of A are interlaced with those of B.

Proof. In view of Theorem 3.2.1, we have

$$\beta_k = \max_{\substack{U \subset \mathbb{C}^{n-1} \\ \dim(U)=k}} \min_{\substack{x \in U \\ x \neq 0}} \frac{x^*Bx}{x^*x} = \min_{\substack{x \in \widetilde{U} \\ x \neq 0}} \frac{x^*Q^*AQx}{x^*Q^*Qx},$$

where $\widetilde{U}$ is a k-dimensional subspace of $\mathbb{C}^{n-1}$ where the maximum is attained. Let $\widehat{U} = Q\widetilde{U} = \{y \in \mathbb{C}^n : \ y = Qx, \text{for some } x \in \widetilde{U}\}$, then $\dim(\widehat{U}) = k$ and

$$\beta_k = \min_{\substack{x \in \widetilde{U} \\ x \neq 0}} \frac{x^*Q^*AQx}{x^*Q^*Qx} = \min_{\substack{y \in \widehat{U} \\ y \neq 0}} \frac{y^*Ay}{y^*y} \leq \max_{\substack{\widehat{U} \subset \mathbb{C}^n \\ \dim(\widehat{U})=k}} \min_{\substack{y \in \widehat{U} \\ y \neq 0}} \frac{y^*Ay}{y^*y} = \alpha_k.$$

The inequality $\beta_{k-1} \geq \alpha_k$ is obtained applying the same argument to the matrices $-A$ and $-B$. □

A direct consequence of Corollary 3.2.2, obtained by considering a subset of $n-1$ columns of the identity matrix for Q, is that the eigenvalues of a Hermitian matrix are interlaced by those of any $(n-1)\times(n-1)$ principal submatrix. Repeating the argument for smaller principal submatrices leads to the following result known in the literature as *Cauchy's interlacing theorem*.

Corollary 3.2.3. *Let $A \in \mathbb{C}^{n\times n}$ be a Hermitian matrix with eigenvalues $\alpha_1 \geq \cdots \geq \alpha_n$ and let $B \in \mathbb{C}^{m\times m}$ be a principal submatrix of A, for $m \leq n$, with eigenvalues $\beta_1 \geq \cdots \geq \beta_m$. Then*

$$\alpha_j \geq \beta_j \geq \alpha_{j+(n-m)},$$

for $j = 1, \ldots, m$.

Proof. The rigorous proof is left as an exercise to reader; hint: proceed by induction. □

Other interesting byproducts of Courant-Fischer's result concern the eigenvalues of the sum of Hermitian matrices. We report below a couple of results without proof (left as exercise), that will be useful in the next section.

Corollary 3.2.4. *Let A, B, C be Hermitian matrices with ordered eigenvalues $\alpha_j, \beta_j, \gamma_j$ and such that $A = B + C$. Then it holds:*

$$\beta_j + \gamma_{n-j+i} \leq \alpha_i \leq \beta_k + \gamma_{i-k+1},$$

for $1 \leq k \leq i \leq j \leq n$.

Corollary 3.2.5. *Let $A = B + uu^* \in \mathbb{C}^{n\times n}$ for B Hermitian and $u \in \mathbb{C}^n$ and let α_j, β_j be the ordered eigenvalues of A and B, respectively. Then:*

$$\beta_j \leq \alpha_j \leq \beta_{j+1},$$

for $j = 1, \ldots, n$ ($\beta_{n+1} = \infty$).

3.3 The divide-and-conquer method for Hermitian tridiagonal matrices

In the case of a Hermitian eigenvalue problem, the Hessenberg reduction returns a Hermitian tridiagonal matrix. In that case, there is an alternative approach, with respect to the QR method, to retrieve the eigenvalues and the eigenvectors. To simplify the exposition we consider a real symmetric triadiagonal matrix of the form

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

although, all we are going to see extends with no major difficulties to the complex Hermitian case. The key idea is that the tridiagonal matrix T can be block diagonalized by means of a rank 1 symmetric perturbation, i.e.:

$$\begin{aligned} T &= \begin{bmatrix} a_1 & b_1 & & & & & & \\ b_1 & \ddots & \ddots & & & & & \\ & \ddots & \ddots & b_{m-1} & & & & \\ & & b_{m-1} & a_m - b_m & & & & \\ & & & & a_{m+1} - b_m & b_{m+1} & & \\ & & & & b_{m+1} & \ddots & \ddots & \\ & & & & & \ddots & \ddots & b_{n-1} \\ & & & & & & b_{n-1} & a_n \end{bmatrix} + \begin{bmatrix} & & & & & & & \\ & & & & & & & \\ & & & b_m & b_m & & & \\ & & & b_m & b_m & & & \\ & & & & & & & \\ & & & & & & & \\ & & & & & & & \end{bmatrix} \\ &= \begin{bmatrix} T_1 & \\ & T_2 \end{bmatrix} + b_m v v^*, \qquad v = \begin{bmatrix} e_m \\ e_1 \end{bmatrix}. \end{aligned}$$

Let us assume that we have already computed (for instance by recursion) the eigendecompositions $T_i = Q_i \Lambda_i Q_i^*$, $i = 1, 2$. Then, the eigenvalues of T and those of T_1, T_2 are related by

$$T = \begin{bmatrix} Q_1\Lambda_1 Q_1^* & \\ & Q_2\Lambda_2 Q_2^* \end{bmatrix} + b_m v v^* = \begin{bmatrix} Q_1 & \\ & Q_2 \end{bmatrix} \left(\begin{bmatrix} \Lambda_1 & \\ & \Lambda_2 \end{bmatrix} + \rho u u^* \right) \begin{bmatrix} Q_1^* & \\ & Q_2^* \end{bmatrix},$$

where

$$u = \begin{bmatrix} Q_1^* & \\ & Q_2^* \end{bmatrix} v = \begin{bmatrix} Q_1^* e_m \\ Q_2^* e_1 \end{bmatrix} = \begin{bmatrix} \text{last column of } Q_1^* \\ \text{first column of } Q_2^* \end{bmatrix}.$$

Therefore, the eigenvalues of T coincides with those of $D+\rho uu^*$; we focus on the problem of "updating" the eigenvalues of a diagonal matrix when a rank 1 symmetric modification is applied. We start with some simplifications:

- $D = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_n \end{bmatrix}$ is such that $d_1 > d_2 > \cdots > d_n$,
- all the entries u_j of the vector u are non zero.

Later on, we will comment on how to deal with the general scenario.

Under these assumptions the eigenvalues of D and those of $D + \rho uu^*$ are different, and this implies

$$\begin{aligned} 0 &= \det(D + \rho uu^* - \lambda I) = \det(D - \lambda I)\det(I + \rho(D - \lambda I)^{-1}uu^*) \\ \Leftrightarrow \quad 0 &= \det(I + \rho(D - \lambda I)^{-1}uu^*) = 1 + \rho u^*(D - \lambda I)^{-1}u =: f(\lambda). \end{aligned}$$

Taking a closer look at $f(\lambda)$ we see that

$$f(\lambda) = 1 + \rho \sum_{j=1}^{n} \frac{u_j^2}{d_j - \lambda},$$

that is a rational function whose poles (vertical asymptotes) coincide with $d_1, \ldots, d_n$. The equation $f(\lambda) = 0$ is said *secular equation.*

With a direct computation we get that

$$f'(\lambda) = \rho \sum_{j=1}^{n} \frac{u_j^2}{(d_j - \lambda)^2},$$

that implies the strict monotonicity of $f(\lambda)$ in all the intervals where it is defined. In particular, $f(\lambda)$ has exactly n roots $\lambda_1, \ldots, \lambda_n$ and, in view of Courant-Fischer's result about the eigenvalues of sums of Hermitian matrices, we have that

$$\begin{aligned} d_j \leq \lambda_j \leq d_{j-1}, \qquad & \lambda_1 \geq d_1 \qquad && \text{if } \rho > 0, \\ d_{j+1} \leq \lambda_j \leq d_j, \qquad & \lambda_n \leq d_n \qquad && \text{if } \rho < 0. \end{aligned}$$

Since the intervals $[d_j, d_{j+1}]$ are a quite good estimate of where to find the λ_js and $f(\lambda)$ is monotonic and smooth inside these intervals, we can compute the roots of $f(\lambda)$ by means of n runs of an optimization method, like the Newton iteration. Once the values λ_j are computed, we can retrieve the associated eigenvectors of $D + \rho uu^*$ by leveraging the following result.

Lemma 3.3.1. *If λ_j is an eigenvalue of $D+\rho uu^*$ then $(D-\lambda_j I)^{-1}u$ is a corresponding eigenvector.*

Proof. With a direct computation we find

$$\begin{aligned}(D+\rho uu^*)(D-\lambda_j I)^{-1}u &= (D-\lambda_j I+\lambda_j I+\rho uu^*)(D-\lambda_j I)^{-1}u\\ &= u+\lambda_j(D-\lambda_j I)^{-1}u+[\underbrace{\rho u^*(D-\lambda_j I)^{-1}u}_{f(\lambda_j)-1}]u\\ &= u+\lambda_j(D-\lambda_j I)^{-1}u-u=\lambda_j(D-\lambda_j I)^{-1}u.\end{aligned}$$

□

The recursive procedure for computing the eigendecomposition of T, that directly comes from the analysis performed in this section, is reported in Algorithm 5. We point out that there are some criticalities that makes Algorithm 5 numerically unstable but the latter can be addressed in an effective manner. In particular, some care has to be taken at the following points:

- One has to handle the cases where $d_j = d_{j+1}$ or $u_j = 0$, for one or several values of j (deflation).
- When $|d_j - d_{j+1}|$ is small (but not negligible), the Newton's method struggle or even do not find the root of $f(\lambda)$ contained in $[d_{j+1}, d_j]$ because of the almost flat derivative of f in the inner part of the interval. To overcome this, a modified Newton's iteration has to be implemented.
- When $|\lambda_j - \lambda_{j+1}|$ is small, the formula for the eigenvector coming from Lemma 3.3.1 can be unstable. The problem is that $(D-\lambda_j I)^{-1}u$ and $(D-\lambda_{j+1}I)^{-1}u$ are supposed to yield orthogonal vectors. However, since the entry d_j is close to both λ_j and λ_{j+1}, there is a great deal of cancelation when evaluating $d_j - \lambda_j$, $d_j - \lambda_{j+1}$, and when evaluating the secular equation in the Newton iteration. In particular, the computed $d_j - \lambda_j$ and $d_j - \lambda_{j+1}$ may contain large relative errors and the corresponding eigenvectors may be inaccurate and far from orthogonal.

3.3.1 Handle deflation

First, observe that, by applying a permutation matrix Π on the left and its inverse Π^T on the right of $D + \rho uu^T$, we can always retrieve the condition $d_1 \geq d_2 \geq \cdots \geq d_n$ for the diagonal entries of D. In the case of $u_j = 0$ or $d_j = d_{j+1}$ for certain values of j the corresponding eigenvalues and eigenvectors can be given immediately. Indeed, if there are zero entries in u then we have

$$(u_j = 0 \quad \Leftrightarrow \quad u^T e_j = 0) \qquad \Rightarrow \qquad (D+\rho uu^T)e_j = d_j e_j. \tag{3.1}$$

Thus, if an entry of u vanishes we can read the eigenvalue from the diagonal of D at once and the corresponding eigenvector is a coordinate vector. If identical entries occur

Algorithm 5 Divide-and-conquer method for a tridiagonal symmetric $T \in \mathbb{C}^{n\times n}$

procedure D&C(T)
  if n=1 **then**
    $Q \leftarrow 1$, $\Lambda \leftarrow T$
  else
    $T = \begin{bmatrix} T_1 & \\ & T_2 \end{bmatrix} + \rho v v^*$
    $[Q_1, \Lambda_1] \leftarrow$ D&C(T_1)
    $[Q_2, \Lambda_2] \leftarrow$ D&C(T_2)
    Compute D and u
    Retrieve the eigenvalues of $D + \rho u u^*$ by means of the Newton method
    Compute the eigenvector matrix Q' of $D + \rho u u^*$ with Lemma 3.3.1
  end if
end procedure

in the diagonal of D, say $d_j = d_{j+1}$, then we can find a Givens rotation G such that it introduces a zero into the $j+1$-th position of u

$$Gu = \begin{bmatrix} u_1 \\ \vdots \\ u_{j-1} \\ \sqrt{u_j^2 + u_{j+1}^2} \\ 0 \\ u_{j+2} \\ \vdots \\ u_n \end{bmatrix}.$$

Note that, since $d_j = d_{j+1}$ we also have the property $GDG^T = D$. So, if there are multiple eigenvalues in D we can reduce all but one of them by introducing zeros in u and then proceed as previously in (3.1). When working with floating point numbers we deflate if

$$|u_i| \leq Cu||T||_2, \qquad |d_j - d_{j+1}| \leq Cu||T||_2,$$

where C is a small constant and u is the machine precision.

3.3.2 Modified Newton iteration (Additional, not done during the lectures)

As said previously, Newton's iteration to solve $f(\lambda) = 0$ does not work when some weights u_i are small, as the tangent at certain points in (d_{i+1}, d_i) crosses the real axis outside this interval. Since $f(\lambda)$ is not well approximated (locally) by a straight line, the zero finder has to be adapted in such a way that it captures the poles at the interval endpoints. The idea is to choose the next point in the Newton iteration as the zero in

(d_{i+1}, d_i) of the ansatz

$$h(\lambda) = \frac{c_1}{\lambda - d_i} + \frac{c_2}{\lambda - d_{i+1}} + c_3, \tag{3.2}$$

for certain parameters c_1, c_2, c_3. Once the parameters c_i are fixed, the solution of $h(\lambda) = 0$ are obtained by solving the equivalent quadratic equation

$$c_1(d_{i+1} - \lambda) + c_2(d_i - \lambda) + c_3(d_i - \lambda)(d_{i+1} - \lambda) = 0.$$

To determine the parameters c_i we rewrite $f(\lambda)$ as

$$f(\lambda) = 1 + \underbrace{\rho \sum_{j=1}^{i} \frac{u_j^2}{\lambda - d_j}}_{\psi_1(\lambda)} + \underbrace{\rho \sum_{j=i+1}^{n} \frac{u_j^2}{\lambda - d_j}}_{\psi_2(\lambda)} = 1 + \psi_1(\lambda) + \psi_2(\lambda).$$

For the current iterate $\widetilde{\lambda} \in (d_{i+1}, d_i)$ of the (modified) Newton iteration, we look for approximants $h_1(\lambda), h_2(\lambda)$ of $\psi_1(\lambda)$ and $\psi_2(\lambda)$, such that

$$\begin{aligned}
&h_1(\lambda) = \widehat{c_1} + \frac{c_1}{d_i - \lambda}, && h_1(\widetilde{\lambda}) = \psi_1(\widetilde{\lambda}), && h_1'(\widetilde{\lambda}) = \psi_1'(\widetilde{\lambda}),\\
&h_2(\lambda) = \widehat{c_2} + \frac{c_2}{d_{i+1} - \lambda}, && h_2(\widetilde{\lambda}) = \psi_2(\widetilde{\lambda}), && h_2'(\widetilde{\lambda}) = \psi_2'(\widetilde{\lambda}).
\end{aligned}$$

It is easy to verify that this yields the following expressions for c_1, c_2, c_3:

$$\begin{aligned}
&c_1 = \psi_1'(\lambda)(d_i - \widetilde{\lambda})^2, && \widehat{c_1} = \psi_1(\widetilde{\lambda}) - \psi_1'(\lambda)(d_i - \widetilde{\lambda}),\\
&c_2 = \psi_2'(\lambda)(d_{i+1} - \widetilde{\lambda})^2, && \widehat{c_2} = \psi_2(\widetilde{\lambda}) - \psi_2'(\lambda)(d_{i+1} - \widetilde{\lambda}),\\
&c_3 = 1 + \widehat{c}_1 + \widehat{c}_2.
\end{aligned}$$

We remark that, the stable implementation of the eigenvectors computation requires that the secular equation solver is applied to the shifted function $g(h) = f(d_j + h)$ where d_j is the diagonal entry of D that is closest to the root. If, for instance, we look for the root in (d_{i+1}, d_i), we can understand who of the two extrema is the closest by evaluating $f(\lambda)$ at the mid point of the interval.

3.3.3 Computing the eigenvectors stably

To compute the eigenvector stably we can not directly rely on the formula given in Lemma 3.3.1, when the difference $\lambda_j - d_i$ has an error of size $\mathcal{O}(u|d_i|)$ instead of only $\mathcal{O}(u|\lambda_j - d_i|)$. To deal with this issue, we rely on the previously described shift of variables in the secular equation solver, to directly approximate the smallest distance $\lambda_j - d_i$, for every i. Once the eigenvalues are approximated, a vector $\widehat{u}$ could be computed such that the λ_i are accurate eigenvalues of $D + \rho\widehat{u}\widehat{u}$. If $\widehat{u}$ approximates well the original u then the new eigenvectors will be the exact eigenvectors of a slightly modified eigenvalue problem, which is all we can hope for. To get vector $\widehat{u}$, we rely on the following result.

