1 Iterative methods for large sparse eigenvalue problems

The eigenvalue problem is one of the fundamental problems in science. We wish to compute pairs $\lambda \in \mathbb{C}$ and $x \in \mathbb{C}^n$ such that

$$Ax = \lambda x, \tag{1.1}$$

and $A \in \mathbb{R}^{n\times m}$. The eigenvector is in this block assumed to be normalized as $\|x\| = 1$, with Euclidean norm such that $\|x\|^2 = x^H x = 1$.

1.1 Basic methods

1.1.1 Computing eigenvalues from eigenvectors

Before diving into the algorithms for eigenvalue problems, we will treat an easier problem.
Problem: Suppose $x \in \mathbb{C}^n$ is an approximation of an eigenvector, compute an associated eigenvalue.

Assume for the moment an idealized situation where x is exactly an eigenvector. This means that (1.1) is satisfied, and we can multiply the equation from the left with x^H:

$$x^H Ax = \lambda x^H x,$$

such that

$$\lambda = \frac{x^H Ax}{x^H x}$$

This quotient can be used also if x is not an eigenvector and is usually referred to as the Rayleigh quotient.

Definition 1.1.1 (Rayleigh quotient)**.** *The quotient defined by*

$$r(x) := \frac{x^H Ax}{x^H x}$$

is referred to as the Rayleigh quotient.

For symmetric matrices, there is an additional interpretation of the Rayleigh quotient. Given an approximate eigenvector x, it minimizes $Ax - \mu x$ in the Euclidean norm: One can show that

$$\operatorname*{argmin}_{\mu\in\mathbb{R}} \|\mu x - Ax\| = \frac{x^H Ax}{x^H x}$$

The Rayleigh quotient will in general only give you an approximation of the eigenvalue. The propagation of the approximation error can also be precisely described if x is sufficiently close to an eigenvector. More precisely, if we suppose $x \in \mathbb{C}^n$ is an eigenvector corresponding to an eigenvalue λ, we have that

$$r(x + \varepsilon y) = \lambda + \mathcal{O}(\varepsilon). \tag{1.2}$$

In words, the error in the eigenvalue from the Rayleigh quotient is essentially of the order of magnitude of the error in the eigenvector. If A is symmetric (or hermitian) we have

Note that $\mathcal{O}(\varepsilon^2)$ is better than $\mathcal{O}(\varepsilon)$ since we consider small ε.

$$r(x + \varepsilon y) = \lambda + \mathcal{O}(\varepsilon^2). \tag{1.3}$$

Rayleigh quotient

Consider the two matrices

$$A_1 = \begin{bmatrix} 2 & 5 \\ 0 & 3 \end{bmatrix} \text{ and } A_2 = \begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix}.$$

Both matrices have an eigenvector $x = \left[\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right]$ with eigenvalue $\lambda = 2$, but A_2 is symmetric. The example code below illustrates that the Rayleigh quotient is much closer to the eigenvalue for the symmetric matrix A_2.

```
>> A1=[2 0;0 3];
>> A2=[2 5;0 3];
>> x=[1;0];
>> y=[1;1]; e=1e-4 % small perturbation
>> z=x+e*y;
>> z'*A1*z/(z'*z)
ans =
   2.000499959998002
>> z'*A2*z/(z'*z)
ans =
   2.000000009998000
```

○

1.1.2 *Basic eigenvalue methods*

The Rayleigh quotient provides a procedure to numerically compute an eigenvalue approximation given an eigenvector approximation. Computing the eigenvector can be done in many ways. We first consider three basic algorithms.

Read about these methods in TB pages 202-209.

- Power method (power iteration) summarized in Algorithm 1
- Inverse iteration summarized in Algorithm 2
- Rayleigh quotient iteration summarized in Algorithm 3

1.1.3 *Power method*

The power method (or sometimes power iteration), is our first eigenvalue method. It consists of starting vector a vector v_0, we this vector with A, scale the resulting vector and repeat the process:

$$v_{k+1} = \alpha_k A v_k, \quad k = 0, \ldots.$$

The scaling factor α_k is used to prevent the iteration values v_k to become very small or very large which makes them more difficult to represent/store. (More precisely, we want to avoid overflow or underflow in the IEE floating point arithmetic.) Typically the scaling is selected such that $\|v_{k+1}\| = 1$, which can be achieved by setting

$$\alpha_k = \frac{1}{\|Av_k\|}.$$

An important property of the power method is that the only way we need to access the matrix A is in combination with a multiplication with a vector Ax: a so-called *matrix-vector product*. In many scientific applications, the matrix A may be so large that it is not possible to store it explicitly, but the matrix-vector product may still be available.

The operations can be re-ordered such it only requires one matrix vector product per iteration as in Algorithm 1.

If we consider v_k as our eigenvector approximation, we can use the Rayleigh quotient to extract an eigenvalue approximation. Since $\|v_k\|_2^2 = v_k^T v_k = 1$, the Reyleigh quotient reduces to

$$\tilde{\lambda}_k = \frac{v_k^T A v_k}{v_k^T v_k} = v_k^T A v_k.$$

Input: A starting vector v with $\|v\| = 1$
Output: Eigenpair approximation $(w, \tilde{\lambda})$
for $n = 1, 2, \ldots$ **do**
  $w = Av$
  $v = w/\|w\|$
  $\tilde{\lambda} = v^T A v$
end

Algorithm 1: Power method (Power iteration).

