Error control in scientific modelling (MATH 500, Herbst)

Sheet 3: Matrix eigenproblems (10 P)

To be handed in via moodle by 12.10.2023

Exercise 1 (1 + 1.5 + 0.5 P)

We saw in the lecture that if $A \in \mathbb{C}^{n \times n}$ is Hermitian (i.e. $A = A^H$, then the Rayleigh quotient

$$R_A(x) = \frac{\langle x, Ax \rangle}{\langle x, x \rangle}$$

is real for all $x \in \mathbb{C}^n$. The point of this exercise is to prove the converse, i.e.

> If Rayleigh quotient $R_A(x)$ is real for all $x \in \mathbb{C}^n$, then A is a Hermitian matrix.

To show this we proceed as follows:

(a) Given an arbitrary matrix S, show that if $\langle x, Sx \rangle = 0$ for all $x \in \mathbb{C}^n$, then we must have

$$\langle y, Sz \rangle + \langle z, Sy \rangle = 0 \qquad \forall y, z \in \mathbb{C}^n$$

Hint: Expand $\langle (y + z), S(y + z) \rangle$.

(b) Use the result of (a) to show if $\langle x, Ax \rangle$ is real for all $x \in \mathbb{C}^n$, then A must be Hermitian. *Hint:* First relate $\langle x, Ax \rangle$ to $\langle x, A^H x \rangle$, then use (a)

(c) Prove the above statement, i.e. if $R_A(x)$ is real for all $x \in \mathbb{C}^n$, then A is Hermitian.

Selection deleted

(a) **Solution:**

Let's consider arbitrary vectors $y, z \in \mathbb{C}^n$. By the property of the vector space, $(y+z)$ is also an element of the same vector space $\mathbb{C}^n$.

Given that $\langle x, Sx \rangle = 0$ for all $x \in \mathbb{C}^n$ it is also true that:

$$\langle z, Sz \rangle = 0,$$

$$\langle y, Sy \rangle = 0,$$

$$\langle (y+z), S(y+z) \rangle = 0$$

since $(y+z) \in \mathbb{C}^n$.

Exploiting the linearity property of the inner product we have:

$$\begin{aligned} 0 &= \langle (y+z), S(y+z) \rangle = \langle y, S(y+z) \rangle + \langle z, S(y+z) \rangle \\ &= \langle y, Sy \rangle + \langle z, Sz \rangle + \langle y, Sz \rangle + \langle z, Sy \rangle \\ &= \langle y, Sz \rangle + \langle z, Sy \rangle \qquad \forall y, z \in \mathbb{C}^n \end{aligned} \qquad \square$$

Selection deleted

(b) **Solution:**

The fact that $\langle x, Ax\rangle$ is real means that it must be equal to its complex conjugate:

$$\langle x, Ax\rangle = \overline{\langle x, Ax\rangle} = \langle Ax, x\rangle = \langle x, A^H x\rangle$$

Applying the linearity property of the inner product:

$$\langle x, Ax\rangle - \langle x, A^H x\rangle = \langle x, (A - A^H)x\rangle = 0.$$

Using (a):

$$\begin{aligned} 0 &= \langle y, (A - A^H)z\rangle + \langle z, (A - A^H)y\rangle = \\ &= \langle (A - A^H)^H y, z\rangle + \langle z, (A - A^H)y\rangle = \\ &= -\langle (A - A^H)y, z\rangle + \langle z, (A - A^H)y\rangle. \end{aligned}$$

Therefore,

$$\langle (A - A^H)y, z\rangle = \langle z, (A - A^H)y\rangle \qquad (\dagger)$$

Now one can replace z with iz, as both are in $\mathbb{C}^n$, let's now expand both sides:

$$\begin{aligned} \langle (A - A^H)y, iz\rangle =& \langle iz, (A - A^H)y\rangle \quad \Longrightarrow \\ \bar{i}\langle (A - A^H)y, z\rangle =& i\langle z, (A - A^H)y\rangle \quad \Longrightarrow \\ -\langle (A - A^H)y, z\rangle =& \langle z, (A - A^H)y\rangle \qquad (\ddagger) \end{aligned}$$

($\dagger$) and ($\ddagger$) can only be both true only if

$$\langle (A - A^H)y, z\rangle = \langle z, (A - A^H)y\rangle = 0 \quad \forall y, z \in \mathbb{C}^n$$

.