Lemma 3.3.2. *Let $D = \operatorname{diag}(d_1, \ldots, d_n)$ with $d_1 > \cdots > d_n$, $\rho > 0$, and $\lambda_1 > \cdots > \lambda_n$ satisfying the interlacing property*

$$d_n < \lambda_n < \cdots < d_{i+1} < \lambda_i < d_i < \cdots < d_1 < \lambda_1.$$

Then there is a vector $\widehat{u}$ such that the λ_i are the exact eigenvalues of $\widehat{D} = D + \rho\widehat{u}\widehat{u}^T$. The entries of $\widehat{u}$ are given by

$$|\widehat{u}_i| = \left[\frac{\prod_{j=1}^{n} (\lambda_j - d_i)}{\rho \prod_{\substack{j=1 \\ j \neq i}}^{n} (d_j - d_i)} \right]^{\frac{1}{2}}.$$

Proof. The characteristic polynomial of $\widehat{D}$ can be written both as $\det(\widehat{D} - \lambda I) = \prod_{j=1}^{n}(\lambda_j - \lambda)$ and as

$$\begin{aligned} \det(\widehat{D} - \lambda I) &= \left[\prod_{j=1}^{n}(d_j - \lambda)\right] \cdot \left(1 + \rho \sum_{j=1}^{n} \frac{\widehat{u}_j^2}{d_j - \lambda}\right) \\ &= \left[\prod_{j=1}^{n}(d_j - \lambda)\right] \cdot \left(1 + \rho \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{\widehat{u}_j^2}{d_j - \lambda}\right) + \rho \left[\prod_{\substack{j=1 \\ j \neq i}}^{n} (d_j - \lambda)\right] \cdot \widehat{u}_i^2. \end{aligned}$$

Setting $\lambda = d_i$ and equating both expressions for $\det(\widehat{D} - \lambda I)$ we get

$$\widehat{u}_i^2 = \frac{\prod_{j=1}^{n}(\lambda_j - d_i)}{\rho \prod_{\substack{j=1 \\ j \neq i}}^{n} (d_j - d_i)}.$$

□

Remark 3.3.3. The sign of each entry $\widehat{u}_i$ of $\widehat{u}$ is chosen as the one of the corresponding entry in u, so that $D + \rho\widehat{u}\widehat{u}^T$ and $D + \rho u u^T$ are close. A result analogous to Lemma 3.3.2 holds for the case $\rho < 0$ and the associated interlacing property.

Finally, the stable procedure for computing the eigenvectors works as follows:

(*i*) Solve the (shifted) secular equations to get the approximated eigenvalues λ_j and the smallest distances $\lambda_j - d_i$.

(*ii*) Compute $\widehat{u}$ by means of the formula given in Lemma 3.3.2 and Remark 3.3.3.

(*iii*) Compute the eigenvectors of $D + \rho\widehat{u}\widehat{u}^T$ via Lemma 3.3.1.

3.4 The Singular Value Decomposition

We shall now introduce an important factorization for a generic rectangular matrix A, which is called *singular value decomposition (SVD)*. The idea behind this factorization is to decompose any linear operator as the product of three matrices, here reported for the case $m \geq n$:

$$A = U\Sigma V^*, \qquad \Sigma = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \\ & & \end{bmatrix},$$

The matrices U, V are unitary, Σ is real and diagonal, and $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$. Here by "diagonal" we mean that Σ may be rectangular, but has non-zero entries only on the diagonal entries Σ_{ii}. The case reported above is for $m \geq n$, but the analogue definition can be given for $n \geq m$.

Geometrically, we can interpret this factorization as the decomposition of the action of A into an isometry, followed by a (non-negative) scaling of the axes, and then again by an isometry. The factorization can be used to provide several explicit solutions to computational problems.

3.4.1 Existence and uniqueness

We now prove that the singular value decomposition exists for any matrix.

Theorem 3.4.1 (Existence of the SVD)**.** *Let $A \in \mathbb{C}^{m\times n}$ with $m \geq n$. Then, there exist two unitary square matrices U, V of size $m \times m$ and $n \times n$, respectively, and an $m \times n$ matrix Σ with non-negative diagonal with decreasing entries and zero elsewhere, such that $A = U\Sigma V^*$. If A is real, U and V can be chosen real as well.*

Proof. We prove this result by induction over n; let $n = 1$, and m be arbitrary. Then, A is a column vector and we may set

$$U = \begin{bmatrix} \frac{1}{\|A\|_2} A & B \end{bmatrix}, \qquad \Sigma = \begin{bmatrix} \|A\|_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad V = \begin{bmatrix} 1 \end{bmatrix},$$

where $B \in \mathbb{C}^{m\times(m-1)}$ is a completion of $\frac{1}{\|A\|_2} A$ to an orthogonal basis of $\mathbb{C}^m$. By direct verification, we have $A = U\Sigma V^*$, and the three matrices satisfy all the requirements to be an SVD of A.

Assume now that the result is valid for $n - 1$ (and arbitrary m). Then, by definition of spectral norm there exists a vector v_1 of unit norm such that

$$w = Av_1, \qquad \|w\|_2 = \|A\|_2.$$

If $A = 0$ the SVD is obtained in a trivial way, hence we can assume that $\|w\|_2 \neq 0$ and define the matrices $\hat{U}, \hat{V}$ as follows:

$$\hat{U} := \begin{bmatrix} \frac{w}{\|w\|_2} & w_2 & \dots & w_m \end{bmatrix}, \qquad \hat{V} := \begin{bmatrix} v_1 & v_2 & \dots & v_n \end{bmatrix},$$

where $w_2, \dots, w_m$ and $v_2, \dots, v_n$ are chosen as any unitary completion of the first column. We now claim that the matrix $\hat{U}^* A \hat{V}$ has the following form:

$$\hat{U}^* A \hat{V} = \left[\begin{array}{c|ccc} \|A\|_2 & 0 & \dots & 0 \\ \hline 0 & & & \\ \vdots & & \hat{A} & \\ 0 & & & \end{array}\right].$$

The fact that the entry in position $(1,1)$ is equal to $\|A\|_2$ can be verified directly:

$$(\hat{U}^* A \hat{V})_{11} = (\hat{U} e_1)^T A (\hat{V} e_1) = \frac{1}{\|w\|_2} w^* A v_1 = \frac{w^* w}{\|w\|_2} = \|w\|_2 = \|A\|_2. \tag{3.3}$$

If any other entry in the first column or row were different from zero, then the matrix A would have a column or row with Euclidean norm strictly larger than $\|A\|_2$, which is a contradiction. Hence, the sparsity structure in (3.3) is an immediate consequence of $(\hat{U}^* A \hat{V})_{11} = \|A\|_2$.

We may now make use of the inductive hypothesis to obtain an SVD of $\hat{A} = \tilde{U} \tilde{\Sigma} \tilde{V}^*$, and to write the following decomposition for A:

$$A = \underbrace{\hat{U} \begin{bmatrix} 1 & \\ & \tilde{U} \end{bmatrix}}_{U} \left[\begin{array}{c|ccc} \|A\|_2 & & & \\ \hline & \tilde{\sigma}_1 & & \\ & & \ddots & \\ & & & \tilde{\sigma}_{n-1} \end{array}\right] \underbrace{\begin{bmatrix} 1 & \\ & \tilde{V}^* \end{bmatrix} \hat{V}^*}_{V^*}.$$

Up to calling the above unitary matrices U and V, and setting $\sigma_1 := \|A\|_2$ and $\sigma_i := \tilde{\sigma}_{i-1}$ for $i > 1$, this is an SVD of the matrix A. The only remaining fact to check is that the singular values are in decreasing order, that is $\tilde{\sigma}_1 \leq \|A\|_2$. To this aim, we may note that for a diagonal matrix the norm is the maximum of the moduli of the entry on the diagonal, which in turn implies $\max\{\|A\|_2, \tilde{\sigma}_1\} = \|A\|_2$ and therefore $\tilde{\sigma}_1 \leq \|A\|_2$. □

An SVD of A is not necessarily unique. We observe that, given any diagonal unitary matrix D, we can diagonally scale U and V to obtain $A = U \Sigma V^* = U D \Sigma D^* V^*$. Since $UD \neq U$ (unless $D = I$), we have an infinite number of different singular value decompositions.

Exercise 3.4.2. Generically, the singular vector scaling described above is the only degree of freedom in the decomposition. This is true unless two consecutive singular values coincide (that is, $\sigma_i = \sigma_{i+1}$ for some i). Prove that this is true, and characterize the additional degrees of freedom obtained in the non-generic situation.

Even though the factorization itself is not uniquely defined, the singular values are. This will be immediately apparent when we will prove the properties of the SVD; we will often need to refer to the ith singular value of A, for which we use the notation $\sigma_i(A)$.

3.4.2 Properties of the SVD

We now present a few essential properties of the singular value decomposition.

Lemma 3.4.3. *Let $A = U\Sigma V^*$ be a SVD of $A \in \mathbb{C}^{m\times n}$ with $m \geq n$. Then,*

(i) The symmetric positive definite matrix A^*A is diagonalized by V: $V^*A^*AV = \Sigma^*\Sigma = D$, and has σ_i^2 as eigenvalues with $i = 1, \ldots, n$.

(ii) The symmetric positive definite matrix AA^* is diagonalized by U: $U^*AA^*U = \Sigma\Sigma^* = D$, and has $m - n$ zero eigenvalues, and the others equal to σ_i^2.

(iii) The following symmetric matrix M has $\pm\sigma_i$ and $m - n$ zeros as eigenvalues:

$$M = \begin{bmatrix} & A^* \\ A & \end{bmatrix} \implies \Lambda(M) = \{\pm\sigma_i \mid \sigma_i \text{ singular value of } A\}.$$

Proof. The proof of (i) and (ii) are obtained by a direct computation. For what concerns M, we make the following observation:

$$\begin{bmatrix} V & \\ & U \end{bmatrix}^* \begin{bmatrix} & A^* \\ A & \end{bmatrix} \begin{bmatrix} V & \\ & U \end{bmatrix} = \begin{bmatrix} & \Sigma^* \\ \Sigma & \end{bmatrix} =: M_\Sigma.$$

Since M and the M_Σ are similar, they have the same eigenvalues, and we just need to prove that the eigenvalues of M_Σ are $\pm\sigma_i$ and the $m-n$ zeros. Consider the permutation π of $\{1, \ldots, m + n\}$ such that

$$\pi(i) = \begin{cases} \frac{i+1}{2} & i \equiv 1 \mod 2 \text{ and } i \leq 2n \\ \frac{i}{2} + n & i \equiv 0 \mod 2 \text{ and } i \leq 2n \\ i & 2n < i \leq m + n \end{cases}.$$

If Π is the permutation matrix associated with π, computing $\Pi^* M_\Sigma \Pi$ yields a block diagonal matrix of the following form:

$$\Pi^* M_\Sigma \Pi = \begin{bmatrix} \Sigma_1 & & & \\ & \ddots & & \\ & & \Sigma_n & \\ & & & 0_{m-n} \end{bmatrix}, \qquad \Sigma_i := \begin{bmatrix} 0 & \sigma_i \\ \sigma_i & 0 \end{bmatrix}.$$

The eigenvalues of the 2×2 matrices Σ_i are exactly $\pm\sigma_i$, so the claim follows. □

Remark 3.4.4. From the definition of the SVD we immediately derive the invariance of the singular values under unitary transformations: $\sigma_i(A) = \sigma_i(QA) = \sigma_i(AZ)$ for any choice of Q, Z unitary (orthogonal in the real case).

Lemma 3.4.5. *Let A be an $m \times n$ matrix, with $m \geq n$ and SVD $A = U\Sigma V^*$. Then, the following identities hold:*

$$\|A\|_2 = \sigma_1(A), \qquad \|A\|_F = \sqrt{\sigma_1^2 + \ldots + \sigma_n^2}.$$

Proof. The claim follows noting that, being the spectral and Frobenius norm invariant under unitary transformation, we have

$$\|A\|_{2/F} = \|U\Sigma V^*\|_{2/F} = \|\Sigma\|_{2/F},$$

and by the definition of spectral and Frobenius norms. □

Exercise 3.4.6. Show that, if a matrix is norm $\|\cdot\|$ is invariant under unitary transformation, then it can be written in the form $\|A\| = f(\sigma_1(A), \ldots, \sigma_n(A))$ for some f.

3.4.3 The Eckart-Young-Mirsky theorem

The singular value decomposition gives an explicit and constructive answer to the low-rank approximation problem of finding B of rank at most k that minimizes $\|A - B\|$, with respect to the spectral or Frobenius norm.

This has applications to data compression (a low-rank matrix is much cheaper to store than a full one), data analysis, and much more.

Theorem 3.4.7. *Let $A \in \mathbb{C}^{m\times n}$, and $A = U\Sigma V^*$ its SVD. Let A_k be defined as follows:*

$$A_k := U\Sigma_k V^*, \qquad \Sigma_k := \begin{bmatrix} \sigma_1 & & & \\ & \ddots & & \\ & & \sigma_k & \\ & & & \end{bmatrix},$$

where Σ_k is equal to Σ with $\sigma_{k+1}, \ldots, \sigma_{\min\{m,n\}}$ set to zero. Then, the following hold true:

(i) The matrix A_k satisfies $\sigma_{k+1} = \|A - A_k\|_2 \leq \|A - B\|_2$ for any matrix B with rank less or equal to k.

(ii) The matrix A_k satisfies $\sqrt{\sigma_{k+1}^2 + \ldots + \sigma_{\min\{m,n\}}^2} = \|A - A_k\|_F \leq \|A - B\|_F$ for any matrix B with rank less or equal to k.

To prove this result, we proceed as follows:

- First, we prove claim (i) for $\|\cdot\|_2$;
- Then, we use it to show an auxiliary Lemma about the singular values of $A_1 + A_2$, the sum of two arbitrary matrices.
- Finally, we use the Lemma to prove the result for (ii).

Proof of Theorem 3.4.7 for $\|\cdot\|_2$. We first check that $\|A - A_k\|_2 = \sigma_{k+1}$. For simplicity, we assume throughout the proof that $m \geq n$, the other case can be obtained transposing A, or with minimal modifications to the indices. Using the SVD, we get

$$A - A_k = U(\Sigma - \Sigma_k)V^* = U \begin{bmatrix} 0 & & & & & \\ & \ddots & & & & \\ & & 0 & & & \\ & & & \sigma_{k+1} & & \\ & & & & \ddots & \\ & & & & & \sigma_n \end{bmatrix} V^*.$$

Taking norms yields $\|A - A_k\|_2 = \|\Sigma - \Sigma_k\|_2 = \sigma_{k+1}$, where we have used the invariance of the spectral norm under unitary transformations, and that the 2-norm of a diagonal matrix is the maximum of the moduli of the diagonal elements.

To conclude we need to check that, for any matrix B of rank at most k, $\|A - B\|_2 \geq \sigma_{k+1}$. We choose a unit norm vector v from the subspace $\mathrm{Ker}(B) \cap \mathrm{colspan}\{v_1, \ldots, v_{k+1}\}$ where $v_j := Ve_j$ are the columns of V. Since $\mathrm{Ker}(B)$ has dimension at least $n - k$, the intersection of the subspaces has dimension at least 1, and this is always possible. We can write such vector v in coordinates with respect to the columns of V:

$$v = \sum_{j=1}^{k+1} \alpha_j v_j = V\alpha, \qquad \alpha := \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_{k+1} \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \qquad \sum_{j=1}^{k+1} |\alpha_j|^2 = 1.$$

This yields an explicit expression for $(A - B)v$, of the form

$$(A - B)v = Av = U\Sigma V^* v = U\Sigma\alpha = U \begin{bmatrix} \sigma_1\alpha_1 \\ \vdots \\ \sigma_{k+1}\alpha_{k+1} \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

The Euclidean norm of $(A - B)v$ can be bounded below as follows:

$$\|(A - B)v\|_2^2 = \sum_{j=1}^{k+1} \sigma_j^2 \alpha_j^2 \geq \sigma_{k+1}^2 \sum_{j=1}^{k+1} \alpha_j^2 = \sigma_{k+1}^2.$$

Since $\|A - B\|_2 \geq \|(A - B)v\|_2$ for any $\|v\|_2 = 1$, this proves the claim. □

We now use the proof of Theorem 3.4.7 in the spectral norm case to state a Lemma known as one of the Weyl's inequalities. These inequalities provide upper and lower bounds for the eigenvalues of symmetric matrices and singular values. In the symmetric case, the main tool to prove them is the Courant-Fischer theorem. A good reference for further info on this topic are [3, 4].

Lemma 3.4.8 (Weyl)**.** *Let A_1, A_2 be two matrices of compatible sizes, and $A = A_1 + A_2$. Then, for any $i, j \geq 0$,*

$$\sigma_{i+j+1}(A) \leq \sigma_{i+1}(A_1) + \sigma_{j+1}(A_2),$$

where we set $\sigma_k(A) = 0$ for any k larger than the smallest dimension of A.

Proof. Thanks to Theorem 3.4.7 we know that there exist two matrices $A_{i,1}$ and $A_{j,2}$ of rank at most i and j, respectively, such that

$$\|A_1 - A_{i,1}\|_2 = \sigma_{i+1}(A_1), \qquad \|A_2 - A_{j,2}\|_2 = \sigma_{j+1}(A_2).$$

If we set $B := A_{i,1} + A_{j,2}$ we have that $\operatorname{rank}(B) \leq i + j$, and therefore, again in view of Theorem 3.4.7,

$$\begin{aligned}\sigma_{i+j+1}(A) &\leq \|A - B\|_2 = \|A_1 - A_{i,1} + A_2 - A_{j-2}\|_2 \\ &\leq \|A_1 - A_{i,1}\|_2 + \|A_2 - A_{j,2}\|_2 = \sigma_{i+1}(A_1) + \sigma_{j+1}(A_2).\end{aligned}$$ □

Exercise 3.4.9. Show that Weyl's Lemma implies the subadditivity of the spectral norm as a corollary.

We now have all the tools to prove the second part of Theorem 3.4.7, concerning the Frobenius norm.

Proof of Theorem 3.4.7 for $\|\cdot\|_F$. The prove the second part of the theorem we start by verifying $\|A - A_k\|_F^2 = \sigma_{k+1}^2 + \ldots + \sigma_n^2$. This follows by the same argument used for the spectral norm, recalling that the square Frobenius norm is the sum of the squares of the entries in a matrix.

We now take B to be any matrix of rank at most k, and claim that

$$\|A - B\|_F^2 \geq \|A - A_k\|_F^2 = \sigma_{k+1}^2 + \ldots + \sigma_n^2.$$

Note that we may write

$$\|A - B\|_F^2 = \sum_{l=1}^{n} \sigma_l^2(A - B).$$

Decomposing $A = (A - B) + B$ and using Weyl's inequality with $i = l - 1$ and $j = k$ we get

$$\sigma_{l+k}^2(A) \leq \sigma_l^2(A - B) + \sigma_{k+1}^2(B) = \sigma_l^2(A - B).$$

Using this inequality in the previous identity and dropping the zero terms from the summation yields

$$\|A - B\|_F^2 = \sum_{l=1}^{n} \sigma_l^2(A - B) \geq \sum_{l=1}^{n-k} \sigma_{l+k}^2(A).$$ □

3.4.4 Computing the SVD

So far, we have not discussed how an SVD of a generic matrix A should be computed in practice, we have only proved its existence and uniqueness up to selected degrees of freedom.

