Sheet 7: Resolvents and spectral projectors

```
begin
    using LinearAlgebra
    using PlutoUI
    using PlutoTeachingTools
    using Plots
end
```

Exercise 1

In this exercise we want to explore resolvents graphically. We consider the following 3x3 matrix:

```
3×3 Matrix{Float64}:
 -1.07177     -0.0132045   0.111385
 -0.0132045  -0.0182637  -0.017395
  0.111385   -0.017395    0.766584
```

```
begin
    A = Diagonal([-1, 0, 1]) + 0.2randn(3, 3)
    A = (A + A') / 2
end
```

The following function defines the resolvent $R_z(A)$ for various values of $z \in \mathbb{C}$:

```
resolvent (generic function with 1 method)
```

```
resolvent(z) = inv(A - z * I)
```

The resolvent is a complex-valued matrix.

(a) Compute the resolvent at $z = 0.0$, $z = i$ and $z = 1 + i$. For the latter also compute the eigenvalues.

(a) solution

```
3×3 Matrix{Float64}:
 -0.924346    0.52894    0.14631
  0.52894   -53.8977    -1.29988
  0.14631    -1.29988    1.25373
```

```
resolvent(0.0)
```

```
3×3 Matrix{ComplexF64}:
   -0.494939+0.462874im    -0.00617835-0.00562017im    0.059102+0.00971461im
 -0.00617835-0.00562017im   -0.0183337+0.999389im    -0.0111658+0.00892876im
    0.059102+0.00971461im   -0.0111658+0.00892876im    0.480153+0.625145im
```

```
resolvent(1.0im)
```

```
3×3 Matrix{ComplexF64}:
   -0.390039+0.190759im    0.000853634-0.00409669im    0.0105975+0.045933im
 0.000853634-0.00409669im    -0.499752+0.491018im    -0.00658736-0.0103259im
   0.0105975+0.045933im    -0.00658736-0.0103259im     -0.216047+0.948276im
```

```julia
resolvent(1.0 + 1.0im)
```

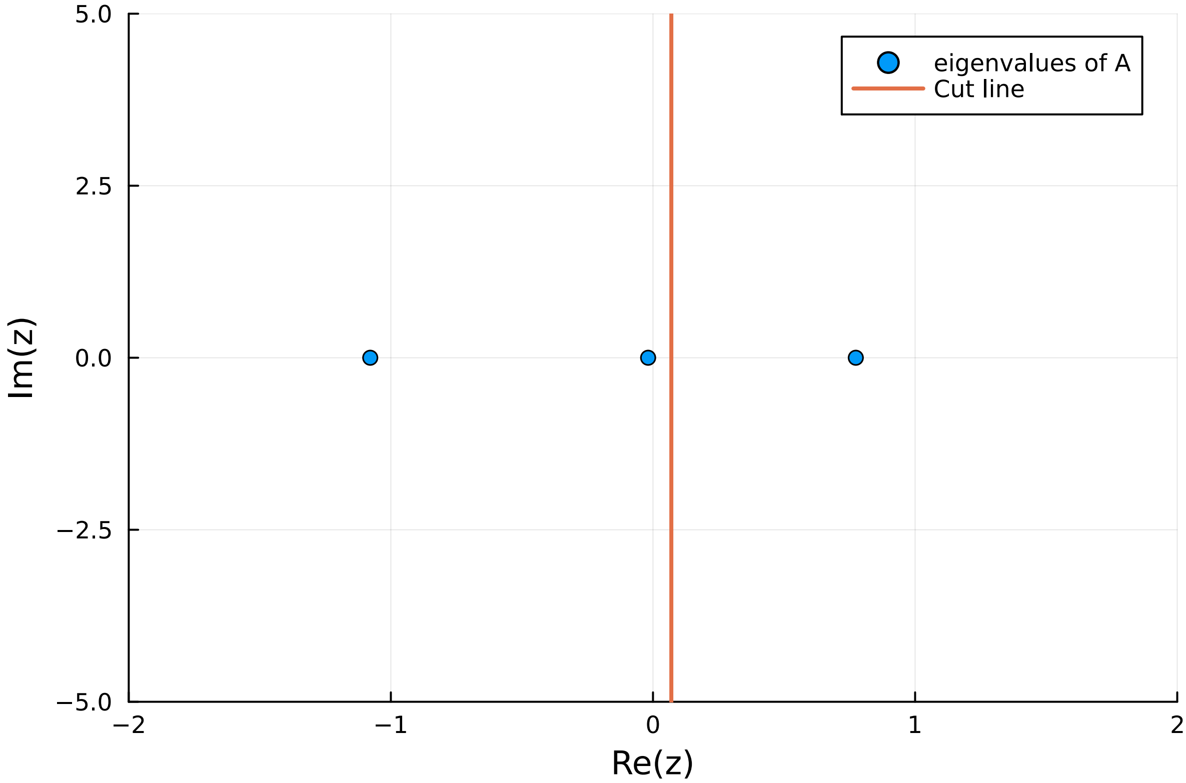
```
[-0.499916+0.490815im, -0.390668+0.187944im, -0.215254+0.951293im]
```

```julia
eigvals(resolvent(1.0 + 1.0im))
```