Which is possible if and only if $A = A^H$. In other words, matrix A should be Hermitian.

□

(c) **Solution:**

For any $x \in \mathbb{C}^n$, inner product $\langle x, x\rangle = x\bar{x} \in \mathbb{R}$. Therefore, Rayleigh quotient $R_A(x)$ is real for all $x \in \mathbb{C}^n$ if and only if $\langle x, Ax\rangle$ is real $\forall x \in \mathbb{C}^n$. The result obtained from (b) completes the proof.

□

Exercise 2 (1.5+1+0.5 P)

Recall that the Frobenius norm of a matrix $A \in \mathbb{C}^{n\times n}$ is given by

$$\|A\|_F = \sqrt{\mathrm{tr}(A^H A)} = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} |A_{ij}|^2}$$

(a) What is the Frobenius norm of a diagonal matrix? What is the p-norm of a diagonal matrix? Conclude whether the Frobenius norm can be associated to any vector p-norm?

(b) Show that

$$\|A\|_2 \leq \|A\|_F$$

and use this to conclude

$$R_A(x) \leq \|A\|_F. \qquad (*)$$

(c) Keeping in mind our corollary of Courant-Fisher, namely that

$$\lambda_n = \max_{0\neq x\in\mathbb{C}} R_A(x)$$

as well as your result from (a) on diagonal matrices, argue why $(*)$ is a crude bound, in particular for large matrices.

(a) Solution:

i) Frobenious norm of a diagonal matrix.

Consider a diagonal matrix $D \in \mathbb{C}^{n \times n}$ with diagonal elements $\{d_i\}_{i=1}^n$:

$$D = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_n \end{bmatrix}.$$

Then, Frobenius norm of D is defined as following:

$$\|D\|_F = \sqrt{\mathrm{tr}(D^H D)} = \sqrt{\sum_{i=1}^n |d_i|^2} = \sqrt{\sum_{i=1}^n \sigma_i^2}$$

ii) The p-norm of a diagonal matrix.

Recalling the definition of the p-norm of a matrix:

$$\|D\|_p = \max_{0 \neq x \in \mathbb{C}^n} \frac{\|Dx\|_p}{\|x\|_p} = \max_{\|x\|_p = 1} \|Dx\|_p = \max_{\|x\|_p = 1} \left(\sum_{i=1}^n |d_i x_i|^p \right)^{1/p} = |d_{max}| = \sigma_{max}(D),$$

where $\sigma_{max}(D)$ is the biggest singular value of the matrix D.

Now it's visible that it's not that easy to associate the Frobenius norm to any vector p-norm.

On the other hand, given a matrix partition into column vectors:

$$A = [a_1 \quad a_2 \quad \cdots \quad a_n],$$

where $a_i = [a_{1i} \quad a_{2i} \quad \cdots \quad a_{ni}]^T$, the **vectorization** of the matrix can be defined as following:

$$vec(A) = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \in \mathbb{C}^{n \cdot n}.$$

Then

$$\|A\|_F = \|vec(A)\|_2.$$

(b) Solution:

As we know:

$$\|A\|_2 = \max_{0 \neq x \in \mathbb{C}^n} \frac{\|Ax\|_2}{\|x\|_2} = \sqrt{\lambda_{\max}(A^H A)}$$

On the other hand,

$$\|A\|_F = \sqrt{\operatorname{tr}(A^H A)} = \sqrt{\sum_{i=1}^{n} \lambda_i(A^H A)}$$

Matrix $A^H A$ is Hermitian since $(A^H A)^H = A^H A$. Which implies that its' eigenvalues are real. Moreover, it is a positive semi-definite matrix, since:

$$\forall x \in \mathbb{C}^n: \quad x^H A^H A x = (Ax)^H A x = \|Ax\|^2 \geq 0$$

.