Algorithms for the SVD can be obtained by trying to rephrase the computation to a symmetric eigenproblem. We now make the simplifying assumption that $m = n$, but it is not difficult to adapt all results with appropriate padding with zero.

The first step is to find orthogonal matrices Q, Z such that

$$QAZ^* = B = \begin{bmatrix} \times & \times & & \\ & \ddots & \ddots & \\ & & \ddots & \times \\ & & & \times \end{bmatrix}$$

This is easily achieved by adapting the construction already seen for the upper Hessenberg form, exploiting the degrees of freedom by having two different unitary matrix Q, Z, instead of imposing $Q = Z$. The cost of this reduction is $\mathcal{O}(n^3)$ flops for an $n \times n$ matrix.

Computing the SVD of B is equivalent to computing the SVD of A. Indeed, if $B = U\Sigma V^*$ then we can retrieve an SVD of A by:

$$A = QBZ^* = QU\Sigma V^*Z^* = (QU)\Sigma(ZV)^*.$$

Thanks to this observation, we shall now assume that A is bidiagonal from the beginning.

Recall that the SVD is linked with the eigenvalue problems for AA^* and A^*A, which are both tridiagonal matrices. Hence, we can use the QR iteration to find one of

$$U^*AA^*U = \Sigma^2, \qquad V^*A^*AV = \Sigma^2.$$

Any of these choices produce a sequence of tridiagonal matrices T_ℓ that converges to Σ^2. However, computing the square of the singular values is inconvenient from the numerical view point, because will make small singular values even smaller, which amplifies their relative error. Therefore, if after ℓ steps of QR iteration we have $Q_\ell^*AA^*Q_\ell = T_\ell$, we select a unitary matrix Z_ℓ such that

$$Q_\ell^*AZ_\ell Z_\ell^*A^*Q_\ell = T_\ell,$$

by imposing that $Q_\ell^*AZ_\ell$ is upper bidiagonal. Such Z_ℓ can be easily computed directly into the chasing procedure, by considering $B_\ell := Q_\ell^*AZ_\ell$ instead of T_ℓ. It is then easy to check that we must have $Q_\ell^*AZ_\ell \to \Sigma$, and therefore $Q_\ell \to U^*$ and $Z_\ell \to V^*$. This idea is equivalent to approximating the factors of the Cholesky factorization of $T_\ell = LL^T$ (L here is lower or upper triangular), which exist for all symmetric positive definite matrices, and allow to avoid the problem of squaring the singular values.

Essentially all algorithms for symmetric eigenproblems (and not just the tridiagonal QR) can be adapted to compute the SVD. We do not discuss them further, but details on these ideas can be found in [1].

4 Least squares problems

Whenever a matrix A is square and invertible, solving a linear system $Ax = b$ can be pursued with standard tools such as the LU decomposition or the QR factorization. The condition number of this problem is well known to be $\|A\|\|A^{-1}\|$, and for the spectral norm can be linked to the singular values by $\sigma_1(A)/\sigma_n(A)$.

We now focus on the more general problem of computing the solution to $Ax = b$ whenever:

- A is rectangular, with more rows than column (an *overdetermined* system).
- A is rectangular, with more columns than rows (an *underdetermined* system).
- Any of the above, possibly even a square case, but with A with a large condition number, so that the problem is intractable with standard tools, because the solution would be completely polluted by floating point errors.

All these cases can be effectively analyzed and solved using the SVD. Note that they can be wildly different: in the first case there could be no solution, whereas in the second we may end up with a lot of different solutions and we need a way to select the "right" one.

In all the above cases, where the solution of the linear system may not exist, it makes sense to transform the problem into finding a minimizer for the functional

$$\Phi(x) := \|Ax - b\|_2^2.$$

When there will be multiple minimizers, we will add additional constraints to select the desired ones (for instance, by picking the one that minimizes $\|x\|_2$ as well).

4.1 Normal equations for overdetermined full rank least squares problems

We now make the simplifying assumption that A is $m \times n$ with $m \geq n$, and full rank. Let us rewrite $\Phi(x)$ in a way that is more amenable to compute its derivatives. In the real case we have

$$\Phi(x) = (Ax - b)^T(Ax - b) \qquad \nabla\Phi(x) = 2(A^TAx - A^Tb)$$
$$\nabla^2\Phi(x) = 2A^TA.$$

If A is full rank, then A^TA is positive definite and therefore $\Phi(x)$ is convex, and has a unique minimum. A similar derivation can be given in the complex case considering the derivatives $\frac{\partial}{\partial z}$ and $\frac{\partial}{\partial \bar{z}}$. Hence, the minimum point can be found by setting solving

$$\nabla\Phi(x) = 0 \iff A^*Ax = A^*b.$$

The linear system with A^*A arising from these considerations goes under the name of *normal equations*. They are a good choice for well-conditioned problems (more on this later), but can encounter loss of accuracy otherwise.

We have not yet defined precisely what the condition number of a least squares problems should be, but it is worth noting that the condition number of A^*A is given by $\sigma_1^2(A)/\sigma_n^2(A)$. Indeed, using $A = U\Sigma V^*$,

$$\kappa_2(A^*A) = \|A^*A\|_2\|(A^*A)^{-1}\| = \|V\Sigma^*\Sigma V\|_2\|V(\Sigma^*\Sigma)^{-1}V\|_2 = \frac{\sigma_1^2(A)}{\sigma_n^2(A)}.$$

If we apply normal equations in the particular case of a square and invertible matrix A, we notice already that we are solving a problem that is numerically more challenging than it should be: we would expect a condition number of $\sigma_1(A)/\sigma_n(A)$, and instead we are faced with its square. This situation can be avoided by relying on different solution methods.

4.1.1 Solving least squares problem using the QR and the SVD

Consider a full-rank overdetermined least squares problem of the form $\min \|Ax - b\|_2^2$. If b does not belong to the column span of A, we cannot hope to achieve an exact solution $Ax = b$. To describe the possible situations, we introduce the definition of angle between a vector and a subspace.

Definition 4.1.1. Let $\mathcal{U} \subseteq \mathbb{C}^n$ be a subspace, and v any non-zero vector in $\mathbb{C}^n$. Then, the cosine and sine of the *angle* $\theta(\mathcal{U}, v)$ between v and $\mathcal{U}$ are defined by

$$\cos\theta(\mathcal{U}, v) := \frac{\|\Pi_{\mathcal{U}} v\|_2}{\|v\|_2}, \qquad \sin\theta(\mathcal{U}, v) := \frac{\|\Pi_{\mathcal{U}^\perp} v\|_2}{\|v\|_2}.$$

where $\Pi_{\mathcal{U}}$ and $\Pi_{\mathcal{U}^\perp}$ denote the orthogonal projections on $\mathcal{U}$ and $\mathcal{U}^\perp$, respectively. By a slight abuse of definition, we write $\theta(A, v)$ to denote $\theta(\mathrm{colspan}(A), v)$.

Exercise 4.1.2. Show that this definition matches the one for the cosine of the angle between two vectors v and w by considering $\mathcal{U} = \mathrm{span}(w)$, and this makes the notation $\theta(w, v)$ defined above compatible with the previous definition.

Clearly, an exact solution to $Ax = b$ exists if and only if $\theta(A, b) = 0$. How do we compute the angle? If $A = QR$ is an economy-size QR factorization of A, we may write the orthogonal projection onto $\mathrm{colspan}(A)$ as QQ^*, and therefore:

$$\cos(\theta(A, b)) = \|QQ^*b\|_2/\|b\|_2 = \|Q^*b\|_2/\|b\|_2.$$

Similarly, we can write $\sin(\theta(A, b)) = \|QQ^*b - b\|_2/\|b\|_2$. The economy-size QR factorization can be used to provide an explicit expression of the solution and of the residual.

Lemma 4.1.3. *Let $A \in \mathbb{C}^{m\times n}$ with $m \geq n$ a full rank matrix, and $A = QR$ its economy-size QR factorization. Then, the solution of the least squares problem $\min_x \|Ax - b\|_2^2$ is given by $x = R^{-1}Q^*b$, and the residual is equal to $\|(I - QQ^*)b\|_2$.*

Proof. If $A = QR$ is an economy-size QR factorization of A, we may write

$$Ax - b = QRx - b = Q(Rx - Q^*b) + (I - QQ^*)b,$$

where we have used $I = QQ^* + (I - QQ^*)$. Taking norms yields

$$\|Ax - b\|_2^2 = \|Rx - Q^*b\|_2^2 + \|(I - QQ^*)b\|_2^2,$$

where we have exploited the identity $\|v + w\|_2^2 = \|v\|_2^2 + \|w\|_2^2$ whenever $v \perp w$. Clearly, the term on the right is independent of x, and therefore the residual of the least squares system satisfies $\|Ax - b\|_2 \geq \|(I - QQ^*)b\|_2$ no matter what we choose for x. On the other hand, we can make the norm of the first term equal to zero by choosing $x = R^{-1}Q^*b$, since R is invertible thanks to the full-rank assumption for A. □

Lemma 4.1.3 provides a theoretical characterization of the solution and the residual, but it also gives an algorithm to compute it. We may note that this algorithm requires to solve a linear system with R, whereas for normal equations we needed to solve one with A^*A. If $A = U\Sigma V^*$, we can write a singular value decomposition for R as follows:

$$R = Q^*A = Q^*U\Sigma V^* \implies \frac{\sigma_1(R)}{\sigma_n(R)} = \frac{\sigma_1(A)}{\sigma_n(A)}.$$

In particular, the condition number of the linear system under consideration is the square root of the one of A^*A. We may wonder if this is truly the condition number of the underlying problem, or we can still improve the situation. The next lemma shows that what we are doing is already optimal.

Lemma 4.1.4. *Let $A \in \mathbb{C}^{m\times n}$ be a full rank matrix, and $b, \delta b \in \mathbb{C}^m$. Then, if x is the solution of the least squares problem $\min_x \|Ax - b\|_2$ and $x + \delta x$ the one of $\min_x \|A(x + \delta x) - b - \delta b\|_2$, we have*

$$\frac{\|\delta x\|_2}{\|x\|_2} \leq \frac{\sigma_1(A)}{\sigma_n(A)} \frac{\|\delta b\|_2}{\|b\|_2} \frac{1}{\cos(\theta(A, b))}.$$

Proof. Thanks to Lemma 4.1.3 we can explicitly write the solutions to the least squares problem using the QR factorization of $A = QR$:

$$x = R^{-1}Q^*b \qquad\qquad x + \delta x = R^{-1}Q^*(b + \delta b).$$

Subtracting these two terms implies $\delta x = R^{-1}Q^*\delta b$, which gives the upper bound

$$\|\delta x\|_2 \leq \|R^{-1}\|_2 \|Q^*\delta b\|_2 \leq \frac{1}{\sigma_n(A)} \|\delta b\|_2,$$

where we have used $\sigma_n(A) = \sigma_n(R)$ and that for an $n \times n$ square matrix M it holds $\|M^{-1}\|_2 = 1/\sigma_n(M)$. We can use once more the singular value decomposition of R to write a lower bound for $\|x\|_2$ as follows:

$$\|x\|_2 \geq \sigma_n(R^{-1}) \|Q^*b\|_2 = \frac{1}{\sigma_1(R)} \|b\|_2 \cos(\theta(A, b)).$$

Combining the two inequalities yields the sought claim. □

The QR factorization is an effective method for solving the least square problem. Recall that the QR factorization of A can be compute through a sequence of Householder reflectors, obtaining a sequence of partial reductions

$$P_k \dots P_1 A = \begin{bmatrix} R_k & X_k \\ & Y_y \end{bmatrix} = \begin{bmatrix} \times & \dots & \times & \times & \dots & \times \\ & \ddots & \vdots & \vdots & & \vdots \\ & & \times & \vdots & & \vdots \\ & & & \times & \dots & \times \\ & & & \vdots & & \vdots \\ & & & \times & \dots & \times \end{bmatrix},$$

where $R_k \in \mathbb{C}^{k\times k}, X_k \in \mathbb{C}^{k\times(n-k)}, Y_k \in \mathbb{C}^{m-k,n-k}$. Applying each of these reflectors P_j to A costs $\mathcal{O}(mn)$ flops. Hence, the cost of finding the R in the QR factorization is of $\mathcal{O}(mn^2)$ flops, since n reflectors are required. The tall and thin Q matrix can be computed at the same cost, but this is not really necessary for solving a least squares problem.

Indeed, since $Q^* = \begin{bmatrix} I_n & 0 \end{bmatrix} P_n \dots P_1$, we only need to compute

$$x = R^{-1}Q^*b = R^{-1}(\begin{bmatrix} I_n & 0 \end{bmatrix} P_n \dots P_1 b).$$

Hence, we can apply the reflectors to b while we compute them and apply them to A and its partial reductions, and extract its first n rows before solving the linear system with R.

4.2 Underdetermined and rank-deficient systems

We now consider the case where A is possibly not-full rank, or the linear system is underdetermined (i.e., $n > m$). In these cases, there may be multiple solutions to the minimum problem $\|Ax - b\|_2$, and to have a well-defined problem we need to choose the one to pick.

Let $\mathcal{S}(A, b)$ be the set of minimizers for the least squares problem:

$$\mathcal{S}(A,b) := \{x \in \mathbb{C}^n \mid \forall y \in \mathbb{C}^n \ \|Ax - b\|_2 \leq \|Ay - b\|_2.\} \tag{4.1}$$

Then, we define the minimum norm solution to $\|Ax - b\|_2$ as $x = \arg\min_{x\in\mathcal{S}(A,b)} \|x\|_2$. It may not be immediately clear that this x is well-defined (i.e., that there is a unique minimum point). This is however true, and we will characterize it with the introduction of the Moore-Penrose pseudoinverse.

Definition 4.2.1. Let $A \in \mathbb{C}^{m\times n}$ with SVD $A = U\Sigma V^*$. Then, the Moore-Penrose pseudoinverse of A is the $n \times m$ matrix

$$A^\dagger := V\Sigma^\dagger U^*, \qquad \Sigma^\dagger = \operatorname{diag}(\sigma_1^\dagger, \dots, \sigma^\dagger_{\min\{m,n\}}) \in \mathbb{C}^{n\times m}$$

where $\sigma_j^\dagger = 1/\sigma_j$ if $\sigma_j \neq 0$, or 0 otherwise.

If A is an invertible square matrix, then the pseudoinverse is exactly the standard matrix inverse, and we have $A^\dagger = A^{-1}$.

Exercise 4.2.2. Prove that if A is $m \times n$ with $m \geq n$ and full-rank, then $A^\dagger = (A^*A)^{-1}A^*$. If instead $n \geq m$, and A is full rank, then $A^\dagger = A(AA^*)^{-1}$.

Exercise 4.2.3. Prove that $(A^*)^\dagger = (A^\dagger)^*$.

Theorem 4.2.4. *Let $A \in \mathbb{C}^{m\times n}$, and $\mathcal{S}(A,b)$ the set of least squares solutions to $\min \|Ax-b\|_2$ as in (4.1). Then, the vector $x := A^\dagger b$ satisfies $x = \arg\min_{x\in\mathcal{S}(A,b)} \|x\|_2$.*

Proof. Let p be the number of singular values of A equal to zero. Then, we may write the SVD of A in blocks as follows:

$$A = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \left[\begin{array}{c|c} \Sigma_1 & 0_{p,n-p} \\ \hline 0_{m-p,p} & 0_{m-p,n-p} \end{array}\right] \begin{bmatrix} V_1 & V_2 \end{bmatrix}^*$$

We may write the residual $r := Ax - b$ as follows:

$$Ax - b = U(\Sigma y - U^*b), \qquad y := V^*x = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}, \qquad U^*b =: \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}.$$

The partitioning of y and U^*b is chosen according to the partitioning of the SVD of A, and using the invariance of the Euclidean norm under unitary transformations we obtain

$$\|Ax-b\|_2^2 = \left\| \begin{bmatrix} \Sigma_1 & 0_{p,n-p} \\ 0_{m-p,p} & 0_{m-p,n-p} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} - \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \right\|_2^2 = \left\| \begin{bmatrix} \Sigma_1 y_1 \\ 0 \end{bmatrix} - \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \right\|_2^2.$$

All minimizers in $\mathcal{S}(A,b)$ are obtained by setting $y_1 = \Sigma_1^{-1}b_1$, and choosing y_2 arbitrarily. Hence, since $y = V^*x$ we have

$$\|x\|_2^2 = \|y\|_2^2 = \|y_1\|_2^2 + \|y_2\|_2^2,$$

and therefore we have a unique minimum norm solution obtained by choosing $y_2 = 0$. We now need to check that this solution is exactly $A^\dagger b$:

$$A^\dagger b = V \begin{bmatrix} \Sigma_1^{-1} & 0^{p,m-p} \\ 0_{n-p,p} & 0_{n-p,m-p} \end{bmatrix} U^*b = V \begin{bmatrix} \Sigma^{-1}b_1 \\ 0 \end{bmatrix} = V \begin{bmatrix} y_1 \\ 0 \end{bmatrix}. \qquad \square$$

The notation $x = A^\dagger b$ is a handy way to write "the least-squares solution to $Ax = b$", no matter what the sizes or rank of A. Hence, we will often use it in the following section. However, depending on the features of A, actually computing the pseudoinverse may not be the best or more efficient way to computing x, as we have already discussed.

5 Krylov methods for linear systems

We now focus on the problem of solving a linear system $Ax = b$, assuming that A is large and structured. Indeed, we know already that if A is "small", say of size at most 1000×1000, then the LU decomposition, the Cholesky or the QR factorization are all valid choices for this task.