1.1.4 *Convergence of the power method*

It turns out that the iterates v_k generated by the power method do indeed in general converge to an eigenvector. Under certain (not very restrictive) conditions one can show that

$$\tilde{\lambda}_k = \lambda_1 + \mathcal{O}\left(\frac{|\lambda_2|^k}{|\lambda_1|^k}\right).$$

where we have ordered the eigenvalues as $|\lambda_1| \geq |\lambda_2| \geq \cdots$. See lecture, TB (and wiki) for a proof.

1.1.5 *Inverse iteration*

The next algorithm we consider is essentially a combination of what we know for the power method, and the observation that the eigenvalues of the matrix A and the matrix

$$B = (A - \mu I)^{-1}$$

are related by a simple relation.

If we denote $\lambda_i(A)$, $\lambda_i(B)$ the eigenvalues of A and B respectively, the eigenvalues are related by

$$\lambda_i(B) = \frac{1}{\lambda_i(A) - \mu} \tag{1.4}$$

The eigenvectors remain unchanged. See TB, lecture (and wiki?) for proof.

Unlike the power method, inverse iteration does not involve a matrix vector product with A per iteration, but one solution to the linear system, $(A - \mu I)^{-1}v$, normally called a *linear solve*. A linear solve is in general much more computationally expensive than one matrix vector product.

The transformation (1.4) has the useful property that the eigenvalues close to μ will be transformed to large eigenvalues. Since inverse iteration converges to the eigenvector corresponding to the largest eigenvalue in general, we obtain with the application of the power method to B, which converges to the eigenvector corresponding to an eigenvalue of A, closest to μ. The eigenvector extraction can be done with the Rayleigh quotient of A, rather than B, as shown in Algorithm 2.

Input: A starting vector v with $\|v\| = 1$ and shift μ
Output: Eigenpair approximation $(w, \tilde{\lambda})$
for $n = 1, 2, \ldots$ **do**
  Solve linear system $(A - \mu I)w = v$
  $v = w/\|w\|$
  $\tilde{\lambda} = v^T A v$
end

Algorithm 2: Inverse iteration

1.1.6 *Rayleigh quotient iteration*

Among the final basic algorithms, is again a combination of previous ideas. We can set μ in inverse iteration as the Rayleigh quotient. This implies that when the eigenvector is a good approximation, the corresponding eigenvalue of $B = (A - \mu I)^{-1}$ will be a very large value and therefore converge faster.

See TB for a formalization.

Note that Rayleigh qoutient iteration can also be used for non-symmetric matrices, although it is presented only for symmetric matrices in TB.

Input: A starting vector v with $\|v\| = 1$
Output: Eigenpair approximation $(w, \tilde{\lambda})$
Set $\tilde{\lambda} = \mu$
for $n = 1, 2, \ldots$ **do**
  Solve linear system $(A - \tilde{\lambda} I)w = v$
  $v = w/\|w\|$
  $\tilde{\lambda} = v^T A v$
end

Algorithm 3: Rayleigh Quotient Iteration

1.2 *Orthogonal matrices and orthogonalizing vectors*

In basic linear algebra, we learn that two vectors $x, y \in \mathbb{R}^n$ are orthogonal when $y^T x = 0$. The concept of orthogonality, and its generalization to matrices is very important in this course. We will use it mostly in different factorizations and decompositions of matrices.

The use of decompositions has been selected as one of the most influential concepts in algorithms in the 20th century: https://www.siam.org/pdf/news/637.pdf In this course we also cover other algorithms in the list of important algorithms.

1.2.1 *Gram-Schmidt procedures*

The Gram-Schmidt procedure is often explained as a procedure to orthogonalize vectors, meaning that given vectors stored in a matrix $F = [f_1, \ldots, f_m] \in \mathbb{R}^{n \times m}$ with $n \geq m$ we try to determine $q_1, \ldots, q_n$ such that $q_1, \ldots, q_n$ are orthonormal and

$$\text{span}(f_1, \ldots, f_m) = \text{span}(q_1, \ldots, q_m).$$

You have normally learned about the Gram-Schmidt procedure in basic linear algebra courses. We repeat it in a slightly different notation than normal (using orthogonal matrices). It turns out that the classical Gram-Schmidt is not always satisfactory.

Such vectors $q_1, \ldots, q_n$ exist if $f_1, \ldots, f_m$ are linearly independent vectors. Note that the matrix $Q = [q_1, \ldots, q_m] \in \mathbb{R}^{n \times m}$ is orthogonal in the sense of definition of orthogonal matrices (see background.pdf).

The Gram-Schmidt procedure can be directly derived by inductively applying the following result.

In numerical linear algebra, the Gram-Schmidt procedure directly derived from Lemma 1.2.1 is typically called the *classical* Gram-Schmidt procedure in order to distinguish it from variants we discuss later.