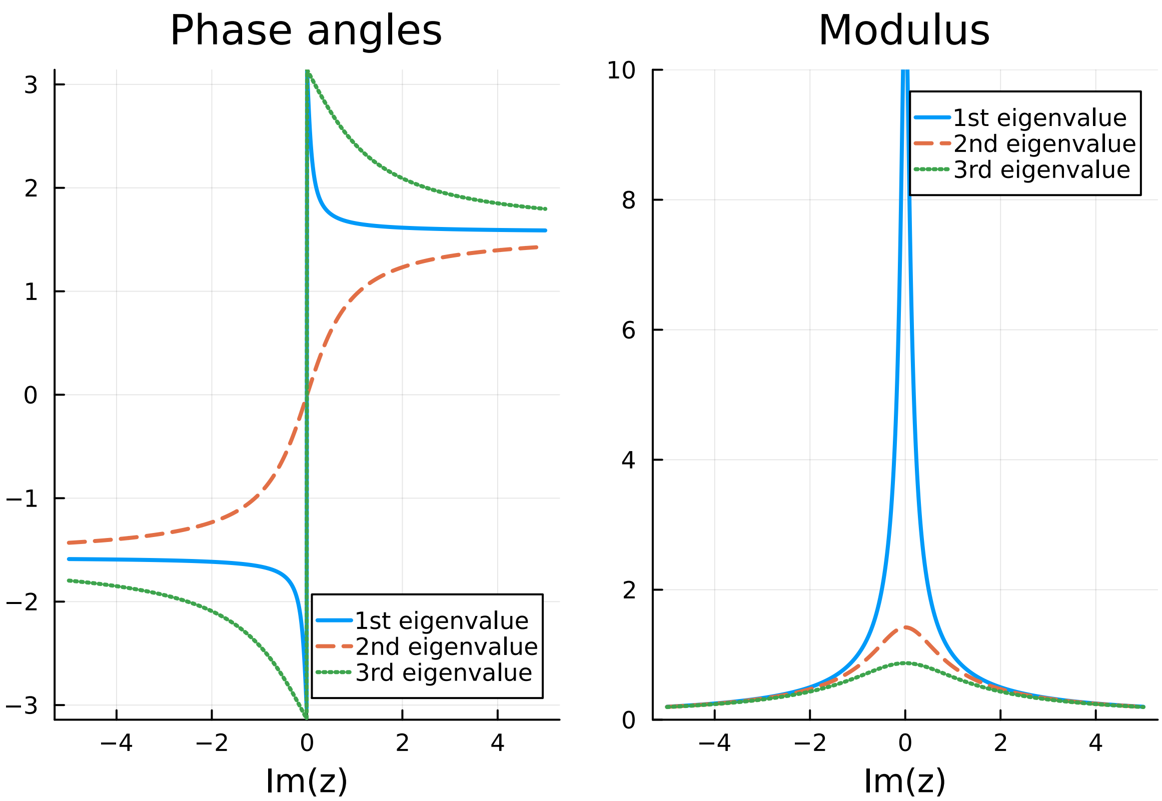
We will now explore how the resolvent varies with changing z.