However, we will often encounter problems where we are able to efficiently compute $v \mapsto Av$ (and, if we are lucky $w \mapsto A^T w$), but unable to store all entries of A explicitly. The most relevant example is when A is sparse, i.e., only $\mathcal{O}(1)$ entries per row are non-zero.

Is the information obtained by performing matrix-vector products enough to solve a linear system? It turns out that the answer is often yes, and the natural tool to answer this question are Krylov subspaces.

5.1 Introduction to Krylov subspaces

An immediate observation is that, by combining at most $\ell - 1$ products of A times a vector, we can build all vectors of the form $p(A)b$ where $p(z)$ is a polynomial of degree at most $\ell - 1$. Is this enough to represent the solution x?

Lemma 5.1.1. *Let A be any invertible square $n \times n$ matrix. Then, there exists a polynomial $p(z)$ of degree at most $n-1$ such that, for any $b \in \mathbb{C}^n$, $x = A^{-1}b = p(A)b$.*

Proof. The result is a simple consequence of the Hamilton-Cayley theorem, that tells us that if $q(z) := \det(zI - A)$ then $q(A) = 0$. On the other hand, we have $q(0) = \det(A)$, so we can rephrase this statement as follows:

$$0 = q(A)b = \det(A)b + \sum_{j=1}^{n} q_j A^j b \implies b = \frac{-1}{\det A} \sum_{j=1}^{n} q_j A^j b.$$

Multiplying the above identity on the left by A^{-1} yields the claim:

$$x = A^{-1}b = \left[\frac{-1}{\det A} \sum_{j=0}^{n-1} q_j A^j \right] b =: p(A)b. \qquad \square$$

Lemma 5.1.1 gives us a good and a bad news at the same time:

- The solution to the linear system $Ax = b$ can be represented as a polynomial in A multiplied by b: there is hope to extract all the required information from matrix-vector products.
- The degree of such polynomial may be high, to the point of making such method not practical.

The idea behind Krylov subspaces is that, even though the exact $p(z)$ may be a high-degree polynomial (it generally is), we may be able to find a lower degree approximation that yields a good approximation $x \approx p_\ell(A)b$.

Definition 5.1.2. The *Krylov subspace of order* ℓ associated with A and b is the subspace

$$\mathcal{K}_\ell(A, b) := \operatorname{span}(b, Ab, \ldots, A^{\ell-1}b).$$

Often, we will find that $\dim \mathcal{K}_\ell(A, b) = \ell$, unless b belongs to a lower dimensional invariant subspace. The idea behind Krylov methods is to look for an approximate solution inside the subspace $\mathcal{K}_\ell(A, b)$. Depending on the criterion used to identity the "optimal" solution, we can construct different methods. For instance, we may consider the following options:

- We may ask that the residual $r_\ell := b - Ax_\ell$ is *orthogonal* to $\mathcal{K}_\ell(A, b)$. This choice yields the Full Orthogonal Method (FOM).
- We may ask that the residual r_ℓ is *minimized* among all possible vectors $x_\ell \in \mathcal{K}_\ell(A, b)$. That is, we choose

  $$x_\ell := \arg\min_{x \in \mathcal{K}_\ell(A,b)} \|b - Ax\|_2.$$

  This choice yields the Generalized Minimal Residual Method (GMRES).

Even though we know that for $\ell = n$ both choices will give us $r_\ell = 0$, we are interested in understanding the behavior for $\ell \ll n$.

5.2 The Arnoldi iteration

Whenever we deal with subspaces, we do so by constructing appropriate bases. The Krylov subspace $\mathcal{K}_\ell(A, b)$ is defined as the column span of the vectors $A^j b$ for $j = 0, \ldots, \ell - 1$. These vectors, however, form a terrible basis from the computational point of view.

Exercise 5.2.1. Show that if A is diagonal then the basis matrix

$$M = \begin{bmatrix} b & Ab & \ldots & A^{\ell-1}b \end{bmatrix}$$

is a (scaled) Vandermonde matrix with the eigenvalues of A as nodes. Show that for any normal matrix Q the resulting M is unitarily similar to such scaled Vandermonde matrix.

From the analysis of subspace iteration algorithms, we know that "good" bases are orthogonal, and therefore we should aim at constructing an orthogonal basis for $\mathcal{K}_\ell(A, b)$. In principle, this may be achieved by taking a QR factorization of the matrix with columns $A^j b$. When the latter matrix is formed, however, the damage is already done.

We need to tweak the procedure of taking matrix-vector products with A to obtain an orthogonal basis directly, by doing the reorthonormalization throughout the procedure. The resulting algorithm is called *Arnoldi iteration*, and can be described as follows:

- We choose an initial vector $v_1 := b/\|b\|_2$
- For $j = 1, 2, \ldots$ we compute the action of A on v_j, by setting $w_{j+1} := Av_j$.

- The vector w_j is orthogonalized with respect to the previous basis elements, and then normalized:
$$v_{j+1} := \frac{w_{j+1} - \sum_{i \leq j} (v_i^* w_{j+1}) v_i}{\|w_{j+1} - \sum_{i \leq j} (v_i^* w_{j+1}) v_i\|_2}$$

This procedure returns, for every ℓ, an orthogonal basis for $\mathcal{K}_\ell(A, b)$, unless b belongs to an invariant subspace and therefore the norm at the denominator vanishes. This possibility is called *breakdown*, and will be further analyzed later.

We note that, by defining $h_{ij} := v_i^* w_{j+1} = v_i^* A v_j$, we obtain that the matrix V_ℓ with $v_1, \dots, v_\ell$ as columns satisfies the following relation:

$$AV_\ell = V_\ell H_\ell + h_{\ell+1,\ell} v_{\ell+1} e_\ell^*. \qquad H_\ell := \begin{bmatrix} h_{11} & \dots & \dots & h_{1,\ell} \\ h_{21} & h_{22} & & \vdots \\ & \ddots & \ddots & \vdots \\ & & h_{\ell,\ell-1} & h_{\ell,\ell} \end{bmatrix}.$$

The matrix H_ℓ is in upper Hessenberg form, since Av_j is a linear combinations of the first $j+1$ columns of V_ℓ, for any $\ell > j$. This relation is known as the *Arnoldi relation* and will be key in proving most convergence results about Krylov subspace methods.

We note that, up to adding another row to H_ℓ and make it rectangular, we can rewrite the Arnoldi relation in the more compact form:

$$AV_\ell = V_{\ell+1} \hat{H}_\ell, \qquad \hat{H}_\ell := \begin{bmatrix} H_\ell \\ h_{\ell+1,\ell} e_\ell^* \end{bmatrix}.$$

We will choose the most convenient form of the relation depending on the context.

5.3 The full-orthogonal method (FOM)

We can now formulate the *full-orthogonal method (FOM)*, that builds an approximate solution for the linear system $Ax = b$ by imposing that the solution belongs to $\mathcal{K}_\ell(A, b)$, and the residual $r_\ell^{\text{FOM}} := b - Ax_\ell^{\text{FOM}}$ is orthogonal to $\mathcal{K}_\ell(A, b)$. We shall call this solution the *FOM solution of order* ℓ to $Ax = b$, and we can write it as $x_\ell^{\text{FOM}} := V_\ell y_\ell^{\text{FOM}}$.

Theorem 5.3.1. *Let ℓ be a positive integer, and assume that $\mathcal{K}_\ell(A, b)$ has dimension ℓ and H_ℓ is invertible. Then, the FOM solution to the linear system $Ax = b$ obtained by imposing that $r_\ell^{\text{FOM}} \perp \mathcal{K}_\ell(A, b)$ exists, is unique, and is given by*

$$x_\ell^{\text{FOM}} := V_\ell y_\ell^{\text{FOM}}, \qquad y_\ell^{\text{FOM}} := \|b\|_2 \cdot H_\ell^{-1} e_1.$$

Proof. The orthogonality condition can be expressed as

$$V_\ell^*(Ax_\ell^{\text{FOM}} - b) = 0 \iff V_\ell^* A V_\ell y_\ell^{\text{FOM}} = V_\ell^* b = \|b\|_2 e_1.$$

Now, it suffices to note that, thanks to $V_\ell^* v_{\ell+1} = 0$ we have $V_\ell^* A V_\ell = H_\ell$. □

The FOM solution to the linear system is obtained by solving a much smaller linear system with $H_\ell = V_\ell^* A V_\ell$. Hence, it is very efficient to compute x_ℓ once the Krylov subspace has been built using the Arnoldi iteration.

One may wonder if the invertibility assumption on H_ℓ is actually needed, or can be automatically derived from the invertibility of A. It turns out that in the general case we can not avoid making this hypothesis, whereas under particular conditions this can be automatically derived, as the next exercises show.

Exercise 5.3.2. Find an example of an invertible matrix A for which the FOM solution is not defined (that is, H_ℓ is not invertible) for at least some ℓ.

Exercise 5.3.3. Show that when A is symmetric (or Hermitian) and positive definite[4], then FOM solutions for all $\ell \geq 1$ are well-defined for the linear system $Ax = b$.

Exercise 5.3.4. Show that the existence and uniqueness property for the FOM solution proved in the previous exercise also holds for any normal matrix A whose spectrum is enclosed in a convex region that does not contain 0. **Hint:** Try to relate the eigenvalues of A with the set $\mathcal{W}(A) = \{x^* A x \mid \|x\|_2 = 1\} \subseteq \mathbb{C}$.

We now have a characterization of the FOM solution; the next natural step is understanding how accurate it can be, and the natural way of doing this is to bound the size of $\|r_\ell^{\text{FOM}}\|_2$.

Theorem 5.3.5. *Let $x_\ell^{\text{FOM}} = V_\ell y_\ell^{\text{FOM}}$ be the FOM solution to $Ax = b$ after ℓ steps of Arnoldi with no breakdown and H_ℓ invertible. Then, the residue $r_\ell^{\text{FOM}} = b - A x_\ell^{\text{FOM}}$ satisfies*

$$\|r_\ell^{\text{FOM}}\|_2 = |h_{\ell+1,\ell}| \cdot |e_\ell^T y_\ell^{\text{FOM}}|.$$

Proof. Note that, by definition, the residual of FOM belongs to $\mathcal{K}_{\ell+1}(A, b) \cap \mathcal{K}_\ell(A, b)^\perp$. Hence, we may write

$$\|r_\ell^{\text{FOM}}\|_2 = \|v_{\ell+1}^* r_\ell^{\text{FOM}}\|_2 = \|v_{\ell+1}^* A x_\ell^{\text{FOM}}\|_2,$$

where we have exploited that $v_{\ell+1}^* b = 0$. Using $x_\ell^{\text{FOM}} = V_\ell y \ell^{\text{FOM}}$ we get

$$\|r_\ell^{\text{FOM}}\|_2 = \|v_{\ell+1}^* A V_\ell y_\ell^{\text{FOM}}\|_2 = \|h_{\ell+1,\ell} e_\ell^T y_\ell^{\text{FOM}}\|_2,$$

from which the claim follows. □

Note that this result does not guarantee the monotonicity of the residuals: it could happen that, even though we make an additional Arnoldi step to enlarge the search space, the resulting approximate solution produced by FOM is larger.

The following example shows that both "good" and "bad" behaviors are found for matrices with the same structure, depending if $|\rho| > 1$ or $|\rho| < 1$.

[4]We will later discover that in this special case FOM reduces to a well-known and much more powerful method, known as Conjugate Gradient (CG).

Exercise 5.3.6. Let A and b be the following matrix and vector, for some $\rho \neq 0$:

$$A = \begin{bmatrix} \rho & & & \\ -1 & \rho & & \\ & \ddots & \ddots & \\ & & -1 & \rho \end{bmatrix}, \qquad b = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

Prove that the FOM residual satisfy $\|r_\ell^{\text{FOM}}\|_2 = \rho^{-\ell}$ for any $\ell < n$, where n is the size of A and b, and $r_n^{\text{FOM}} = 0$.

5.4 GMRES

The fact that the FOM residual is not monotonically decreasing, and that solutions may not even exist for some ℓ, can be disturbing. We note that we can easily fix this by slightly modifying our request, and asking that the solution x_ℓ minimizes the residual norm $\|Ax_\ell - b\|_2$. This choice produces the so called GMRES method (Generalized Minimal RESidual). We can characterize the solution as follows.

Theorem 5.4.1. *Let V_ℓ, H_ℓ be the matrices obtained after ℓ steps of Arnoldi without breakdown. Then, the approximate solution $x_\ell^{\text{GMRES}} \in \mathcal{K}_\ell(A,b)$ that minimizes the residual norm $\|Ax_\ell^{\text{GMRES}} - b\|_2$ over $\mathcal{K}_\ell(A,b)$ is given by:*

$$x_\ell^{\text{GMRES}} = V_\ell y_\ell^{\text{GMRES}}, \qquad y_\ell^{\text{GMRES}} = \|b\|_2 \begin{bmatrix} H_\ell \\ h_{\ell+1,\ell} e_\ell^T \end{bmatrix}^\dagger e_1,$$

where † denotes the Moore-Penrose pseudoinverse.

Proof. Note that if x_ℓ belongs to $\mathcal{K}_\ell(A,b)$ then the residual $r_\ell = Ax_\ell - b$ belongs to $\mathcal{K}_{\ell+1}(A,b)$. Hence, by writing $x_\ell = V_\ell y_\ell$ we get

$$r_\ell = V_{\ell+1} V_{\ell+1}^* r_\ell = V_{\ell+1} \left(V_{\ell+1}^* A V_\ell y_\ell - \|b\|_2 e_1 \right),$$

where we have used that $V_\ell^* b = \|b\|_2 e_1$. Since the columns of $V_{\ell+1}$ are orthonormal, we have

$$\|r_\ell\|_2 = \|V_{\ell+1}^* A V_\ell y_\ell - \|b\|_2 e_1\|_2.$$

By a direct computation we finally get:

$$V_{\ell+1}^* A V_\ell = \begin{bmatrix} H_\ell \\ v_{\ell+1}^* A V_\ell \end{bmatrix} = \begin{bmatrix} H_\ell \\ h_{\ell+1,\ell} e_\ell^T \end{bmatrix}.$$

We conclude by recalling that the Moore-Penrose pseudoinverse computes the least-squares solution. □

Recall that $x = A^\dagger b$ corresponds to solving the least squares problem $\|Ax - b\|_2$, and in practice this is not done by explicitly computing the pseudoinverse, but instead relying on the QR factorization (assuming A is full-rank). We now want to build an algorithm

that, for any ℓ, produces a sequence of approximate solutions $x_1, x_2, \ldots, x_\ell$ of the linear system. The result (GMRES) is given in Algorithm 6.

Algorithm 6

procedure GMRES(A, b, ϵ)
  $V_1 = [v_1] \leftarrow b/\|b\|_2$
  for $j = 1, 2, \ldots$ **do**
    $w_{j+1} \leftarrow Av_j$
    $h_{1:j,j} \leftarrow V_j^* w_{j+1}$
    $w_{j+1} \leftarrow w_{j+1} - V_j h_{1:j,j}$
    $v_{j+1} \leftarrow w_{j+1}/\|w_{j+1}\|_2$
    $V_{j+1} \leftarrow \begin{bmatrix} V_j & v_{j+1} \end{bmatrix}$
    $y_{j+1} \leftarrow \|b\|_2 \begin{bmatrix} H_j \\ h_{j+1,j} e_j^T \end{bmatrix}^\dagger e_1$
    $r_{j+1} \leftarrow \begin{bmatrix} H_j \\ h_{j+1,j} e_j^T \end{bmatrix} y_{j+1} - \|b\|_2 e_1$
    if $\|r_{j+1}\|_2 < \epsilon$ **then**
      return $x_{j+1} := V_{j+1} y_{j+1}$
    end if
  end for
end procedure

In GMRES, the residual of the linear system is immediately available by

$$\|r_\ell^{\text{GMRES}}\|_2 = \|Ax_\ell^{\text{GMRES}} - b\|_2 = \left\| \begin{bmatrix} H_\ell \\ h_{\ell+1,\ell} e_\ell^T \end{bmatrix} y_\ell^{\text{GMRES}} - \|b\|_2 e_1 \right\|_2 .$$

For this reason, there is no need for an analogue of Theorem 5.3.5; nevertheless, we will see in the next section that the GMRES solution can be computed much more efficiently than what appears from Algorithm 6, and the residual is immediately available even without explicitly finding y_ℓ^{GMRES}.

5.5 Solving the least squares GMRES problem

GMRES solves a least squares problem of the following form at each step

$$\min_{y_\ell \in \mathbb{C}^\ell} \left\| \begin{bmatrix} h_{11} & \ldots & h_{1\ell} \\ h_{21} & \ldots & h_{2\ell} \\ & \ddots & \vdots \\ & & h_{\ell+1,\ell} \end{bmatrix} \begin{bmatrix} [y_\ell]_1 \\ \vdots \\ [y_\ell]_\ell \end{bmatrix} - \begin{bmatrix} \|b\|_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \right\|_2$$

By denoting with $\hat{H}_\ell$ the rectangular upper Hessenberg matrix of size $(\ell+1) \times \ell$ above, this may be solved by computing a QR factorization $\hat{H}_\ell = QR$, and obtaining the

equivalent linear system:

$$\min_{y_\ell \in \mathbb{C}^\ell} \left\| \begin{bmatrix} r_{11} & \dots & r_{1\ell} \\ & \ddots & \vdots \\ & & r_{\ell,\ell} \\ & & 0 \end{bmatrix} \begin{bmatrix} [y_\ell]_1 \\ \vdots \\ [y_\ell]_\ell \end{bmatrix} - \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_{\ell+1} \end{bmatrix} \right\|_2, \qquad \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_{\ell+1} \end{bmatrix} = Q^* \begin{bmatrix} \|b\|_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

The solution and the residual of the GMRES step are then available as

$$y_\ell = \begin{bmatrix} r_{11} & \dots & r_{1\ell} \\ & \ddots & \vdots \\ & & r_{\ell,\ell} \end{bmatrix}^{-1} \begin{bmatrix} \gamma_1 \\ \vdots \\ \gamma_\ell \end{bmatrix}, \qquad \|r_\ell^{\text{GMRES}}\|_2 = \|\hat{H}_\ell y_\ell - \|b\|_2 e_1\|_2 = |\gamma_{\ell+1}|. \tag{5.1}$$

Generically, a QR factorization of a $(\ell+1) \times \ell$ matrix requires $\mathcal{O}(\ell^3)$ flops. This would be a negligible cost in the first GMRES step, when $\ell \ll n$. However, when the dimension of the Krylov subspace grows, it may become relevant. Hence, we make the following observations:

- The matrix $\hat{H}_\ell$ is upper Hessenberg, and therefore relatively close to upper triangular; this allows us to compute a QR factorization using Givens rotations at a quadratic cost (we can use the same trick in the QR iteration for eigenvalues).
- The two matrices $\hat{H}_\ell$ and $\hat{H}_{\ell+1}$ are fairly close: if we know the QR factorization of $\hat{H}_\ell$ we can obtain the one of $\hat{H}_{\ell+1}$ at an even lower cost.