Lemma 1.2.1. *Suppose $Q = [q_1, \ldots, q_m] \in \mathbb{R}^{n \times m}$ is an orthogonal matrix and suppose $b \notin \text{span}(q_1, \ldots, q_m)$. Let*

$$h = Q^T b$$

and

$$z = b - Qh = (I - QQ^T)b. \tag{1.5}$$

The vector $h \in \mathbb{R}^n$ is typically referred to as the Gram-Schmidt coefficients

Let $\beta = \|z\|$ and define

$$q_{m+1} := \frac{z}{\beta} \tag{1.6}$$

Then,

(a) the matrix $[q_1, \ldots, q_{m+1}]$ is an orthogonal matrix;

(b) $b = h_1 q_1 + \cdots + h_m q_m + \beta q_{m+1}$; and

(c) $\text{span}(q_1, \ldots, q_{m+1}) = \text{span}(q_1, \ldots, q_m, b)$.

Proof. Proof of (b): This is a direct consequence of (1.5) and (1.6). Proof of (a): Note that

$$[q_1, \ldots, q_{m+1}]^T [q_1, \ldots, q_{m+1}] = [Q, q_{m+1}]^T [Q, q_{m+1}] = \begin{bmatrix} Q^T Q & Q^T q_{m+1} \\ q_{m+1}^T Q & q_{m+1}^T q_{m+1} \end{bmatrix}$$

The conclusion (a) follows from the fact that $Q^T Q = I$,

$$Q^T q_{m+1} = Q^T (I - QQ^T) b = 0$$

and $q_{m+1}^T q_{m+1} = 1$.
Proof of (c): In this course we will several times use the general property that if two rectangular matrices $W \in \mathbb{R}^{n \times m}$ and $V \in \mathbb{R}^{n \times m}$ are related by

$$W = VP \tag{1.7}$$

for some non-singular matrix $P \in \mathbb{R}^{m \times m}$, then then $\text{span}(W) = \text{span}(V)$. If we select P as

$$P = \begin{bmatrix} I & h \\ 0 & \|z\| \end{bmatrix}$$

then (1.7) is satisfied with $V = [Q, q_{m+1}]$ and $W = [Q, b]$. □

Classical Gram-Schmidt example

```
>> Q=(1/sqrt(2))*[1 -1; 1 1; 0 0; 0 0];
>> Q'*Q   % Check if Q is orthogonal
ans =
    1.0000         0
         0    1.0000
>> b=randn(4,1);
>> h=Q'*b;            % Compute Gram-Schmidt coefficients
>> z=b-Q*h;           % Compute "orthogonal complement"
>> beta=norm(z);
>> q_new=z/beta;
>> Q_new=[Q,q_new];   % Construct new Q-matrix
>> Q_new'*Q_new       % Check that Q_new is orthogonal
ans =
    1.0000         0         0
         0    1.0000         0
         0         0    1.0000
>> norm(Q_new*[eye(2), h; zeros(1,2), norm(z)]-[Q,b])
>> P=[eye(2), h; zeros(1,2), beta];
>> norm(Q_new*P-[Q,b])  % Check that span(Q_new)=span([Q,b])
ans =
   1.1444e-16
```


Although the above example suggests that classical Gram-Schmidt works, it will in general not be satisfactory in our context. It turns out that the classical Gram-Schmidt is very sensitive to round-off errors in certain situations.

We now investigate what happens if we have an error in the computation of the Gram-Schmidt coefficients. In other words, we assume that h is approximated by

$$\tilde{h} = \begin{bmatrix} (1+\varepsilon_1)h_1 \\ \vdots \\ (1+\varepsilon_m)h_m \end{bmatrix} = \Big(\begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix} + \underbrace{\begin{bmatrix} \varepsilon_1 & & \\ & \ddots & \\ & & \varepsilon_m \end{bmatrix}}_{\Lambda_\varepsilon} \Big) Q^T b \tag{1.8}$$

In practice, we have round-off errors in every floating point operation and a complete round-off error analysis is quite cumbersome. In our simplified analysis we assume that no error is intruduced in the computation of z and $\tilde{q}_{m+1}$. In particular, no additional round-off error is introduced in (1.9) and (1.10).

where $\varepsilon_1, \ldots, \varepsilon_m$ are a small number introduced by the inexact evaluation of $Q^T b$, typically of order of the same order of magnitude ϵ_{mach}. Our approximation of z satisfies

$$\tilde{z} = b - Q\tilde{h} = b - Q\Lambda_\varepsilon Q^T b(1+\varepsilon) = z - Q\Lambda_\varepsilon Q^T b \tag{1.9}$$

such that

$$\tilde{q}_{m+1} = \frac{1}{\|\tilde{z}\|}\tilde{z} = \frac{1}{\|z - Q\Lambda_\varepsilon Q^T b\|}\tilde{z} = \frac{1}{\sqrt{(z - Q\Lambda_\varepsilon Q^T b)^T (z - Q\Lambda_\varepsilon Q^T b)}}\tilde{z} = \frac{1}{\sqrt{(z - Q\Lambda_\varepsilon Q^T b)^T (z - Q\Lambda_\varepsilon Q^T b)}}\tilde{z} = \frac{1}{\sqrt{\|z\|^2 + \|\Lambda_\varepsilon\|^2 \|QQ^T b\|^2}}\tilde{z} = \tilde{z}\left(\frac{1}{\|z\|} + \mathcal{O}(\varepsilon^2)\right), \tag{1.10}$$

where $\varepsilon = \|\Lambda_\varepsilon\|$. The approximation of the new vector is

$$\tilde{q}_{m+1} = (z - Q\Lambda_\varepsilon Q^T b)\left(\frac{1}{\|z\|} + \mathcal{O}(\varepsilon^2)\right) = \frac{z}{\|z\|} - \frac{1}{\|z\|}Q\Lambda_\varepsilon Q^T b + \mathcal{O}(\varepsilon^2) \tag{1.11}$$