Therefore, eigenvalues of $A^H A$ are non-negative. Thus,

$$\|A\|_2 = \sqrt{\lambda_{\max}(A^H A)} \leq \sqrt{\sum_{i=1}^{n} \lambda_i(A^H A)} = \|A\|_F$$

Using Cauchy-Schwartz inequality we obtain:

$$|R_A(x)| = \frac{|\langle x, Ax \rangle|}{\langle x, x \rangle} \leq \frac{\|x\|_2 \|Ax\|_2}{\|x\|_2^2} \leq \frac{\|A\|_2 \|x\|_2}{\|x\|_2} = \|A\|_2,$$

and using the results above, we can conclude:

$$R_A(x) \leq \|A\|_F.$$

(c) Solution:

As we remember:

$$|R_A(x)| = \frac{|\langle x, Ax\rangle|}{\langle x, x\rangle} \leq \sqrt{\sum_{i=1}^{n} \lambda_i(A^H A)} = \|A\|_F.$$

This shows that it is a rough estimate of the Rayleigh quotient especially for large matrices since with growing n the sum $\sum_{i=1}^{n} \lambda_i(A^H A)$ will also grow. This may lead to a larger discrepancy between the upper bound given by the Frobenius norm and the true maximum eigenvalue.

Exercise 3 (2+1+1 P)

In the lectures we saw that minimising the Rayleigh quotient provides a numerical tool for computing eigenvalues via optimisation problems. But even beyond that setting the Rayleigh quotient can be interpreted as a tool to obtain an approximation for the eigenvalue corresponding to an approximate eigenvector. We will explore this in this exercise.

We consider the setting where we want to compute the eigenpair $(\lambda, v) \in \mathbb{R} \times \mathbb{C}^n$ of the Hermitian matrix $A \in \mathbb{C}^{n\times n}$. Employing some numerical scheme we do not get the exact eigenvector v but only the approximation $u = v + \delta$, where $\delta \in \mathbb{C}^n$ is the error compared to v. As usual we take v to be normalised. Moreover δ is always orthogonal to v (Why?).

(a) To simplify our calculations in this exercise, we introduce

$$t = \|\delta\| \in \mathbb{R} \quad d = \delta/t \in \mathbb{C}^n,$$

such that we can write $u = v + td$. Note that both v and d are unit vectors. The Rayleigh quotient $R_A(u)$ provides an approximation to λ. Show that

$$R_A(u) = \lambda + t^2\left(\langle d, Ad\rangle - \lambda\right) + O(t^4).$$

(b) Assume now a numerical scheme (e.g. power iteration) yields an approximation u to the exact eigenvector v, which is accurate to a tolerance ε, i.e. $\|\delta\| = \varepsilon$. We want to estimate the corresponding eigenvalue using $R_A(u)$. How does the error between this estimate and the true eigenvalue λ scale with ε?

(c) Reconsider your power iteration implementation from Sheet 1. Extend it, such that it employs the iterated eigenvector $x^{(i)}$ as well as the Rayleigh quotient to estimate the eigenvalue at each step. For the procedure computing the largest eigenvalue of the matrix

$$A = \begin{pmatrix} 30\,000 & -10\,000 & 10\,000 \\ -10\,000 & -30\,000 & 0 \\ 10\,000 & 0 & 1 \end{pmatrix}$$

record both the approximate eigenvalue as well as the approximate eigenvector in each iteration in two separate arrays. Use this data to plot the error in the approximate eigenvalue as well as the error in the eigenvector as the iteration proceeds. You should find numerical confirmation to your analysis of (a) and (b).