We proved the first item while dealing with the upper Hessenberg form in the QR iteration, and more precisely in Lemma 2.10.5. In the current notation, it implies the existence of a factorization

$$G_1 \dots G_\ell \begin{bmatrix} r_{11} & \dots & r_{1\ell} \\ & \ddots & \vdots \\ & & r_{\ell\ell} \\ 0 & \dots & 0 \end{bmatrix} = \hat{H}_\ell, \qquad G_i = I_{i-1} \oplus \begin{bmatrix} c_i & \overline{s_i} \\ -s_i & \overline{c_i} \end{bmatrix} \oplus I_{\ell-i-1}.$$

This factorization allows us to rewrite the solution of the least squares problem in a more explicit way:

$$y_\ell = \begin{bmatrix} R_\ell^{-1} & 0_{\ell\times 1} \end{bmatrix} G_\ell^* \dots G_1^* \|b\|_2 e_1 = \|b\|_2 R_\ell^{-1} \begin{bmatrix} c_1 \\ c_2 s_1 \\ c_3 s_2 s_1 \\ \dots \\ c_\ell s_{\ell-1} \dots s_1 \\ s_\ell s_{\ell-1} \dots s_1 \end{bmatrix}.$$

In addition, using Equation (5.1) we can derive an explicit formula for the residual imposing $x_\ell = V_\ell y_\ell$:

$$\|Ax_\ell - b\|_2 = \|e_{\ell+1}^T G_\ell^* \dots G_1^* \|b\|_2 e_1\|_2 = |s_\ell s_{\ell-1} \dots s_1| \cdot \|b\|_2. \tag{5.2}$$

Remark 5.5.1. The expression of the residual from (5.2) does not require to compute y_ℓ. It is then possible to carry out the GMRES iteration without explicitly solving the least squares problem, but just by finding R_ℓ, unless the residual is small enough. Then, the actual solution is computed only at last step.

5.6 Convergence

GMRES produces a sequence of decreasing residuals (in norm), which is already a remarkable property. However, if the convergence to zero is slow, we may need a large number of iterations to actually find an approximate solution of $Ax = b$.

This section is dedicated to characterizing the convergence speed, in terms of bounding the norm of the residuals $r_\ell^{\text{GMRES}} = b - Ax_\ell^{\text{GMRES}}$. We will link the convergence speed to some spectral properties of A, and this will guide us in transforming the problem to accelerate convergence.

Theorem 5.6.1. *Let x_ℓ be the sequence generated by GMRES for the linear system $Ax = b$, without breakdown. Then, $r_\ell = Ax_\ell - b$ satisfies*

$$\|r_\ell\|_2 = \|Ax_\ell - b\|_2 = \min_{\substack{p(x)\in\mathcal{P}_\ell \\ p(0)=1}} \|p(A)b\|_2,$$

where $\mathcal{P}_\ell$ è is the set of polynomial of degree at most ℓ.

Proof. Recall that x_ℓ is chosen to minimize the residuals $\|Ax - b\|_2$ among all $x \in \mathcal{K}_\ell(A, b)$ and, by definition of Krylov subspace, we may write $x_\ell = q(A)b$ with $q(x)$ polynomial of degree at most $\ell - 1$. We then obtain

$$-r_\ell = b - Ax_\ell = b - Aq(A)b = (I - Aq(A))b = p(A)b,$$

where $p(x) = 1 - xq(x)$, a generic polynomial of degree ℓ with $p(0) = 1$. The claim follows by the minimizing property of GMRES. □

An expression of the form $p(A)b$ can be linked, for normal matrices, to the value of $p(x)$ over the spectrum of A. The same holds for diagonalizable matrices, although a constant $\kappa_2(V)$ appears in the bound.

Corollary 5.6.2. *Let A be diagonalizable with eigenvector matrix V; then, GMRES produces a sequence of residuals $\{r_\ell\}_{\ell\geq 1}$ satisfying*

$$\|r_\ell\|_2 \leq \kappa_2(V) \cdot \min_{\substack{p(x)\in\mathcal{P}_\ell \\ p(0)=1}} \max_{\lambda\in\Lambda(A)} |p(\lambda)| \cdot \|b\|_2.$$

Proof. The proof follows by diagonalizing A, showing that $p(A)b = Vp(D)V^{-1}b$, then using Theorem 5.6.1 and computing the spectral norm. □

Exercise 5.6.3. Find an upper bound for the convergence of GMRES for a normal matrix A with eigenvalues enclosed in $B(1, \rho) = \{|z - 1| \leq \rho\}$ for some $\rho < 1$.

These results show that the optimal situation for GMRES convergence is having (at least for normal matrices) the eigenvalues clustered around $\lambda = 1$ (or any other non-zero value, since the problem is scale invariant). This is clearly not always the case, and is the reason preconditioning techniques have been developed to modify linear systems to fall back into this case.

5.7 Spectral sets

Bounding the size of $\|p(A)b\|_2$ when A is a non-normal matrix can be challenging: the result from Corollary 5.6.2 can be rather loose when the condition number $\kappa_2(V) \gg 1$.

In this particular case, one solution is to introduce sets larger than the spectrum where to measure the effect of the polynomial, and reduce the constant. This idea leads to the definition of K-spectral sets.

Definition 5.7.1. A set $\mathcal{S}$ is a K-spectral set for a matrix A and a given norm $\|\cdot\|$ if, for any analytic function $f(z)$ over $\mathcal{S}$ it holds

$$\|f(A)\| \leq K \cdot \max_{z \in \mathcal{S}} |f(z)|.$$

Let us make a few examples:

- Whenever A is normal, then the spectrum of A is a 1-spectral set for the spectral norm.
- If A is diagonalizable, then the spectrum is a $\kappa_2(V)$-spectral set, again for the spectral norm.

Finding more general spectral sets that work well with non-normal matrices is not an easy task. We report the following theorem, whose proof requires advanced tools (and some effort), and is therefore omitted from these notes.

Theorem 5.7.2 (Crouzeix-Palencia)**.** *Let A be a matrix, and $\mathcal{W}(A)$ its field of values, defined as*

$$\mathcal{W}(A) := \{x^* A x \mid \|x\|_2 = 1\}.$$

Then, $\mathcal{W}(A)$ is a $(1 + \sqrt{2})$-spectral set for A with the spectral norm.

The key observation is that now the constant does not depend on the matrix under consideration, whereas the set does. Clearly, this result can be used to limit the size of $\|p(A)b\|_2$ on specific examples.

There are a few limitations to this approach:

1. Computing explicitly $\mathcal{W}(A)$ is not an easy task. A few properties can be proven (the set is convex and contains the spectrum), but the set is difficult to handle both theoretically and numerically.
2. There is no guarantee that $0 \notin \mathcal{W}(A)$ even when A is invertible. Whenever this happens, all upper bounds for $\|p(A)b\|_2$ will be equal to 1 (why?), and therefore are useless to understand the convergence of GMRES.

5.8 GMRES preconditioning

The previous results on the convergence of GMRES tell us that, when the spectrum of A has eigenvalues that are not clustered away from 0, we may expect a slow convergence. The idea of preconditioning is to solve this issues by modifying the original problem using the degrees of freedoms that we have at our disposal. Indeed, note that for any invertible matrices M_1, M_2 we have

$$Ax = b \iff M_1^{-1} A M_2^{-1} M_2 x = M_1^{-1} b,$$

and by setting $y := M_2 x$ we can solve the linear system $\tilde{A} y = \tilde{b}$ with $\tilde{A} := M_1^{-1} A M_2^{-1}$ and $\tilde{b} = M_1^{-1} b$, and only the recovering $x = M_2^{-1} y$. If we choose M_1 and M_2 wisely we may end up with a problem with much better convergence properties of the original one.

We remark that it is not necessary to explicitly compute $M_1^{-1} A M_2^{-1}$, but instead to efficiently implement the action of this matrix on a vector:

$$v \mapsto M_1^{-1} A M_2^{-1} v = M_1^{-1}(A(M_2^{-1} v)).$$

With a smart ordering of the arithmetic operations, this is equivalent to a matrix multiplication with A, and two linear systems with M_1 and M_2. To summarize, our aim is to select M_1, M_2 in a way that:

- $M_1^{-1} A M_2^{-1}$ has good convergence properties with GMRES; in practice, this often means that the preconditioned matrix is well conditioned, or a low-rank perturbation of a well-conditioned matrix.
- The linear systems with M_1 and M_2 can be solved efficiently.

A well-chosen preconditioner will make GMRES converge in less iterations, but at a higher cost per iteration. Finding the right balance is critical to an efficient implementation.

Often, M_1 is called *left preconditioner*, whereas M_2 is called a *right preconditioner*. We will now simplify the notation assuming that $M_1 = M$ and $M_2 = I$ (i.e., we only apply the left preconditioner). Most considerations that we make will be easy to transfer to the other case.

Note that using a right preconditioner does not change the residual, whereas using a left preconditioner does not need to recover the solution at the end.

5.8.1 Diagonal preconditioners and splitting methods

The easiest preconditioner is obtained by taking $M = D$, where D is the diagonal of A. Whenever A is diagonally dominant, we may expect the diagonal to "tell most of the story", and therefore this to be a good choice. This choice is sometimes called a *Jacobi preconditioner*, and has a link with splitting methods. We briefly recall that, given an additive partitioning $A = M - N$ with $\det M \neq 0$, we have

$$Ax = b \iff Mx = Nx + b \iff x = M^{-1} N x + M^{-1} b \iff x = Px + q,$$

where $P = M^{-1}N$ and $q = M^{-1}b$. This suggests to set up the fixed point iteration $x^{(k+1)} = Px^{(k)} + q$, which gives the so-called *splitting methods*, and is globally convergent whenever $\rho(P) < 1$. In this context, the Jacobi method is obtained by choosing M as the diagonal of A, and Gauss-Seidel by taking M to be the upper or lower triangular part.

All choices of that provide a convergence splitting (i.e., for which $\rho(P) < 1$) work as a good preconditioner, as the next results shows.

Lemma 5.8.1. *Let $A = M - N$ an additive splitting of A with $\det M \neq 0$ and $\rho(M^{-1}N) = \rho < 1$. Then, the $M^{-1}A$ has eigenvalues in $B(1, \rho)$ and the GMRES method for $Ax = b$ preconditioned with M satisfies $r_\ell \sim \mathcal{O}(\rho^\ell)$.*

Proof. We write

$$M^{-1}A = M^{-1}(M - N) = I - M^{-1}N.$$

Hence, $M^{-1}A$ has $1 - \lambda$ as eigenvalues, where $\lambda \in \Lambda(M^{-1}N) \subseteq B(0, \rho)$. □

In practice, the preconditioner may be better than what is predicted by Lemma 5.8.1, since GMRES can "deflate" a few eigenvalues after a few steps. Hence, if the spectrum of $M^{-1}A$ has a few eigenvalues of the form $1 - \lambda$ with $|\lambda| \approx \rho$, and the rest much closer to 1, we can expect an acceleration of the convergence speed after a few steps. This phenomenon is known as the *superlinear convegence* of Krylov methods, and is never found in fixed point iterations.

5.8.2 Sparse approximate inverse

The preconditioner should ideally approximate $M^{-1} \approx A^{-1}$ as well as possible, while being still easy to apply. A class of effective choices for sparse matrices A is trying to find M such that

$$M = \arg \min_{M^{-1} \in \mathcal{S}} \|I - AM^{-1}\|_F,$$

where $\mathcal{S}$ is the class of a matrices with a specific structure. A common choice is to take $\mathcal{S}$ the set of matrices with the same sparsity structure of A. The choice of the Frobenius norm here is not by chance, since we have

$$\|I - AM^{-1}\|_F^2 = \sum_{j=1}^{n} \|e_j - AM^{-1}e_j\|_2^2.$$

Hence, we can determine the columns of M^{-1} independently by solving a constrained least-square problem. In particular, let us fix the index k, and denote that index sets R_j and C_j as the rows that are non-zero in Ae_j and the rows that can be non-zero in $M^{-1}e_j$, respectively. Then, $M^{-1}e_j$ can be determined by solving the least-square problem

$$\min \|A(R_j, C_j)w_j - e_j(I_j)\|_2,$$

and then by setting $M^{-1}e_j$ equal to w_j in the rows C_j, and zero-elsewhere. In practice, this is a very small least square problem, which is cheap to solve.

5.9 Symmetric problems: Lanczos and the conjugate gradient

In the real symmetric case, when $A = A^T \in \mathbb{R}^{n\times n}$, the Arnoldi procedure is considerably simplified. The GMRES and FOM methods are usually given a specific name: MINRES, and Conjugate Gradient (CG)[5].

5.9.1 Lanczos iteration and MINRES

By rewriting the Arnoldi iteration when $A = A^T$ we observe that $H_\ell = H_\ell^T$, and therefore it is both upper and lower Hessenberg. In other words, H_ℓ is *tridiagonal*. For this reason, we refer to the matrix as T_ℓ, and the Arnoldi relation (that takes the name of Lanczos relation) takes the form

$$AV_\ell = V_\ell T_\ell + \beta_\ell v_{\ell+1} e_\ell^T, \qquad T_\ell = \begin{bmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_{\ell-1} \\ & & \beta_{\ell-1} & \alpha_\ell \end{bmatrix}.$$

Since the upper triangular part of T_ℓ is mostly zero, most of the scalar products required to carry on the Arnoldi iteration are known a priori, and are zero. This means that Av_ℓ is already orthogonal to $v_1, \dots, v_{\ell-2}$ by construction, without the need of a full reorthogonalization procedure.

Using this observation, we may reformulate the iteration as follows:

$$v_1 = b/\beta_0, \qquad \beta_0 = \|b\|_2, \qquad \beta_\ell v_{\ell+1} = Av_\ell - \alpha_\ell v_\ell - \beta_{\ell-1} v_{\ell-1}.$$

We remark that:

- After ℓ steps, the term $\beta_{\ell-1}$ is known from the previous computations. To make the first step well-defined, we set $v_0 = 0$.
- α_ℓ is computed as $\alpha_\ell = v_\ell^T A v_\ell$, or alternatively as $\alpha_\ell = v_\ell^T (Av_\ell - \beta_{\ell-1} v_{\ell-1})$. The two forms are equivalent because $v_\ell^T v_\ell = 1$, but the latter can be preferred for stability reasons.
- β_ℓ is determined by computing $\beta_\ell := \|Av_\ell - \alpha_\ell v_\ell - \beta_{\ell-1} v_{\ell-1}\|_2$.

Once T_ℓ is known, it is possible to proceed by solving the linear system by projecting and minimizing the residual, as in the GMRES case. This gives the MINRES method. The cost is reduced because of the special structure of T_ℓ, and this is the method of choiced for symmetric but indefinite problems.

However, when A is symmetric and positive definite (SPD), the same property holds for T_ℓ, and this guarantees the applicability of the FOM method, which in this context is called Conjugate Gradient (CG). It turns out that this method is particularly appealing in the SPD case, and deserves to be ananlyzed in detail.

[5]Historically, the theory for the symmetric case has been developed before the unsymmetric ones, and that's the reason for the G in GMRES, which stands for Generalized.

5.9.2 The Conjugate Gradient method

From now on, we assume that A is SPD. Recall that FOM computes at every step the solution of the linear system $T_\ell y_\ell = \|b\|_2 e_1$, and then this is used to define the solution vector $x_\ell := V_\ell y_\ell$.

Since A is SPD, the same holds for T_ℓ for any ℓ, thanks to the Courant-Fischer theorem. Indeed, the eigenvalues of T_ℓ are enclosed in $[\lambda_{\min}(A), \lambda_{\max}(A)]$. Making use of these observations, we have the following explicit expression for the solution vector at step ℓ:

$$x_\ell = V_\ell T_\ell^{-1} \begin{bmatrix} \|b\|_2 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

We now show how we can rephrase the solution linking x_ℓ and $x_{\ell-1}$. This will trasform the CG method into a method that, at each step, directly updates the current solution vector.

Consider an LU factorization $T_\ell = L_\ell U_\ell$ without pivoting. This exists and can be stably computed thanks to the fact that T_ℓ is SPD. Then,

$$T_\ell = L_\ell U_\ell = \begin{bmatrix} 1 & & & \\ \lambda_1 & 1 & & \\ & \ddots & \ddots & \\ & & \lambda_{\ell-1} & 1 \end{bmatrix} \begin{bmatrix} \eta_1 & \beta_1 & & \\ & \ddots & \ddots & \\ & & \ddots & \beta_{\ell-1} \\ & & & \eta_\ell \end{bmatrix},$$

where β_i are exactly the super and sub-diagonal entries of T_ℓ. By reading the above matrix relation we have

$$\lambda_j = \beta_j/\eta_j, \qquad \eta_{j+1} = \alpha_{j+1} - \lambda_j \beta_j, \qquad \eta_1 = \alpha_1.$$

Hence, the LU factorization may be computed during the Lanczos iteration by using these recurrence relations, instead of recomputing it from scratch at every step. We remark that x_ℓ can be rewritten in a different form as

$$x_\ell = \underbrace{V_\ell U_\ell^{-1}}_{P_\ell} L_\ell^{-1} e_1 = \begin{bmatrix} p_1 & \dots & p_\ell \end{bmatrix} \begin{bmatrix} z_1 \\ \vdots \\ z_\ell \end{bmatrix}.$$

The columns of P_ℓ are well defined because U_ℓ and therefore also U_ℓ^{-1} are upper triangular. This allows us to write x_ℓ as an update of $x_{\ell-1}$ as

$$x_\ell = x_{\ell-1} + z_\ell p_\ell. \tag{5.3}$$

From the relation $P_\ell U_\ell = V_\ell$ we may derive the identity

$$\eta_\ell p_\ell = v_\ell - \beta_{\ell-1} p_{\ell-1}.$$

We now want to exploit (5.3) to update the solution vector x_ℓ at each step. To achieve this, we need relations to obtain z_ℓ and p_ℓ from the previous iterations. We will reach this goal by appropriately characterizing some orthogonality properties of v_ℓ and p_ℓ. For the former vectors, we already know that they are orthogonal with respect to the canonical scalar product. The latter satisfy an orthogonality property guaranteed by the following result.