In this first-order estimation, we see that the error is small if

$$\frac{\|Q\Lambda_\varepsilon Q^T b\|}{\|z\|} = \frac{\|\Lambda_\varepsilon Q^T b\|}{\|z\|} \leq \varepsilon \frac{\|Q^T b\|}{\|z\|}$$

is small.

A bad situation can easily be identified, since we can construct a situation where $\|z\|$ is small but $Q^T b$ is not: Suppose $b = q + \delta e$ where $q = Qd$ and $e \perp Q$ and $\|e\| = 1$. A direct computation leads to

$$\|\tilde{q}_{m+1} - \frac{z}{\|z\|}\| \leq \frac{|\varepsilon|}{|\delta|}\|Qd\| + \mathcal{O}(\varepsilon^2).$$

which suggests that the round-off error is proportional to $|\varepsilon|/|\delta|$.

Conclusion of error analysis of classical Gram-Schmidt method. The Gram-Schmidt procedure is likely to have a large round-off error if the vector b almost lies in the subspace span(Q).

Modified Gram-Schmidt

In this course we consider two variations of Gram-Schmidt which aim to improve the floating-point arithmetic problems described above.

We now derive the algorithm called *the modified Gram-Schmidt procedure* from the classical Gram-Schmidt procedure. For theoretical purposes we express the classical Gram-Schmidt in for-loops:

The modified Gram-Schmidt procedure is equivalent to the classical Gram-Schmidt procedure in exact arithmetic, but different floating-point arithmetic.

```
for i=1:m
  h(i)=Q(:,i)'*b;
end
z=b;
for i=1:m
  z=z-h(i)*Q(:,i)
end
```

Note that at iteration i of the second loop, we only need `h(i)` computed at the ith iteration the first loop such that we can merge the two loops:

```
z=b;
for i=1:m
  h(i)=Q(:,i)'*b;
  z=z-h(i)*Q(:,i);
end
beta=norm(z);
```

Although modified Gram-Schmidt yields a different result in floating point arithmetic, it is not always clear that the result is better. In fact, theoretical understanding for this is still disputed by some scientists. You will investigate this in practice by for a specific situation in the homeworks.

Caution regarding terminology: In this course we consider $Q \in \mathbb{R}^{n\times m}$ as an orthogonal matrix and want to orthogonalize b which result in algorithms above. In some literature (such as TB) Gram-Schmidt procedures are described for orthogonalizing an entire matrix $A \in \mathbb{R}^{n\times(m+1)}$.

In the first step inside the for-loop, the vector `z` can be explicitly expressed as:

- Iteration $i = 1$: $z = b$
- Iteration $i = 2$: $z = b - h_1 q_1$
- $\vdots$
- Iteration $i = m$: $z = b - h_1 q_1 - \cdots - h_m q_{m-1}$

Now recall that the vectors $q_1, \ldots, q_m$ are assumed to be orthogonal. The following identies can be directly identified.

- Iteration $i = 1$: $q_i^T z = q_i^T b$
- Iteration $i = 2$: $q_i^T z = q_2^T (b - h_1 q_1) = q_2^T b - h_1 q_2^T q_1 = q_i^T b$
- $\vdots$
- Iteration $i = m$: $q_i^T z = q_m^T b - h_1 q_m^T q_1 - \cdots - h_m q_m^T q_{m-1} = q_m^T b = q_i^T b$

Note that for every iteration we have $q_i^T z = q_i^T b$. Therefore, we can replace `Q(:,i)'*b` with `Q(:,i)'*z` in the for-loop. This is what we call the modified Gram-Schmidt method.


```
z=b;
for i=1:m
  h(i)=Q(:,i)'*z;
  z=z-h(i)*Q(:,i)
end
```

Double Gram-Schmidt

The next approach to improve the classical Gram-Schmidt procedure is very naive. Since we know that round-off errors will make the vector $z = b - Qh$ to not be orthogonal in practice, we can try to make it orthogonal by applying classical Gram-Schmidt again. This is what is called repeated Gram-Schmidt, or the special case double Gram-Schmidt.

```
>>  h=Q'*b;
>>  z=b-Q*h;
>>  g=Q'*z;
>>  z=z-Q*g
>>  h=h+g;
>>  beta=norm(z);
```

1.3 *Krylov methods*

The power method was the basis of both inverse iteration and Rayleigh quotient iteration. These algorithms can be used to compute one eigenvector given an initial guess. In order to compute several eigenvalues we now extend the power method in a different way. We consider the space spanned by the iterates of the power method.