Some hints:

- For computing the eigenvalue error just take the modulus of the absolute error, for the eigenvector error take the l_2-norm (`norm` function in Julia).
- You can compute the exact eigenvalue and eigenvector using the `eigen` routine of Julia. Note, however, that (real) eigenvectors are only determined up to the sign, so you have to ensure that the same sign convention is used in `eigen` as well as your own algorithm. The easiest is to determine the sign of the first element of the vector returned by your routine as well as `eigen` and multiply one of the vectors by `-1` in case these differ.
- Usually it is best to employ a `log`-scale on the y-axis for such error plots.

(a) Answer

By definition:

$$R_A(x) = \frac{\langle x, Ax\rangle}{\langle x, x\rangle}$$

and thus replacing x with $u = v + td$ and using that $\langle v, d\rangle = 0$ due to orthogonality:

$$\begin{aligned} R_A(u) &= \frac{\langle v+td, A(v+td)\rangle}{\langle v+td, v+td\rangle} = \frac{\langle v, Av\rangle}{\langle v,v\rangle + \langle td, td\rangle} + \frac{\langle td, Atd\rangle}{\langle v,v\rangle + \langle td, td\rangle} = \\ &= \frac{\lambda\langle v,v\rangle + \lambda(t^2\langle d,d\rangle - t^2\langle d,d\rangle)}{\langle v,v\rangle + t^2\langle d,d\rangle} + \frac{t^2\langle d, Ad\rangle}{\langle v,v\rangle + t^2\langle d,d\rangle} = \\ &= \lambda + \frac{t^2\left(-\lambda\langle d,d\rangle + \langle d, Ad\rangle\right)}{\langle v,v\rangle + t^2\langle d,d\rangle}. \end{aligned}$$

Using the fact that both v and d are unit vectors:

$$\langle v, v\rangle = \|v\|^2 = 1,$$

and $\langle d, d\rangle = \|d\|^2 = 1$.

Therefore,

$$\begin{aligned} R_A(u) &= \lambda + \frac{\left(t^2\langle d, Ad\rangle - t^2\lambda\right)}{1+t^2} = \\ &= \lambda + \frac{\left(1+t^2\right)\left(t^2\langle d, Ad\rangle - t^2\lambda\right) - t^4\left(-\lambda + \langle d, Ad\rangle\right)}{1+t^2} = \\ &= \lambda + t^2\left(\langle d, Ad\rangle - \lambda\right) - \frac{t^4\left(-\lambda + \langle d, Ad\rangle\right)}{1+t^2} = \\ &= \lambda + t^2\left(\langle d, Ad\rangle - \lambda\right) + O(t^4). \end{aligned}$$

(b) Answer

From exercise (a) it is known that the Rayleigh quotient approximation error scales with t^2, so if $t = ||\delta|| = \epsilon$, then the error scales with ϵ^2.

(c) Solution:

```julia
using LinearAlgebra, Plots
```

Rayleigh_quotient (generic function with 1 method)

```julia
function Rayleigh_quotient(A, u)
    return dot(u, A * u) / dot(u, u)
end
```

```julia
A = [30000. -10000. 10000.;
     -10000. -30000. 0.;
     10000. 0. 1.];
```

0.333333333334546

```julia
begin
    λ_exact, v_exact= eigen(A)
    r_quotient_exact = Rayleigh_quotient(A, v_exact)
end
```

34453.99905307134

```julia
λ_exact[3] # the biggest exact eigenvalue
```

power_method_with_RQ (generic function with 2 methods)

```julia
function power_method_with_RQ(A, x0=randn(eltype(A), size(A, 2)); tol=1e-8, maxiter=500)
    v_list = []
    λ_list = []

    x = x0
    for i in 1:maxiter
        xprev = x
        x = A * x

        x /= norm(x)

        λ = Rayleigh_quotient(A, x)
        push!(λ_list, λ)
        push!(v_list, x)

        if min(norm(x - xprev), norm(-x - xprev)) < tol
            break
        end
    end
    (; λ_list, v_list)
end
```

```julia
begin
    # Calculating eigen pair using power method
    e_values, e_vectors = power_method_with_RQ(A, maxiter=400);

     if v_exact[1, 3] * e_vectors[length(e_vectors)][1] < 0.
            e_vectors *= -1
     end
end;
```