Theorem 5.9.1. *The vectors $p_1, \ldots, p_\ell$ are A-orthogonal, and therefore $P_\ell^T A P_\ell$ is a diagonal matrix.*

Proof. Note that we may rewrite $P_\ell^T A P_\ell$ as follows:

$$P_\ell^T A P_\ell = U_\ell^{-T} V_\ell^T A V_\ell U_\ell^{-1} = U_\ell^{-T} T_\ell U_\ell^{-1} = U_\ell^{-T} L_\ell.$$

The above matrix is lower triangular, and at the same time symmetric, and therefore is diagonal. □

Let us summarize the information on the CG method:

- The residuals r_ℓ are orthogonal (the method is the symmetric equivalent of FOM).
- The vectors p_ℓ are A-orthogonal, thanks to Theorem 5.9.1

We shall now describe a sequence of steps that allow to derive x_ℓ. We now make use of the notation $r_\ell = b - Ax_\ell$ for the residual, since this sign choice is the most convenient. We have

$$x_\ell = x_{\ell-1} + z_\ell p_\ell \implies r_\ell = r_{\ell-1} - z_\ell A p_\ell$$

By imposing the orthogonality condition $r_\ell^T r_{\ell-1} = 0$ we get:

$$z_\ell = \frac{r_{\ell-1}^T r_{\ell-1}}{r_{\ell-1}^T A p_\ell},$$

that allows us to compute x_ℓ and r_ℓ from $r_{\ell-1}$. The definition of p_ℓ implies that v_ℓ is a linear combination of p_ℓ e $p_{\ell-1}$, and therefore the same holds for $r_{\ell-1}$ (thanks to the properties of FOM). Up to an appropriate rescaling of the vectors p_j, we obtain

$$p_\ell = r_{\ell-1} + \xi_{\ell-1} p_{\ell-1}, \tag{5.4}$$

and we may rewrite the relation for z_ℓ as follows

$$z_\ell = \frac{r_{\ell-1}^T r_{\ell-1}}{p_\ell^T A p_\ell}, \tag{5.5}$$

where we have exploited the A-orthogonality of the p_j. We note that, formally, this is just a rescaled version of the previous relation, with different z_js. We now need to

determine the vectors p_ℓ, by finding ξ_ℓ. To this aim, we use once more Equation (5.4) for $\ell+1$, imposing the A-orthogonality with p_ℓ, that yields the following

$$\xi_\ell = -\frac{p_\ell^T A r_\ell}{p_\ell^T A p_\ell} = \frac{1}{z_\ell}\frac{r_\ell^T r_\ell}{p_\ell^T A p_\ell} = \frac{r_\ell^T r_\ell}{r_{\ell-1}^T r_{\ell-1}},$$

where we have used the definition of z_ℓ and the relation $Ap_\ell = \frac{r_{\ell-1}-r_\ell}{z_\ell}$. Combining these relations, we obtain the classical formulation of the conjugate gradient, reported in Algorithm 7. This is extremely simple to implement, and has excellent numerical properties.

Algorithm 7

procedure CG(A, b, ϵ)
  $x_0 \leftarrow 0$
  $r_0 \leftarrow b$, $p_1 \leftarrow b/\|b\|_2$
  for $\ell = 1, \dots, k$ **do**
    $z_\ell \leftarrow \frac{r_{\ell-1}^T r_{\ell-1}}{p_\ell^T A p_\ell}$
    $x_\ell \leftarrow x_{\ell-1} + z_\ell p_\ell$
    $r_\ell \leftarrow r_{\ell-1} - z_\ell A p_\ell$
    $\xi_\ell \leftarrow \frac{r_\ell^T r_\ell}{r_{\ell-1}^T r_{\ell-1}}$
    $p_{\ell+1} = r_\ell + \xi_\ell p_\ell$
    if $\|r_\ell\|_2 \le \epsilon$ **then**
      return x_ℓ
    end if
  end for
end procedure

5.9.3 CG as an optimization method

The conjugate gradient method can be interpreted as an optimization method applied to the quadratic objective function

$$\Phi(x) = \frac{1}{2}x^T A x - x^T b, \qquad x \in \mathbb{R}^n.$$

If A is positive definite, then the function $\Phi(x)$ is convex and has a unique global minimum, which corresponds to the critical point where the gradient $\nabla\Phi(x) = x^T A - b^T$ vanishes, which is the solution to the linear system.

Since A is SPD, we may define a norm induced by the scalar product associated with A: $\|x\|_A := \sqrt{x^T A x}$. The conjugate gradient iteration has the property of minimizing the error with respect to this norm at each step.

Theorem 5.9.2. *The approximate solution x_ℓ given by the conjugate gradient method at step ℓ minimizes the error with respect to the norm $\|\cdot\|_A$ over $K_\ell(A, b)$, that is*

$$x_\ell = \arg\min_{\tilde{x}\in\mathcal{K}_\ell(A,b)} \|\tilde{x} - x\|_A.$$

Proof. For this condition to be verified, it is sufficient to verify that the gradient of $\Psi(\tilde{x}) = (\tilde{x}-x)^T A(\tilde{x}-x)$ is orthogonal to the search space $\mathcal{K}_\ell(A,b)$, where x is the exact solution of the linear system. We may write:

$$\nabla\Psi(\tilde{x}) = 2(\tilde{x}-x)^T A = 2(\tilde{x}^T A - b^T),$$

which is orthogonal to $\mathcal{K}_\ell(A,b)$ if and only if $\tilde{x} = x_\ell$, thanks to the residual orthogonality property of CG. □

The previous result allows us to state an analogue of Theorem 5.6.1, but in terms of the error and not the residual, and with respect to $\|\cdot\|_A$. We can still obtain a characterization of the residual, but with respect of $\|\cdot\|_{A^{-1}}$.

Theorem 5.9.3. *The solution x_ℓ given by step ℓ of CG satisfies*

$$\|x - x_\ell\|_A = \|b - Ax_\ell\|_{A^{-1}} = \min_{\substack{p\in P_\ell \\ p(0)=1}} \|p(A)b\|_{A^{-1}}.$$

Proof. Let us note that the A-norm of the error at step ℓ can be written as follows:

$$\begin{aligned}\|x - x_\ell\|_A^2 &= (A^{-1}b - q(A)b)^T A(A^{-1}b - q(A)b) \\ &= b^T(I - Aq(A))A^{-1}(I - Aq(A))b = \|p(A)b\|_{A^{-1}}^2,\end{aligned}$$

where we have used $x_\ell = q(A)b$ and $p(x) = 1 - xq(x)$. With an argument analogue to Theorem 5.6.1 we conclude that CG minimizes the norm $\|\cdot\|_{A^{-1}}$ of the residual with respect to all polynomials of degree ℓ with constant term equal to 1. It is readily verified that it holds

$$\|Ax_\ell - b\|_{A^{-1}} = \|x_\ell - x\|_A,$$

which gives also the second part of the thesis. □

The derivation of CG can be presented in a different way by building a gradient descent method that optimizes $\Phi(x)$, and then imposing the A-orthogonality of the descent directions p_ℓ. This has been the historical derivation of the method, and the reason why we call p_ℓ descent directions, and z_ℓ step-length. However, the presentation as a Krylov method allows for much easier proofs of convergence.

Remark 5.9.4. It is useful to remark once more the advantage in the symmetric setting: the solution of the linear system does not require to store the entire basis V_ℓ, thanks for the short recurrence relation. This can make an enormous different for large problems, in order to avoid memory issues. Nonsymmetric problems of similar sizes can only be handled with restart techniques when using GMRES, which may degrade the convergence of the method.

5.9.4 Characterization of the convergence

Understanding the convergence of GMRES in an explicit manner, instead of the representation $\|r_\ell\|_2 = \|p(A)b\|_2$ given by Theorem 5.6.1 is not trivial, in particular for matrices far from normality.

For the conjugate gradient that operates on symmetric matrices allows to explicitly state the rate of convergence in terms of the condition number of A. We report the following result without proof, for which we refer to [6, Cap. 6.11.3].

Theorem 5.9.5. *Let A be a real SPD matrix, and x_ℓ the sequence of approximate solutions built by CG for $Ax = b$. Then, it holds*

$$\|x - x_\ell\|_A \leq 2 \left(\frac{\sqrt{\kappa_2(A)} - 1}{\sqrt{\kappa_2(A)} + 1} \right)^\ell \|x\|_A.$$

5.9.5 Preconditioning in the symmetric case

Similarly to GMRES, preconditioning is often essential to make CG practical. If the original problem has a large condition number, then the standard iteration is bound to converge slowly, as per Theorem 5.9.5. However, even assuming a good preconditioner M such that $M^{-1}A \approx I$ is available, it cannot be applied as it is, since it would lead to the loss of the symmetric structure.

We shall now make the reasonable assumption that the chosen preconditioner M is symmetric and positive definite. Then, we can find a matrix L such that $M = LL^T$, which is its Cholesky factorization.

Exercise 5.9.6. Show that the standard procedure for computing the LU factorization can be slightly tuned to produce a Cholesky factorization if M is SPD, and that this factorization is unique if we require the diagonal entries to be positive.

Instead of applying M as a left or right preconditioner, we can make use of its Cholesky factor to split it in a symmetric way, and consider the equivalent linear system

$$Ax = b \iff L^{-1}AL^{-T}L^T x = L^{-1}b.$$

The matrix $L^{-1}AL^{-T}$ is the preconditioned matrix, and is still symmetric and positive definite. If $M^{-1}A \approx I$, the same holds for $L^{-1}AL^{-T}$.

In principle, this approach works exactly as the standard preconditioning for GMRES. However, we make the following observation. Even though $M^{-1}A$ is not symmetric, it is self-adjoint with respect to a non-standard scalar product, the one induced by M. In fact, we have for any vector v

$$\langle M^{-1}A, v\rangle_M = (M^{-1}A)^T Mv = AM^{-T}Mv = v^T A = v^T MM^{-1}A = \langle v, M^{-1}A\rangle_M.$$

We shall now use this idea to avoid explicitly involving the Cholesky factor —which would be often expensive to compute—, and end up working with just the matrix M.

We now rewrite the CG for the matrix $L^{-1}AL^{-T}$, with initial vector $L^{-1}b$. We have the following updates, according to Algorithm 7:

$$\begin{aligned} z_\ell &\leftarrow (r_{\ell-1}^T r_{\ell-1})/(p_\ell^T L^{-1} A L^{-T} p_\ell) \\ x_\ell &\leftarrow x_{\ell-1} + z_\ell p_\ell \\ r_\ell &\leftarrow r_{\ell-1} - z_\ell L^{-1} A L^{-T} p_\ell \\ \xi_\ell &\leftarrow (r_\ell^T r_\ell)/(r_{\ell-1}^T r_{\ell-1}) \\ p_{\ell+1} &\leftarrow r_\ell + \xi_\ell p_\ell \end{aligned}$$

This modified method would not compute the solution x, but instead $L^T x$. Hence, it is reasonable to pre-multiply the x_ℓ by L^{-T}, to make sure that the final limit is the sought solution. By setting $\tilde{x}_\ell := L^{-T} x_\ell$, $\tilde{p}_\ell := L^{-T} p_\ell$, we get

$$\begin{aligned} z_\ell &\leftarrow (r_{\ell-1}^T r_{\ell-1})/(\tilde{p}_\ell^T A \tilde{p}_\ell) \\ \tilde{x}_\ell &\leftarrow \tilde{x}_{\ell-1} + z_\ell \tilde{p}_\ell \\ r_\ell &\leftarrow r_{\ell-1} - z_\ell L^{-1} A \tilde{p}_\ell \\ \xi_\ell &\leftarrow (r_\ell^T r_\ell)/(r_{\ell-1}^T r_{\ell-1}) \\ \tilde{p}_{\ell+1} &\leftarrow L^{-T} r_\ell + \xi_\ell \tilde{p}_\ell \end{aligned}$$

We still have two appearances of L, which we can remove by defining a preconditioned residual $\tilde{r}_\ell := L^{-T} r_\ell$, which finally gives equations involving only A and M:

$$\begin{aligned} z_\ell &\leftarrow (\tilde{r}_{\ell-1}^T M \tilde{r}_{\ell-1})/(\tilde{p}_\ell^T A \tilde{p}_\ell) \\ \tilde{x}_\ell &\leftarrow \tilde{x}_{\ell-1} + z_\ell \tilde{p}_\ell \\ \tilde{r}_\ell &\leftarrow \tilde{r}_{\ell-1} - z_\ell M^{-1} A \tilde{p}_\ell \\ \xi_\ell &\leftarrow (\tilde{r}_\ell^T M \tilde{r}_\ell)/(\tilde{r}_{\ell-1}^T M \tilde{r}_{\ell-1}) \\ \tilde{p}_{\ell+1} &\leftarrow \tilde{r}_\ell + \xi_\ell \tilde{p}_\ell \end{aligned}$$

This idea can be used (with a few variants, see [6]) to implement the preconditioned CG without the need to explicitly determine L. The iteration can indeed by interpreted as a CG iteration with a non-standard scalar product, the one induced by M.

5.10 Computing eigenvalues and eigenvectors with Arnoldi

The Krylov methods considered in the previous sections allow to solve large scale linear systems; however, they can be adapted to computing selected eigenvalues for large matrices.

Clearly, whenever A is large, there is no hope of computing all the eigenvalues at a low cost (in general). As in the setting for the solution of linear system, we assume to know A implicitly by its action as a matrix-vector product. Then, we can consider the following prototype of algorithm

- For increasing ℓ, we perform the Arnoldi iteration and build a sequence of upper Hessenberg matrices $H_1, H_2, \dots, H_\ell$.
- For each of these, we compute eigenvalues and eigenvectors.

The eigenvalue of H_ℓ are called *Ritz values*, and the eigenvectors *Ritz vectors*. We claim that such eigenvalues are approximation of (some) eigenvalues of the large matrix A. Indeed, it turns out that H_ℓ solves a minimization problem that resembles the one for GMRES or CG.

Before giving the formal result, in Theorem 5.10.2, we prove the following Lemma.

Lemma 5.10.1. *Let $q(z)$ be any polynomial of degree at most ℓ, and $AV_\ell = V_\ell H_\ell + h_{\ell+1,\ell} v_{\ell+1} e_\ell^T$ the Arnoldi relation without breakdown with initial vector b. Then, we have*

$$q(A)b = \begin{cases} V_\ell q(H_\ell) e_1 \|b\|_2 & \text{if } k < \ell \\ V_\ell q(H_\ell) e_1 \|b\|_2 + q_\ell h_{\ell+1,\ell} v_{\ell+1} e_\ell^T H_\ell^{\ell-1} e_1 \|b\|_2 & \text{if } k = \ell. \end{cases}$$

Proof. Using the linearity of $q(z)$ as a sum of monomial, we can reduce the statement to proving that, for each $k < \ell$, the following identity holds:

$$A^k b = V_\ell H_\ell^k e_1 \|b\|_2.$$

By repeatedly using the Arnoldi relation to $A^k b = A^k V_\ell e_1 \|b\|_2$ we obtain

$$\begin{aligned} A^k b = V_\ell H_\ell^k e_1 \|b\|_2 &= \left(A^{k-1} V_\ell H_\ell + h_{\ell+1,\ell} A^{k-1} v_{\ell+1} e_\ell^T \right) e_1 \|b\|_2 \\ &= \left(A^{k-2} V_\ell H_\ell^2 + h_{\ell+1,\ell} A^{k-2} v_\ell e_\ell^T H_\ell + h_{\ell+1,\ell} A^{k-1} v_{\ell+1} e_\ell^T \right) e_1 \|b\|_2 \\ &\vdots \\ &= \left(V_\ell H_\ell^k + h_{\ell+1,\ell} \sum_{i=0}^{k-1} A^i v_{\ell+1} e_\ell H_\ell^{k-i-1} \right) e_1 \|b\|_2. \end{aligned}$$

We now note that $e_\ell^T H_\ell^{k-i-1}$ has at most the last k entries different from zero, and since we assumed that $k < \ell$, this implies that $e_\ell H_\ell^{k-i-1} e_1 = 0$ for all $i = 0, \dots, k-1$.

Hence, the result follows by expanding the last inequality. □

Theorem 5.10.2. *Let H_ℓ be the upper Hessenberg matrix from the Arnoldi process for A without breakdown, starting from an initial vector b. Then, the characteristic polynomial $p(\lambda) = \det(\lambda I - H_\ell)$ satisfies the following minimization problem:*

$$\min_{\substack{p \in \mathcal{P}_\ell \\ p_\ell = 1}} \|p(A)b\|_2.$$

Proof. The minimum problem can be rewritten in constrained least squares form:

$$\min_{\substack{p \in \mathcal{P}_\ell \\ p_\ell = 1}} \|p(A)b\|_2 = \min_{y \in \mathbb{R}^\ell} \|A^\ell b - V_\ell y\|_2.$$

In the above identity, we are using that $p(A)b$ has the form $A^\ell + q(A)b$ where $q(x)$ is any polynomial of degree at most $\ell - 1$, and the two equivalent characterizations for a vector to belong to a Krylov subspace:

$$\begin{aligned} v \in \mathcal{K}_\ell(A,b) &\iff v = V_\ell y \text{ for some } y \\ &\iff v = q(A)b \text{ for some } q(x) \text{ with degree at most } \ell - 1. \end{aligned}$$

Hence, the optimality condition can be imposed by asking the orthogonality $p(A)b \perp \mathcal{K}_\ell(A,b)$. Since $p(A)b \in \mathcal{K}_{\ell+1}(A,b)$ we may rewrite this condition as follows, for $j = 1, \ldots, \ell$:

$$v_j^* p(A)b = 0 \iff v_j^* V_\ell p(H_\ell) e_1 \|b\|_2 = 0,$$

where we have used Lemma 5.10.1 and the fact that $v_j^* v_{\ell+1} = 0$.