Definition 1.3.1. *Krylov subspace The span of the iterates of the power methods is called a Krylov subspace*

$$\mathcal{K}_m(A, b) := \text{span}(b, Ab, A^2b, \ldots, A^{m-1}b).$$

Due to rounding error issues, the Krylov subspace is usually not computed from $[b, Ab, A^2b, \ldots, A^{m-1}b]$, but rather represented with an orthogonal basis of $\mathcal{K}_m(A, b)$. The Arnoldi method can be seen as method to compute an orthogonal bais of a Krylov subspace. More precisely, the Arnoldi method is a method which generates an orthogonal matrix $Q_m \in \mathbb{C}^{n\times m}$ such that

$$AQ_m = Q_{m+1}\underline{H}_m$$

where $\underline{H}_m \in \mathbb{R}^{(m+1)\times m}$ and $Q_{m+1} = [Q_m, q_{m+1}]$. The matrix $\underline{H}_m$ is a so-called Hessenberg matrix, which means that the elements below the

first lower off-diagonal are zero:

$$\underline{H}_m = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}$$

The Arnoldi method can be used to compute many quantites. In the context of eigenvalue computations, we take the eigenvalues of $H_m \in \mathbb{C}^{m\times m}$ as eigenvalue approximations.

Arnoldi's method for eigenvalue problems is also discussed in TB pages 251–264.

1.3.1 *The Arnoldi method*

If we combine the (Gram-Schmidt) orthogonalization process with the Krylov subspace, we obtain the algorithm called the Arnoldi method.

Input: A starting vector b
Output: Eigenpair approximation
Set $q_1 = b/\|b\|$, H_0 =empty matrix
for $m = 1, 2, \ldots$ **do**
  Compute $x = Aq_m$
  Orthogonalize x against $q_1, \ldots, q_m$ by computing $h \in \mathbb{C}^m$ and $x_\perp \in \mathbb{C}^n$ such that $Q^T x_\perp = 0$ and

$$x_\perp = x - Qh.$$

  Let $\beta = \|x_\perp\|$
  Let $q_{n+1} = x_\perp/\beta$
  Expand $\underline{H}_{m-1}$ with one column:

$$\underline{H}_m := \begin{bmatrix} \underline{H}_{m-1} & h \\ 0 & \beta \end{bmatrix}$$

end

Algorithm 4: Arnoldi's method for eigenvalue problems.

1.3.2 The Lanczos method

Specialization of Arnoldi's method for symmetric matrices.

The Lanczos iteration is also described in TB pages 276-278.

Output: Eigenpair approximations
Input: The matrix A and vector b.
b =arbitrary, $q_1 = b/\|b\|$, H_0 =empty matrix
for $m = 1, 2, \ldots$ **do**
  $v = Aq_m$
  $\alpha_m = q_m^T v$
  $v = v - \beta_{m-1} q_{m-1} - \alpha_m q_m$
  $\beta_m = \|v\|$
  $q_{m+1} = v/\beta_m$
end

1.4 Convergence of Arnoldi's method for eigenvalue problems

Recall that, unless it breaks down, k steps of the Arnoldi method generates an orthogonal basis of a Krylov subspace, represented by a matrix $Q = [q_1, \ldots, q_k] \in \mathbb{C}^{n\times k}$ such that $Q^*Q = I$ and

$$\text{span}(q_1, \ldots, q_k) = \mathcal{K}_k(A, b) := \text{span}(b, Ab, \ldots, A^{k-1}b).$$

The eigenvalue approximations (called Ritz values) are subsequently found from the eigenvalues of

$$H = Q^* A Q.$$

The matrix $H \in \mathbb{C}^{k\times k}$ is a Hessenberg matrix and can be generated as a by-product of the Arnoldi method. We call a pair (μ, Qv) a Ritz pair and Qv a Ritz vector, if v and μ safisfy

$$Hv = \mu v.$$

1.4.1 Bound for subspace-eigenvector angle

As a first indicator of the convergence we will characterize the following quantity

$$\text{error in eigenvector } x_i \sim \|(I - QQ^*)x_i\| \tag{1.12}$$

Recall: $Q \in \mathbb{C}^{n\times k}$ is an orthogonal matrix which means that $Q^*Q = I \in \mathbb{C}^{k\times k}$. However, $I \neq QQ^* \in \mathbb{C}^{n\times n}$.

where

$$Ax_i = \lambda_i x_i.$$

It is very natural to associate the accuracy of the eigenvector with this quantity from a geometric perspective. The indicator in the right-hand side of (1.12) is called (the norm of) the orthogonal complement of the projection of x_i onto the space spanned by Q and it can be interpreted

as the sine of the canonical angle between the Krylov subspace and an eigenvector. For the moment, we will only justify this indicator with this geometric reasoning and the following observation:

Lemma 1.4.1. *Suppoe (λ_i, x_i) is an eigenpair A. If the Krylov subspace contains the eigenvector ($x_i \in \mathcal{K}_k(A,b)$), then the indicator vanishes $\|(I - QQ^*)x_i\| = 0$ and there is at least one Ritz value μ such that $\mu = \lambda_i$.*

The Arnoldi method produces an exact approximation if the Krylov subspace contains an eigenvector, or equivalently the indicator is zero.