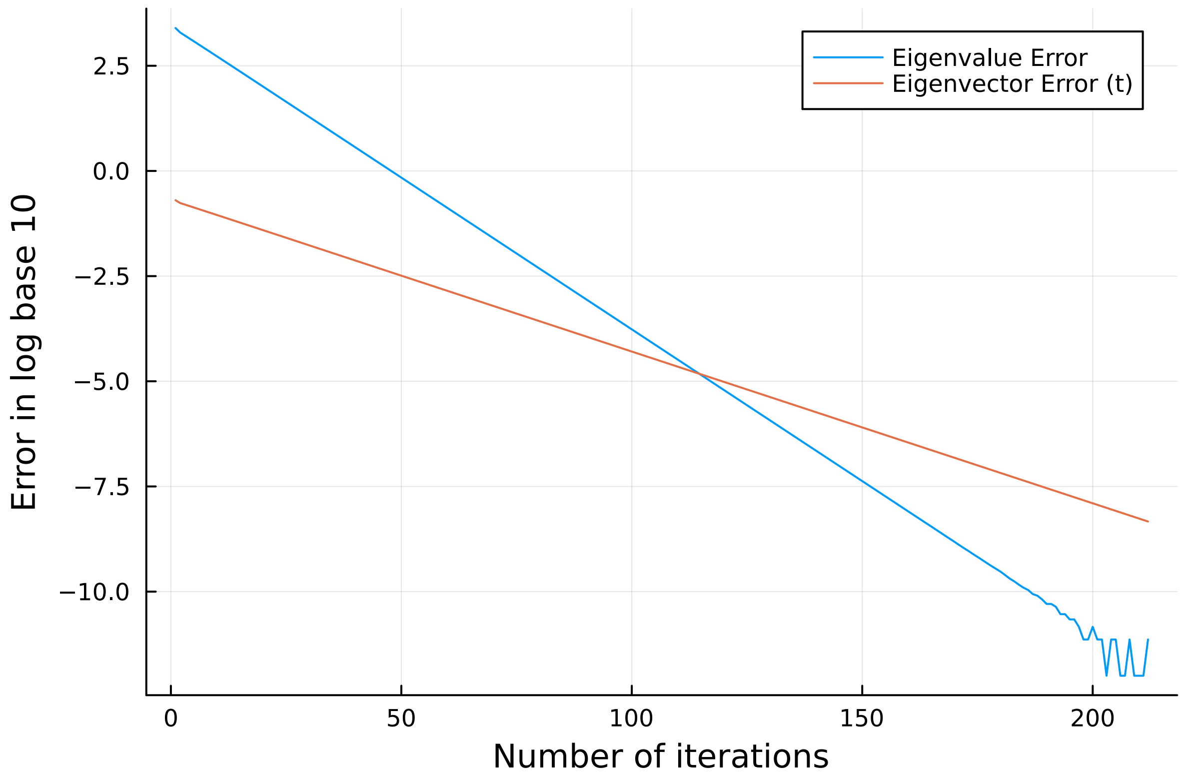
n_iterations = 212

```julia
n_iterations = length(e_values)
```