Since $v_j^* V_\ell = e_j^*$ we get

$$p(A)b \perp \mathcal{K}_\ell(A,b) \iff e_j^* p(H_\ell) e_1 = 0, \qquad j = 1, \ldots, \ell.$$

Thanks to Hamilton-Cayley's theorem if $p(z)$ is the characteristic polynomial of H_ℓ then $p(H_\ell) = 0$, which allows to conclude. □

Theorem 5.10.2 yields some information on the approximated eigenvalues computed by the Arnoldi iteration.

The Ritz values implicitly define the characteristic polynomial of H_ℓ, which approximates the one of A in the Hamilton-Cayley sense: even if it is not possible to have $p(A) \equiv 0$ (because of the lower degree), we try to at least make it as small as possible.

We then make a few remarks:

- The Ritz values are invariant for translations and scaling, in the sense that the process applied to $\mu A + \eta I$ yields Ritz values scaled and translated with the same factors.
- It is reasonable to assume that the largest eigenvalues (in modulus) are approximated well: their linear factors in the characteristic polynomials are the ones who contribute the most to make $p(A)$ small.
- If b lives in an invariant subspace, only the spectrum of A restricted to that invariant subspace can be determined through this method.

To give a more formal analysis of the convergence, we refer to the book by Saad [6]. The construction also allows to approximate eigenvalues in particular regions of the spectrum, for instance close to a certain point σ, by replacing A with $(A - \sigma I)^{-1}$; These two matrices have the same eigenvectors, and the latter has eigenvalues of the form $(\lambda_i - \sigma)^{-1}$. These are large if and only if λ_i is close to σ. This idea is known as *Inverse Arnoldi Iteration*, or *shift-and-invert Arnoldi*.

6 Sparse direct solvers for symmetric positive definite linear systems

In the context of solving linear systems with Krylov subspace iterative methods, we have seen that it is crucial to have under control the conditioning of the problem or to have at our disposal a good preconditioner. However, preconditioning is an art and often requires additional insights into the problem at hand.

On the other hand, many ill-conditioned linear systems coming from the discretization of PDEs are highly *sparse*, i.e., the coefficient matrix has only $\mathcal{O}(n)$ non zero entries. This is due to the fact that differential operators are local, e.g., the value of the derivative at a certain location depends only on the value of the function in the neighbourhood of that location. As we are going to see in this section, sparsity can sometimes be exploited to design efficient direct methods. The so-called *sparse direct solvers* are based on suitable modifications of the LU/Cholesky factorization, trying to minimize the amount of required memory and operations. Such factorization-based methods are only possible for an explicitly given sparse matrix A and their success depends on the pattern of sparsity in a complicated way. Nevertheless, these methods can be very successful and represent the methods of choice for dealing with finite differences (FD) and finite element discretizations (FE) of 2D PDEs. In these notes we restrict to the case where A is symmetric positive definite; this assumption simplifies the discussion and avoids the need for pivoting to ensure numerical stability. The general case is beyond the scope of this course as it is more complicated; indeed, pivoting and preservation of sparsity are in conflict and a compromise has to be made, e.g., use non-optimal pivot elements.

6.1 Cholesky factorization of positive definite matrices

Let us start by proving that any symmetric positive definite matrix admits a symmetric analogue of the LU decomposition, known as the Cholesky decomposition.

Lemma 6.1.1. *$A \in \mathbb{R}^{n\times n}$ is symmetric positive definite if and only if exists a lower triangular invertible matrix $L \in \mathbb{R}^{n\times n}$ such that $A = LL^T$.*

Proof. If $A = LL^T$ then A is clearly symmetric and for every $x \in \mathbb{R}^n \setminus \{0\}$ we have

$$x^T Ax = x^T LL^T x = \|L^T x\|_2^2 \geq 0.$$

For the other implication, we proceed by induction on n. For $n = 1$, A is a positive real number and the claim follows by choosing L as its square root. For $n > 1$, let us observe that the symmetric positive definite matrix A can be factorized as

$$A = \begin{bmatrix} a_{11} & a_1^T \\ a_1 & A_{22} \end{bmatrix} = \underbrace{\begin{bmatrix} \sqrt{a_{11}} & \\ \frac{a_1}{\sqrt{a_{11}}} & I_{n-1} \end{bmatrix}}_{L_1} \begin{bmatrix} 1 & \\ & A_{22} - \frac{a_1 a_1^T}{a_{11}} \end{bmatrix} \begin{bmatrix} \sqrt{a_{11}} & \frac{a_1^T}{\sqrt{a_{11}}} \\ & I_{n-1} \end{bmatrix}, \tag{6.1}$$

where $A_{22} - \frac{a_1 a_1^T}{a_{11}} \in \mathbb{R}^{(n-1)\times(n-1)}$ is again symmetric and positive definite as it can be seen as a principal submatrix of $L_1^{-1} A L_1^{-T}$, which is symmetric and positive definite. Using the induction step, we can claim that $A_{22} - \frac{a_1 a_1^T}{a_{11}} = L_2 L_2^T$ for a certain lower triangular matrix L_2. Then, conclude by writing

$$A = L_1 \begin{bmatrix} 1 & \\ & L_2 L_2^T \end{bmatrix} L_1^T = L_1 \begin{bmatrix} 1 & \\ & L_2 \end{bmatrix} \begin{bmatrix} 1 & \\ & L_2^T \end{bmatrix} L_1^T$$

and setting $L = L_1 \begin{bmatrix} 1 & \\ & L_2 \end{bmatrix}$. □

Definition 6.1.2. The matrix L and the product LL^T of Lemma 6.1.1 are said the *Cholesky factor* and the *Cholesky factorization* of A, respectively.

Iteratively repeating the main step in the proof of Lemma 6.1.1, i.e. equation (6.1), leads to Algorithm 8 for computing the Cholesky factorization. The Matlab function `chol` can be used to compute the Cholesky factorization. Attempting to compute the Cholesky factorization also happens to be the best way to check whether a given symmetric matrix is positive definite; in case of a non positive definite argument one encounters a negative diagonal entry during the process.

Algorithm 8 Cholesky factorization of A

```
1: procedure CHOL(A)
2:     for i = 1, ... n − 1 do
3:         l_ii = √a_ii
4:         l_{i+1:n,i} = a_{i+1:n,i} / l_ii
5:         a_{i+1:n,i+1:n} ← a_{i+1:n,i+1:n} − l_{i+1:n,i} l_{i+1:n,i}^T
6:     end for
7:     return L = (l_ij)
8: end procedure
```

6.2 Sparsity and Cholesky factorization

A natural question is whether a sparse symmetric positive definite matrix A has a sparse Cholesky factor L. In general, the Cholesky factor will not inherit the sparsity pattern of the matrix A. In Figure 1 and Figure 2 are reported two examples, involving arrowhead matrices, that have extreme behaviors: in the first the sparsity pattern is exactly preserved while, in the second, the matrix L is fully populated under the main diagonal.

Since the sparsity pattern of the first example can be obtained by applying the same row and column permutation on the matrix of the second example, we may learn an important lesson: The ordering of the matrix can have a tremendous impact on the sparsity of its Cholesky factor. This effect is well captured by the so-called *envelope* of the matrix A.

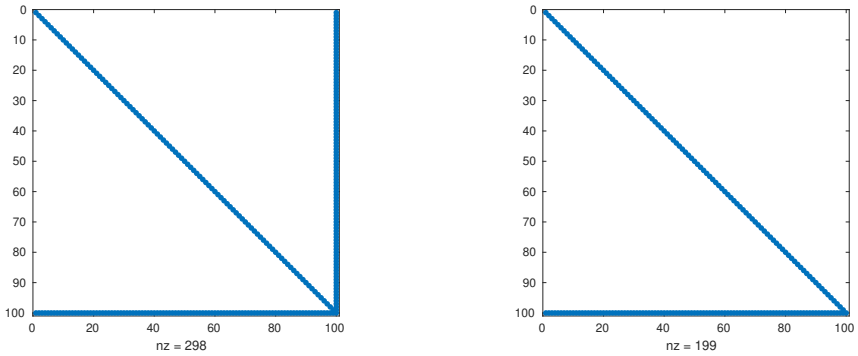


Figure 1: An arrowhead matrix A with arrow pointing to bottom right corner (left) and its Cholesky factor (right).

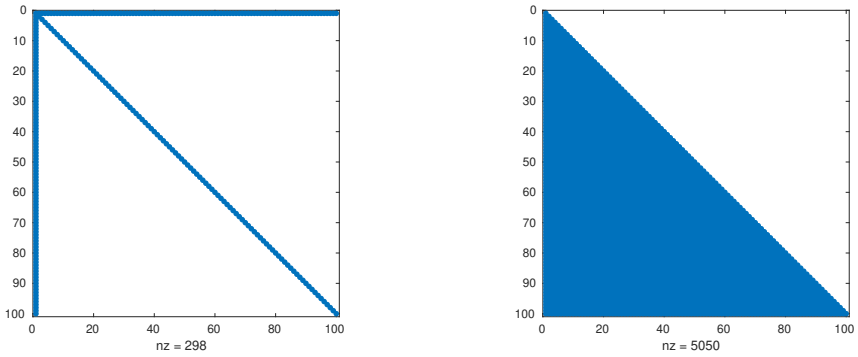


Figure 2: An arrowhead matrix A with arrow pointing to top left corner (left) and its Cholesky factor (right).

Definition 6.2.1. Let $A \in \mathbb{C}^{n\times n}$, we call the envelope of A the subset of indices of positions defined as

$$\text{env}(A) = \{(i,j) : \ J_i(A) \le j \le i\}, \qquad J_i(A) := \min\{j : \ a_{ij} \neq 0\}.$$

Note that the envelope of a matrix with the sparsity pattern displayed in Figure 2 contains all entries in lower triangular part. In contrast, the envelope of a matrix with the sparsity pattern displayed in Figure 1 only contains the entries on the main diagonal and in the last row. Indeed, we can prove that the envelope of a matrix is inherited by its Cholesky factor.

Theorem 6.2.2. *Let $A = LL^T \in \mathbb{R}^{n\times n}$ be a symmetric positive definite matrix, then*

$$(i,j) \notin \text{env}(A) \quad \Rightarrow \quad l_{ij} = 0.$$

Proof. We proceed by induction on n. For $n = 1$ the claim is trivially true. For $n > 1$, looking at Algorithm 8 we see that $l_{:,1}$ has the same non zero entries of $a_{:,1}$. Moreover, we observe that

$$a_{h1} = 0 \quad \Rightarrow \quad J_h(A_{22} - \frac{a_1 a_1^T}{a_{11}}) = J_h(A_{22}).$$

Using the induction step we have that

$$L = \begin{bmatrix} \sqrt{a_{11}} & \\ \frac{a_1}{\sqrt{a_{11}}} & L_2 \end{bmatrix}, \qquad L_2 L_2^T = A_{22} - \frac{a_1 a_1^T}{a_{11}},$$

where $J_h(L_2) \ge J_h(A_{22})$, for all h such that $a_{h,1} = 0$. The latter implies $\text{env}(L) \subseteq \text{env}(A)$ that is equivalent to the claim. □

One can easily modify Algorithm 8 such that it only computes the entries of L within the envelope. This reduces the cost from $\frac{1}{3}n^3 + \mathcal{O}(n^2)$ operations to

$$\frac{1}{2}\sum_{k=1}^{n} \omega_k(A)^2 + \text{ lower-order terms}$$

operations for the factorization, where $\omega_k(A) = |\{j \le k : \ (k,j) \in \text{env}(A)\}|$.

6.3 Fill-in and basic concepts from graph theory

Given a Cholesky factorization $A = LL^T$, indices (i,j) that satisfy $l_{ij} \neq 0$ but $a_{ij} = 0$ are called fill-in. From Theorem 6.2.2 we already know that fill-in can only take place within the envelope of A. In the following, we will discuss orderings of A that aim to reduce the envelope and the fill-in. On the matrix level, a symmetry-preserving reordering of the columns and rows of A corresponds to $P^T AP$ with a permutation matrix P. It is conceptually simpler to phrase such a reordering and its effect on the sparsity pattern

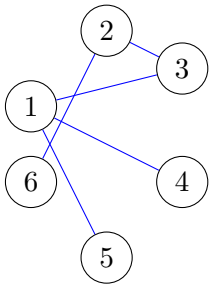
in terms of the associated graph. Given a symmetric matrix $A \in \mathbb{R}^{n\times n}$, we define an undirected graph $G = (V, E)$ with vertices $V = \{v_1, \ldots, v_n\}$ and

$$(v_i, v_j) \in E \quad \Leftrightarrow \quad a_{ij} \neq 0.$$

Then $P^T AP$ simply corresponds to a renumbering of the vertices. We will now study the effect of Algorithm 8 on the graphs associated with the matrices generated during the factorization.

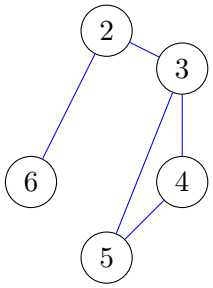
Example 6.3.1. Let $A \in \mathbb{R}^{6\times 6}$ with the following sparsity pattern:

$$\begin{bmatrix} \times & 0 & \times & \times & \times & 0 \\ 0 & \times & \times & 0 & 0 & \times \\ \times & \times & \times & 0 & 0 & 0 \\ \times & 0 & 0 & \times & 0 & 0 \\ \times & 0 & 0 & 0 & \times & 0 \\ 0 & \times & 0 & 0 & 0 & \times \end{bmatrix}$$



In terms of the graph, the first step of Algorithm 8 corresponds to introducing edges between vertices that are indirectly connected by a path of length 2 via v_1. Afterwards, all edges attached to v_1 and the vertex v_1 itself are eliminated:

$$\left[\begin{array}{c|ccccc} \times & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & \times & \times & 0 & 0 & \times \\ 0 & \times & \times & \times & \times & 0 \\ 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & \times & \times & \times & 0 \\ 0 & \times & 0 & 0 & 0 & \times \end{array}\right]$$



The construction in the example above can be easily generalized. Let $G^{(1)}$ be the graph associated with A. More generally, let $G^{(k)}$ denote the graph associated with the submatrix $A^{(k)}_{k:n,k:n}$ before the kth step of Algorithm 8 is executed. Then $G^{(k+1)}$ is obtained from $G^{(k)}$ by introducing edges between vertices that are indirectly connected by a path of length 2 via v_k, and then eliminating v_k along with its attached edges. The memory requirement for storing the Cholesky factor L is then given by

$$n + \sum_{k=1}^{n-1} \deg_{G^{(k)}}(v_k),$$

where the degree deg of a vertex is defined as the number of attached edges.

6.4 Ordering strategies

Algorithms for factorizing a sparse matrix typically consist of two phases. In the first phase, the sparsity pattern of A is analyzed and a suitable reordering aiming at a reduced fill-in is computed (this is often referred to as the *symbolic phase*). The second phase consists of a sophisticated implementation of Algorithm 8, restricting its storage and computations to the predicted sparsity pattern of L. In the following, we discuss three quite different strategies for the first phase: Reverse Cuthill-McKee, Approximate minimum degree, and nested dissection.

6.4.1 Reverse Cuthill-McKee

The Cuthill-McKee (CM) and Reverse Cuthill-McKee (RCM) aim at minimizing the bandwidth of a sparse matrix in a heuristic way. To attain this goal, an ordering is produced in which neighboring vertices obtain positions close together. The choice of

Algorithm 9 Cuthill-McKee ordering

```
1: procedure CM(G, E, v)
2:     Initialize FIFO queue F = (v)
3:     k ← 1
4:     while F ≠ ∅ do
5:         Remove first element v from F and assign number k to v
6:         Let W contain all unnumbered vertices in adj(v) = {w ∈ V : (v, w) ∈ E}
7:         Put all vertices from W into F in ascending order of degree.
8:         k ← k + 1
9:     end while
10: end procedure
```

starting vertex v for Algorithm 9 is important. Preferably, v should be a peripheral vertex, that is, there exists a path of maximal length starting from v. A (nearly) peripheral vertex can be found from repeated breadth first searches. If we let P_{CM} denote the permutation matrix associated with the numbering produced by Algorithm 9 then the corresponding reordered matrix takes the form $P_{\mathrm{CM}}^T A P_{\mathrm{CM}}$. It turns out that a better ordering is obtained when reversing the numbering obtained from Algorithm 9, the so-called *reverse Cuthill-McKee ordering* (RCM). We indicate the corresponding permutation matrix with P_{RCM}. It has been shown in the literature that we always have

$$|\mathrm{env}(P_{\mathrm{RCM}}^T A P_{\mathrm{RCM}})| \leq |\mathrm{env}(P_{\mathrm{CM}}^T A P_{\mathrm{CM}})|,$$

so the RCM procedure is usually preferred; the latter is implemented in the Matlab command `symrcm`.