In words:

- Suppose the Krylov subspace contains the eigenvector ($x_i \in \mathcal{K}_k(A,b)$). Then, there exists a vector $z \in \mathbb{C}^k$ such that $x_i = Qz$. Moreover, this is an eigenvector of H such that the Arnoldi method will generate an exact eigenvalue of A. Moreover, the indicator is $\|(I - QQ^*)x_i\| = \|(I - QQ^*)Qz\| = 0$.
- If, similar to above, $x_i \approx x \in \mathcal{K}_k(A,b)$, we expect the indicator to be small and an eigenvalue of H also to be close λ_i.

The indicator can be bounded as follows, where we assume diagonalizability of the matrix.

Theorem 1.4.2. *Suppose $A \in \mathbb{C}^{n\times n}$ is diagonalizable and let the matrix $X = (x_1, \ldots, x_n) \in \mathbb{C}^{n\times n}$ and diagonal matrix $\Lambda \in \mathbb{C}^{n\times n}$ be the Jordan decomposition such that*

$$A = X\Lambda X^{-1}.$$

Suppose $\alpha_1, \ldots, \alpha_n \in \mathbb{C}\backslash\{0\}$ are such that

$$b = \alpha_1 x_1 + \cdots + \alpha_n x_n \tag{1.13}$$

Recall: The eigenvectors of a diagonalizable matrix form a basis of $\mathbb{C}^n$.

and

$$\varepsilon_i^{(m)} := \min_{\substack{p\in P_{m-1} \\ p(\lambda_i)=1}} \max(|p(\lambda_1)|, \ldots, |p(\lambda_{i-1})|, |p(\lambda_{i+1})|, \ldots, |p(\lambda_n)|)$$

where P_n denotes polynomials of degree n. Suppose the Arnoldi method does not break down when applied to A and started with b. Let $Q \in \mathbb{C}^{n\times m}$ be the orthogonal basis generated after m iterations. Then,

$$\|(I - QQ^*)x_i\| \leq \xi_i \varepsilon_i^{(m)}, \tag{1.14}$$

The indicator can be bounded by a product consisting of two scalar values: $\varepsilon_i^{(m)}$ which only depends on the eigenvalues and iteration number; and ξ_i only depending on the starting vector and eigenvectors.

where

$$\xi_i = \sum_{\substack{j=1 \\ j\neq i}}^{n} \frac{|\alpha_j|}{|\alpha_i|}.$$

Proof. The proof consists of three steps.

1. Consider any vector $u \in \mathbb{C}^n$. Then

$$\min_{z \in \mathbb{C}^m} \|u - Qz\|_2$$

is a linear least squares problem with a solution given by the normal equations $Q^* u = Q^* Q z$. Hence, $z = Q^* u$. This implies that (for any vector u) we have

$$\min_{z \in \mathbb{C}^m} \|u - Qz\|_2 = \|u - QQ^* u\| = \|(I - QQ^*)u\|$$

2. Although we ultimately want to bound the left-hand side of (1.14), the proof is simplified by considerations of a scaling the left-hand side of (1.14) with α_i as follows:

Apply step 1 reversely with $u = \alpha_i x_i$

$$\begin{aligned} \|(I - QQ^*)\alpha_i x_i\| &= \min_{z \in \mathbb{C}^m} \|\alpha_i x_i - Qz\| \\ &= \min_{y \in \mathcal{K}_m(A,b)} \|\alpha_i x_i - y\| \end{aligned}$$

Now note that the space $\mathcal{K}_m(A, b)$ can be characterized with polynomials. It is easy to verify that $y \in \mathcal{K}_m(A, b)$ is equivalent to the existance of a polynomial $p \in P_{m-1}$ such that $y = p(A)b$. Consequently,

$$\|(I - QQ^*)\alpha_i x_i\| = \min_{p \in P_{m-1}} \|\alpha_i x_i - p(A)b\|.$$

3. The final step consists of inserting the expansion of b in terms of eigenvectors (1.13) and applying appropriate bounds:

$$\begin{aligned}
\|(I-QQ^*)\alpha_i x_i\| &= \min_{p\in P_{m-1}} \|\alpha_i x_i - p(A)\sum_{j=1}^{n}\alpha_j x_j\| \\
&= \min_{p\in P_{m-1}} \left\|\alpha_i x_i - \sum_{j=1}^{n}\alpha_j p(\lambda_j)x_j\right\| && \text{Since } x_i \text{ eigenvector, } p(A)x_i = p(\lambda_i)x_i \\
&\leq \min_{\substack{p\in P_{m-1}\\ p(\lambda_i)=1}} \left\|\alpha_i x_i - \sum_{j=1}^{n}\alpha_j p(\lambda_j)x_j\right\| && \text{For any two sets } S\subset Z: \\
&&& \min_{z\in Z} g(z) \leq \min_{z\in S} g(z) \\
&= \min_{\substack{p\in P_{m-1}\\ p(\lambda_i)=1}} \|\alpha_i x_i - \alpha_i x_i - \sum_{\substack{j=1\\ j\neq i}}^{n}\alpha_j p(\lambda_j)x_j\| \\
&= \min_{\substack{p\in P_{m-1}\\ p(\lambda_i)=1}} \|\sum_{\substack{j=1\\ j\neq i}}^{n}\alpha_j p(\lambda_j)x_j\| \\
&\leq (\sum_{\substack{j=1\\ j\neq i}}^{n}|\alpha_j|)\cdot \min_{\substack{p\in P_{m-1}\\ p(\lambda_i)=1}} \max_{j\neq i}(|p(\lambda_j)|) \\
&= (\sum_{\substack{j=1\\ j\neq i}}^{n}|\alpha_j|)\cdot \varepsilon_i^{(m)}
\end{aligned}$$