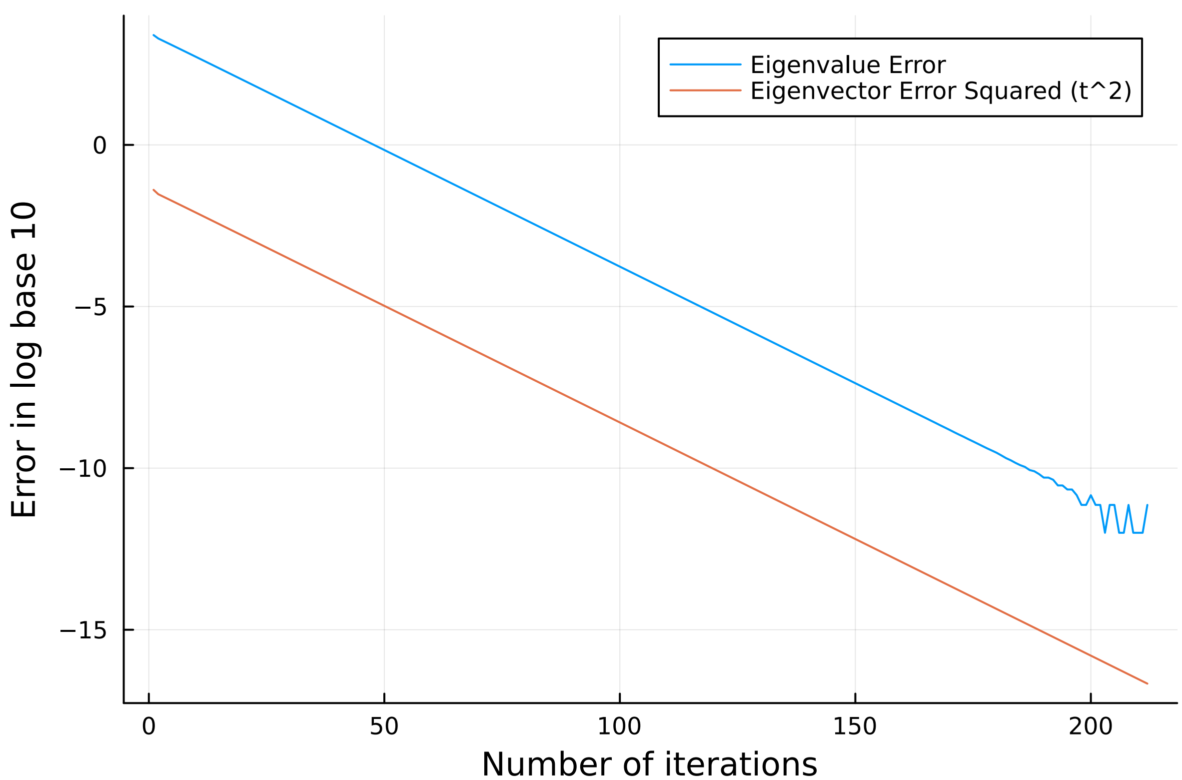
```julia
begin
    eigenvalue_errors = abs.(e_values .- λ_exact[3])
    eigenvector_errors = [norm(e_vectors[i] - v_exact[:, 3]) for i in 1:n_iterations]
end;
```

```julia
begin
    eigenvector_errors_shifted = [value == 0.0 ? 1.e-12 : value for value in
    eigenvector_errors]

    eigenvalue_errors_shifted = [value == 0.0 ? 1.e-12 : value for value in
    eigenvalue_errors]
end;
```




```
begin
    plot(1:n_iterations, log.(10,eigenvalue_errors_shifted),  ylabel="Error in log base 10",xlabel="Number of iterations", label="Eigenvalue Error", legend=:topright)
    plot!(1:n_iterations, log.(10,eigenvector_errors_shifted),  label="Eigenvector Error (t)")
end
```



Analyzing the plot obtained above we can easily see the result from (a) and (b), particularly, that the Rayleigh quotient approximation error scales with t^2.

```
# using EasyFit
```

```
# fit_μ= fitlinear(50:n_iterations,log.(10,eigenvalue_errors_shifted[50:end]))
```

```
# fit_v=fitlinear(50:n_iterations,log.(10,eigenvector_errors_shifted[50:end]))
```