6.4.2 Approximate minimum degree

While RCM is cheap and still quite popular, the obtained reordering is usually far from optimal. One problem with RCM is that it aims at minimizing the bandwidth, while

the ultimate target is to minimize the fill-in and not the bandwidth. Both minimization problems are NP-hard. A greedy approach to minimize the fill-in is presented in Algorithm 10. Note that $G^{(k)}$ refers to the elimination graphs introduced in Section 6.3, with the notable difference that the vertices to be eliminated are selected dynamically. Algorithm 10 is quite costly, in particular due to the need to evaluate the degrees of

Algorithm 10 Minimum degree ordering

1: **procedure** MD(G, E)
2: Set $G^{(1)} = (V, E)$
3: **for** $k = 1, \ldots, n$ **do**
4: Select vertex v of $G^{(k)}$ with minimum degree and assign number k to v.
5: Obtain $G^{(k+1)}$ by eliminating the vertex v and the associated edges as described in Section 6.3.
6: **end for**
7: **end procedure**

all vertices in $G^{(k)}$. A cheaper, nearly equally effective alternative has been developed, called the *Approximate Minimum Degree* (AMD) ordering. It is available in the Matlab command `symamd`.

6.4.3 Nested dissection

A graph separator S for an undirected graph G partitions the set of vertices V into three disjoint sets

$$V = V_1 \cup V_2 \cup S,$$

such that no edges exist that connect vertices in V_1 with vertices in V_2. If we use a numbering such that the vertices in V_1 appear first, then the vertices in V_2, and then the vertices in S, this means that the matrix takes the form

$$A = \begin{bmatrix} A_{V_1,V_1} & 0 & A_{V_1,S} \\ 0 & A_{V_2,V_2} & A_{V_2,S} \\ A_{S,V_1} & A_{S,V_2} & A_{S,S} \end{bmatrix}. \tag{6.2}$$

By Theorem 6.2.2, the Cholesky factor will inherit the zero off-diagonal block. The cardinality of the separator S should be small and preferably the cardinalities of V_1 and V_2 should be balanced. Recursively applying the dissection (6.2) leads to Algorithm 11. The art in Algorithm 11 lies in selecting a "good" separator. This is relatively easy for problems where the underlying geometry is known, for example in FE discretizations of 2D or 3D PDEs. Then repeated geometric subdivision of the computational domain generally leads to a good choice of separators. For problems without an underlying geometry, graph clustering techniques can be used to determine good separators.

Algorithm 11 Nested dissection ordering

1: **procedure** NESTEDDISSECTION(G, E)
2: Select a "good" separator S and corresponding partitioning $V = V_1 \cup V_2 \cup S$
3: Determine a numbering of the vertices within V_1 by applying the algorithm recursively to the subgraph of G associated with V_1
4: Determine a numbering of the vertices within V_2 by applying the algorithm recursively to the subgraph of G associated with V_2
5: Put vertices in V_1 first (according to the numbering from Step 3), vertices in V_2 second (according to the numbering from Step 4), and vertices in S third.
6: **end procedure**

7 Matrix functions

Together with linear systems and eigenvalue problems, the evaluation of matrix functions is an evergreen topic in numerical linear algebra. Indeed, a lot of applications involve the problem of evaluating a matrix function or a matrix function times a vector. For instance, the solution of a system of linear differential equations with constant coefficients of the form

$$\begin{cases} \dot{u}(t) = Au(t) \\ u(0) = u_0 \in \mathbb{R}^n \end{cases}, \qquad A \in \mathbb{R}^{n\times n},$$

is given in terms of the matrix exponential as $u(t) = e^{tA}u_0$. Other examples of functions of interest are

- $\log(A)$;
- $\sqrt{A}$;
- A^α, $\alpha \in (0, 1)$;
- $\operatorname{sign}(A)$.

This section is meant to be a brief excursion around the rigorous definitions of matrix functions and the methods for their computations; we refer to the excellent book by Higham [2] for a complete overview.

7.1 Equivalent definitions of $f(A)$

Given a square matrix $A \in \mathbb{C}^{n\times n}$ and a scalar function $f : \Omega \to \mathbb{C}$ with $\Omega \subseteq \mathbb{C}$, the matrix function $f(A)$ is again an $n \times n$ matrix. When $f \equiv p$ happens to be a polynomial

$$p(z) = p_0 + p_1 z + \cdots + p_m z^m$$

we can define the matrix function $p(A)$ by simply replacing z with A:

$$p(A) = p(z) = p_0 I + p_1 A + \cdots + p_m A^m,$$

where the power A^j is understood as multiplying j times A with itself. We can extend this definition to functions that are analytic on the whole $\mathbb{C}$, by replacing A in their scalar series expansion:

$$f(z) = \sum_{j=0}^{\infty} c_j z^j \quad \Rightarrow \quad f(A) = \sum_{j=0}^{\infty} c_j A^j.$$

However, we would like to have a definition that does not require analiticity on the entire complex plane and also provides an insight on how to compute or approximate $f(A)$. Moreover, we would like to preserve 2 properties that hold in the polynomial case:

(i) The eigenvalues of $f(A)$ are $f(\lambda_j)$ (where λ_j are the eigenvalues of A),

(ii) The eigenvectors of $f(A)$ coincides with those of A.

In view of the target properties, in the following, we will always assume that f is analytic on Ω and that $\Omega \subseteq \mathbb{C}$ contains the eigenvalues of A. Note that, when A is diagonalizable, properties (i) and (ii) already determine the expression of $f(A)$; indeed, if $A = VDV^{-1}$ with $D = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, then $f(A)$ is necessarily given by

$$f(A) = V f(D) V^{-1}, \qquad f(D) = \begin{bmatrix} f(\lambda_1) & & \\ & \ddots & \\ & & f(\lambda_n) \end{bmatrix}. \tag{7.1}$$

In the non diagonalizable case, things are slightly more complicated and it is instructive to look at the behavior of the simplest functions, i.e. the powers z^j, applied to the simplest non diagonalizable matrix, i.e. the Jordan block. Direct computations show that

$$A = \begin{bmatrix} \lambda & 1 & & \\ & \ddots & \ddots & \\ & & & 1 \\ & & & \lambda \end{bmatrix}, \qquad A^2 = \begin{bmatrix} \lambda^2 & 2\lambda & 1 & \\ & \ddots & \ddots & \ddots \\ & & \ddots & \ddots & 1 \\ & & & \ddots & 2\lambda \\ & & & & \lambda^2 \end{bmatrix}, \qquad A^3 = \begin{bmatrix} \lambda^3 & 3\lambda^2 & 3\lambda & 1 & \\ & \ddots & \ddots & \ddots & \ddots \\ & & \ddots & \ddots & \ddots & 1 \\ & & & \ddots & \ddots & 3\lambda \\ & & & & \ddots & 3\lambda^2 \\ & & & & & \lambda^3 \end{bmatrix}, \quad \ldots$$

and this suggests the following definition.

Definition 7.1.1 (Jordan canonical form)**.** Let $A = VJV^{-1}$, with $J = \mathrm{diag}(J_1, \ldots, J_s)$, be the Jordan canonical form of $A \in \mathbb{C}^{n \times n}$, then $f(A) := V f(J) V^{-1}$ where

$$f(J) = \begin{bmatrix} f(J_1) & & \\ & \ddots & \\ & & f(J_s) \end{bmatrix}$$

and if J_i is an $h \times h$ Jordan block associated with the eigenvalue λ, then

$$f(J_i) := \begin{bmatrix} f(\lambda) & f'(\lambda) & \dots & \frac{f^{(h-1)}(\lambda)}{(h-1)!} \\ & \ddots & \ddots & \vdots \\ & & \ddots & f'(\lambda) \\ & & & f(\lambda) \end{bmatrix}.$$

We remark that, the above definition is only based on the evaluation of the function f and some of its derivatives at the eigenvalues of A; more precisely, two functions that coincide on the eigenvalues of A, up to the right amount of derivatives yield the same matrix function, when applied to A. This inspires an alternative definition based on the Hermite polynomial approximation of the function f.

Definition 7.1.2 (Hermite interpolant)**.** Let $\text{ind}_{\lambda_i}(A)$ denote the size of the largest Jordan block associated to the eigenvalue λ_i of the matrix A. Then, we define $f(A) = q(A)$ where $q(z)$ is the Hermite polynomial of $f(z)$ that verifies

$$\frac{dq}{dz^j}(\lambda_i) = \frac{df}{dz^j}(\lambda_i), \qquad j = 0, \dots, \text{ind}_{\lambda_i}(A),$$

for every eigenvalue λ_i of A.

Remark 7.1.3. Note that, in the above definition the polynomial changes whenever the argument matrix A changes. In particular, we are not saying that the matrix function associated with f is a matrix polynomial.

Exercise 7.1.4. Leveraging that $f(A)$ coincides with evaluating a certain matrix polynomial in A, prove that for any invertible matrix $S \in \mathbb{C}^{n \times n}$, we have $S^{-1} f(A) S = f(S^{-1} A S)$.

Finally, a third definition is based on the Cauchy integral formula for functions that are holomorphic in a certain domain of $\mathbb{C}$.

Definition 7.1.5 (Contour integral)**.** Let Γ be a closed curve (possibly made of several connected components) contained in Ω and encircling the eigenvalues of A. Then

$$f(A) := \frac{1}{2\pi i} \int_\Gamma f(z)(zI - A)^{-1} dz$$

where the integral is applied component-wise to the argument matrix.

Remark 7.1.6. The definition based on the Cauchy integral formula is sometimes combined with quadrature formula for the integral, to provide approximation of $f(A)$.

Remark 7.1.7. The three definitions (Jordan canonical form, Hermite interpolant and contour integral) are equivalent, see [2] for the proof, and coincide with the matrix power series expansion when $f(z)$ has a single expansion on the whole Ω (for example this happens in the case $f(z) = e^z$).

Exercise 7.1.8. Prove the following properties concerning the matrix exponential:

(i) $\frac{\partial e^{tA}}{\partial t} = Ae^{tA}$.

(ii) If two $n \times n$ matrices A, B verify $AB = BA$ then $e^{A+B} = e^A \cdot e^B = e^B \cdot e^A$.

(iii) e^A is always invertible and $(e^A)^{-1} = e^{-A}$.

(iv) $\det(e^A) = e^{\mathrm{trace}(A)}$.

7.2 The Schur-Parlett algorithm for computing $f(A)$

On first sight, the most natural way to compute $f(A)$ seems to consist of diagonalizing A and using (7.1). However, this leads to loss of accuracy if the matrix V is not particularly well-conditioned. Unless A is Hermitian (or, more generally, a normal matrix) approaches based on diagonalization should be avoided. For a general matrix function $f(A)$, the Matlab command `funm` is based on first computing the Schur form

$$A = QTQ^*,$$

with an upper triangular matrix T and unitary matrix Q, and exploiting the relation $f(A) = Qf(T)Q^*$. This way, the initial problem boils down to evaluating $F = f(T)$, i.e., the matrix function of an upper triangular matrix. From the definition of matrix functions, it is clear that F is also upper triangular and its diagonal entries are given by

$$F_{11} = f(T_{11}), \quad F_{22} = f(T_{22}), \quad \dots \quad F_{nn} = f(T_{nn}).$$

The elements in the strictly upper triangular part are determined from the fact that F and T must commute, $FT = TF$. For instance in the 2×2 case

$$T = \begin{bmatrix} T_{11} & T_{12} \\ & T_{22} \end{bmatrix}, \qquad F = \begin{bmatrix} F_{11} & F_{12} \\ & F_{22} \end{bmatrix},$$

looking at the entry $(1, 2)$ of the matrices FT and TF yields the relation

$$T_{11}F_{12} + T_{12}F_{22} = F_{11}T_{12} + F_{12}T_{22} \quad \Rightarrow \quad F_{12} = T_{12}\frac{F_{11} - F_{22}}{T_{11} - T_{22}}.$$

In the general case, resolving this relation column by column gives Algorithm 12, which requires the diagonal entries (that is, the eigenvalues) of T to be mutually distinct. If this condition is not satisfied or if some diagonal elements are close to each other (in a certain sense), a block variant of the algorithm should be used. This block variant of Algorithm 12 is state-of-the-art for evaluating general functions of small to medium-sized matrices. Nevertheless, it is rarely used in practice. For nearly all functions of practical interest, specialized methods are the preferred choice. For example, the Matlab commands `expm` and `logm` are based on a totally different class of methods, the so-called *scaling and squaring algorithm*, see Higham's book for the details.

Remark 7.2.1. If we denote by c_f the cost of one evaluation of f then the asymptotic cost of the Schur-Parlett algorithm is $\mathcal{O}(n^3 + c_f n)$.

Algorithm 12 The Schur-Parlett algorithm for computing $f(A)$.

procedure SCHURPARLETT(A)
  Compute the Schur form $A = QTQ^*$
  for $i = 1, \dots n$ **do**
    $F_{ii} = f(T_{ii})$
  end for
  for $j = 2, \dots n$ **do**
    for $i = j-1, j-2 \dots 1$ **do**
      $F_{ij} = T_{ij}\frac{F_{ii}-F_{jj}}{T_{ii}-T_{jj}} + \sum_{k=i+1}^{j-1}(F_{ik}T_{kj} - T_{ik}F_{kj})$
    end for
  end for
  return QFQ^*
end procedure

7.3 The Arnoldi method for computing $f(A)b$

The inverse of a matrix, i.e. A^{-1}, can be viewed as the matrix function corresponding to $f(z) = z^{-1}$. With this perspective, we can look at computing a quantity of the form $f(A)b$, for a vector b, as a generalization of solving a square system of linear equations. Then, it is quite natural to extend FOM for approximating $f(A)b$.

Suppose that we have run ℓ steps of the Arnoldi method to generate an orthonormal basis for $\mathcal{K}_\ell(A, b)$. This yields the Arnoldi decomposition

$$AV_\ell = V_\ell H_\ell + h_{\ell+1,\ell} v_{\ell+1} e_\ell^*.$$

Then the approximation considered by the Arnoldi method for $f(A)b$ (the extension of FOM for linear systems) is

$$f_\ell := \|b\|_2 V_\ell f(H_\ell) e_1 \approx f(A)b.$$

The latter requires the evaluation of the $\ell \times \ell$ matrix function $f(H_\ell)$, that can be addressed by Algorithm 12 with cost $\mathcal{O}(\ell^3)$ in the typical case, i.e., when evaluating the scalar function f is not the dominant cost. Similarly to FOM, we can link the convergence of the method to the best polynomial approximation of the function $f(z)$ over spectral sets for A. In particular, in the Hermitian case we have the following result.

Theorem 7.3.1. *Let $A \in \mathbb{C}^{n\times n}$ be a Hermitian matrix with eigenvalues contained in the interval $[\alpha, \beta] \subset \mathbb{R}$ and let $f : \Omega \to \mathbb{C}$ be analytic with $[\alpha, \beta] \subset \Omega$. Then*

$$\|f(A)b - f_\ell\|_2 \leq 2\|b\|_2 \min_{p(z)\in\mathcal{P}_{\ell-1}} \max_{z\in[\alpha,\beta]} |f(z) - p(z)|.$$

Proof. Since V_ℓ is an orthogonal basis for $\{p(A)b : \deg(p) \leq \ell - 1\}$ we have that

$$V_\ell V_\ell^* b = b, \qquad V_\ell V_\ell^* p(A)b = p(A)b$$

for any polynomial $p(z)$ of degree at most $\ell - 1$. Then, in view of Lemma 5.10.1, $p(A)b = \|b\|_2 V_\ell p(H_\ell)e_1$, and therefore the approximation returned after ℓ steps of the Arnoldi method is exact if the $f(z)$ is a polynomial of degree at most $\ell - 1$. Therefore, for any polynomial $p(z)$ of degree at most $\ell - 1$:

$$\begin{aligned}
\|f(A)b - f_\ell\|_2 &= \|f(A)b - p(A)b + \|b\|_2 V_\ell p(H_\ell)e_1 - f_\ell\|_2 \\
&\leq \|f(A)b - p(A)b\|_2 + \|b\|_2 \|p(H_\ell)e_1 - f(H_\ell)e_1\|_2 \\
&\leq \|b\|_2 \left(\|f(A) - p(A)\|_2 + \|p(H_\ell) - f(H_\ell)\|_2\right) \\
&= \|b\|_2 \left(\max_{z\in\Lambda(A)} |f(z) - p(z)| + \max_{z\in\Lambda(H_\ell)} |f(z) - p(z)|\right) \\
&\leq 2\|b\|_2 \max_{z\in[\alpha,\beta]} |f(z) - p(z)|,
\end{aligned}$$

where, in the last inequality, we have used that both $\Lambda(H_\ell)$ and $\Lambda(A)$ are contained in $[\alpha, \beta]$. Taking the minimum over p yields the claim. □

A bound for a general matrix A can be retrieved by combining the argument used for the proof of Theorem 7.3.1 with Crouzeix-Palencia's result (Theorem 5.7.2).

Corollary 7.3.2. *Let $A \in \mathbb{C}^{n\times n}$ and let $f : \Omega \to \mathbb{C}$ be analytic with $\mathcal{W}(A) \subset \Omega$. Then*

$$\|f(A)b - f_\ell\|_2 \leq 2(1 + \sqrt{2})\|b\|_2 \min_{p(z)\in\mathcal{P}_{\ell-1}} \max_{z\in\mathcal{W}(A)} |f(z) - p(z)|.$$

Proof. The proof is analogous to the one of Theorem 7.3.1, apart from using Crouzeix-Palencia's bound and the property of the numerical range: $\mathcal{W}(H_\ell) = \mathcal{W}(V_\ell^* A V_\ell) \subseteq \mathcal{W}(A)$. □

References

[1] James W. Demmel. *Applied numerical linear algebra.* SIAM, 1997.

[2] Nicholas J Higham. *Functions of matrices: theory and computation.* SIAM, 2008.

[3] Roger A. Horn and Charles R. Johnson. *Matrix analysis.* Cambridge university press, 1985.

[4] Roger A. Horn and Charles R. Johnson. *Topics in Matrix analysis.* Cambridge university press, 2012.

[5] Tosio Kato. *Perturbation theory for linear operators*, volume 132. Springer Science & Business Media, 2013.

[6] Yousef Saad. *Iterative methods for sparse linear systems.* SIAM, 2003.

[7] David S. Watkins. The transmission of shifts and shift blurring in the QR algorithm. *Linear algebra and its applications*, 241:877–896, 1996.