The conclusion of the theorem is established by dividing the equation by $|\alpha_i|$.

□

Note that $\|b\| = 1$ and $\|x_1\| = \cdots = \|x_n\| = 1$. Hence the coefficients $\alpha_1, \ldots, \alpha_n$ are balanced. In particular they satisfy

$$1 = \|\alpha_1 x_1 + \cdots + \alpha_n x_n\| \leq |\alpha_1| + \cdots + |\alpha_n|.$$

and

$$\xi_i = \frac{1}{|\alpha_i|}\sum_{j=1}^{n}|\alpha_j| - 1 \geq \frac{1}{|\alpha_i|} - 1$$

From this we can easily identify a very good situation and a very bad situation.

- Suppose for all $j \neq i$, $\alpha_j = \delta$ and suppose δ is small. We have that $\xi_i = \frac{(n-1)\delta}{\alpha_i}$. Due to balancing α_i cannot be small. Hence, ξ_i is small, showing fast convergence for this eigenvalue.
- On the other hand, if α_i (the component of the starting vector in the direction of the ith eigenvector) is very small, we have $\xi_i \gg 1$ which implies that the right-hand side of (1.14) is large and we have slow convergence.

This serves as a justification for a more general property.

Rule-of-thumb. Starting vector dependency. The Arnoldi method for eigenvalue problems will "favor" eigenvectors which have large components in the starting vector.

The word "favors" is purposely vague. It should be interpreted as the situation that one observes often in practice, but certainly not always. If we have a particular structure in the matrix or starting vector, we might observe convergence to other eigenvalues.

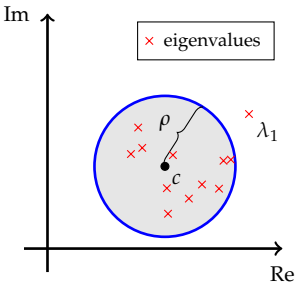
Bounding $\varepsilon_i^{(m)}$

In the characterization of the indicator in Theorem 1.4.2 above we introduced the quantity $\varepsilon_i^{(m)}$. This quantity bounds (up to a constant) the error in eigenvector x_i at iteration m. Although $\varepsilon_i^{(m)}$ is defined through a polynomial optimization problem, which is complicated to solve, it is surprisingly easy to use this to obtain bounds providing qualitative understanding of the convergence of the Arnoldi method for eigenvalue problems. We illustrate the power with a specific bound.

Think: $\varepsilon_i^{(m)}$ measures how "difficult" it is to push down a polynomial in points λ_j, for all $j \neq i$ and maintain $p(\lambda_i) = 1$.

Corollary 1.4.3. *Suppose $C(\rho, c) \subset \mathbb{C}$ is a disk centered at $c \in \mathbb{C}$ with radius ρ such that it contains all eigenvalues but λ_1. That is, $\lambda_2, \ldots, \lambda_n \in C(\rho, c)$ and $\lambda_1 \notin C(\rho, c)$. Then,*

$$\varepsilon_1^{(m)} \leq \left(\frac{\rho}{|\lambda_1 - c|} \right)^{m-1}.$$



Proof. The proof consists of selecting a particular polynomial in the polynomial optimization problem,

$$\begin{aligned} \varepsilon_1^{(m)} &:= \min_{\substack{p \in P_{m-1} \\ p(\lambda_1)=1}} \max(|p(\lambda_1)|, \ldots, |p(\lambda_{i-1})|, |p(\lambda_{i+1})|, \ldots, |p(\lambda_n)|) \\ &= \max_{j \neq i} |q(\lambda_j)|, \end{aligned}$$

for any $q \in P_{m-1}$ satisfying $q(\lambda_1) = 1$, in particular

$$q(z) = \frac{1}{(\lambda_1 - c)^{m-1}} (z - c)^{m-1}.$$

Hence, from the definition of ρ and c we have that

$$\begin{aligned} \varepsilon_1^{(m)} &\leq \max_{i>1} \frac{|\lambda_i - c|^{m-1}}{|\lambda_1 - c|^{m-1}} \\ &\leq \frac{\rho^{m-1}}{|\lambda_1 - c|^{m-1}}. \end{aligned}$$

□

The result can be intuitively interpreted as follows. If we can construct a small disc that encloses all eigenvalues but one eigenvalue we expect fast (at least linear geometric) convergence for that eigenvalue. This can be achieved for an eigenvalue which is well separated from the rest of the eigenvalues and also in an outer part of the spectrum. We call this "extreme" isolated eigenvalues.

Rule-of-thumb. Eigenvalue dependency. Arnoldi's method for eigenvalue problems favors convergence to "extreme" isolated eigenvalues.

Note the difference between an "extreme" eigenvalue and the eigenvalues which are largest in modulus (absolute value). The Arnoldi method will favor "extreme" whereas the power method will essentially always converge to the eigenvalue largest in modulus.

1.4.2 *An a posteriori theorem*

In the previous section we saw a characterization of the error involving the eigenvectors and eigenvalues of the matrix A. The following result provides an explicit characterization of $\|Av - \mu v\|$ where (μ, v) is an approximate eigenpair. It is expressed in terms of quantities computed during the iteration.

A priori vs. a posteriori: Error characterizations can be classified into two types. An *a priori* (latin for "from before") error estimate involves quantities which are known before the algorithm is carried out. An *a posteriori* (latin for "from after") error characterization involves quantites computed during the iteration. Theorem 1.4.2 is an a priori error bound. Theorem 1.4.4 is an (exact) a posteriori error characterization since the right-hand side involves H_k and z which are computed from the iteration.

Theorem 1.4.4. *Suppose Q_k and $\underline{H}_k$ satisfy the Arnoldi relation*

$$AQ_k = Q_{k+1}\underline{H}_k \tag{1.15}$$

where $Q_k \in \mathbb{C}^{n\times k}$ and $Q_{k+1} = [Q_k, q_{k+1}] \in \mathbb{C}^{n\times k}$ are orthogonal matrices. Moreover, suppose (μ, v) is a Ritz pair such that $H_k z = \mu z$ and $v = Q_k z$. Then,

$$\|Av - \mu v\|_2 = |h_{k+1,k}||e_k^T z|. \tag{1.16}$$

Proof. From the fact that (μ, v) is a Ritz pair, we have

$$\begin{aligned} Av - \mu v &= AQ_k z - \mu Q_k z \\ &= (AQ_k - Q_k H_k)z \\ &= h_{k+1,k} q_{k+1} e_k^T z \end{aligned}$$

Use $v = Q_k z$.

Use that since $\underline{H}_k$ is a Hessenberg matrix, (1.15) can be written as $AQ_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T$.

The conclusion follows from the fact that $e_k^T z$ is a scalar and q_{k+1} is normalized since Q_{k+1} is orthogonal. More precisely, $\|Av - \mu v\|_2 = |h_{k+1,k}|\|q_{k+1} e_k^T z\| = |h_{k+1,k}|\|q_{k+1}\||e_k^T z| = |h_{k+1,k}||e_k^T z|$. □

The result can be used to study break-down. Break-down corresponds to the situation where we cannot carry out that Gram-Schmidt orthogonalization process since the new vector is contained in the span

of previous iterations. It implies that the $y_\perp = 0$ and $\beta = 0$. This implies in turn that $h_{k+1,k} = 0$. Hence, due to (1.16), if we have breakdown the error is already zero and the Ritz pairs are eigenpairs of the original problem.

1.5 *Literature and further reading*

The proof and reasoning above is inspired by [5]. Other convergence bounds involving Schur factorizations, that lead to similar qualitative understanding can be found in [6], where also complications of the non-generic cases are discussed. There are also further characterizations of convergence and the connection with potential theory [4]. In the above reasoning we characterized the angle between the subspace and the eigenvector. Although this serves as a very accurate prediction of the error in practice, it does not directly give a rigorous bound on the accuracy of Ritz pair. Several approaches to describe the convergence of Ritz values and Ritz vectors have been done in for instance [2, 3]. There is also considerable research on the effect of rounding errors in Krylov methods. Unlike many other numerical methods, the effect of finite arithmetic can improve the performance of the algorithm. See also the recent summary of the convergence of the Arnoldi method for eigenvalue problems [1]. The a posteriori error estimate in Theorem 1.4.4 is contained in some recent text-books in numerical linear algebra such as [7].

1.6 *References*

[1] M. Bellalij, Y. Saad, and H. Sadok. Further analysis of the Arnoldi process for eigenvalue problems. *SIAM J. Numer. Anal.*, 48(2):393–407, 2010.

[2] Z. Jia. The convergence of generalized Lanczos methods for large unsymmetric eigenproblems. *SIAM J. Matrix Anal. Appl.*, 16(3):843–862, 1995.

[3] Z. Jia and G. W. Stewart. On the convergence of ritz values, ritz vectors, and refined ritz vectors. Technical report, 1999.

[4] A. B. Kuijlaars. Convergence analysis of Krylov subspace iterations with methods from potential theory. *SIAM Rev.*, 48(1):3–40, 2006.

[5] Y. Saad. *Numerical methods for large eigenvalue problems*. SIAM, 2011.

[6] G.W. Stewart. *Matrix Algorithms volume 2: eigensystems*. SIAM publications, 2001.

[7] D. S. Watkins. *Fundamentals of matrix computations. 3rd ed*. Wiley, 2010.