We consider how the eigenvalues of $R_z(A)$ change as the **imaginary part** of z is varied. The following slider you can vary the real part of z:

- `Re(z) =` 0.07



As you vary the slider the following plot updates, showing the change of the phase angle and modulus of the *complex* eigenvalues of $R_z(A)$:




```julia
let
    ys = (-5:0.01:5) .+ 1e-6
    λs = zeros(ComplexF64, 3, length(ys))
    for i in 1:length(ys)
        z = x + im * ys[i]
        λ, X = eigen(resolvent(z))
        sidx = sort(1:3; by=i -> abs(X[1, i]))
        λs[:, i] = λ[sidx]
    end

    p = plot(ys, angle.(λs[1, :]), label="1st eigenvalue",
             lw=2, title="Phase angles", legend=:bottomright, ylims=(-π, π), xlabel="Im(z)")
    plot!(p, ys, angle.(λs[2, :]), label="2nd eigenvalue", ls=:dash, lw=2)
    plot!(p, ys, angle.(λs[3, :]), label="3rd eigenvalue", ls=:dot, lw=2)

    q = plot(ys, abs.(λs[1, :]), label="1st eigenvalue",
             lw=2, title="Modulus", xlabel="Im(z)")
    plot!(q, ys, abs.(λs[2, :]), label="2nd eigenvalue", ls=:dash, lw=2, ylims=(0, 10))
    plot!(q, ys, abs.(λs[3, :]), label="3rd eigenvalue", ls=:dot, lw=2)

    plot(p, q)
end
```

(b) Play with the slider. In particular note what happens with the modulus as you get close to an eigenvalue. Can you explain ? You should notice that each eigenvalue colour is associated to one eigenvalue in the first plot. What happens to the phase angle as you are sweeping over the corresponding eigenvalue ?

(b) solution

As we have seen in earlier lectures on spectral transformations, the eigenvalues of the resolvent $(A - zI)^{-1}$ are given by $\frac{1}{\lambda_i - z}$ in terms of the eigenvalues λ_i of A. This explains why the modulus of an associated eigenvalue explodes towards $+\infty$ as $z \to \lambda_i$ for some λ_i.

Exercise 2

In this exercise we want to numerically compute spectral projectors and show that they indeed reproduce the eigenspaces of a matrix, i.e. that Theorem 2 of the notes is correct.

We consider the matrix A introduced above.

Our goal will be to compute projectors of the form

$$P_i = -\frac{1}{2\pi i} \oint_{C_{\lambda_i}} R_z(A) dz$$

where C_{λ_i} is a contour around the eigenvalue λ_i. Our matrix A has the eigenvalues:

```
[-1.07863, -0.0185413, 0.773725]
```

```
eigvals(A)
```

We will consider computing the spectral projector of the first eigenvalue λ_1

```
λ1 = -1.0786349302945766
```

```
λ1 = minimum(eigvals(A))
```

with corresponding eigenvector

```
X1 =   [0.998124, 0.0114427, -0.060143]
```

```
X1 = let
    λ, X = eigen(A)
    imin = argmin(λ)
    X1 = X[:, imin]
end
```

To compute P_1 we first need to understand how to numerically compute such contour integrals. A simple idea is to consider a circular contour, i.e. one where

$$z(t) = z_0 + re^{2\pi i\, t}$$

for $t \in [0, 1]$, $r > 0$ a specified radius and z_0 the centre. Then a standard result in analysis tells us that

$$P_i = -\frac{1}{2\pi i} \oint_{C_{\lambda_i}} R_z(A) dz = -\frac{1}{2\pi i} \int_0^1 R_{z(t)}(A)\, z'(t) dt$$

where z' is the derivative wrt. t. Employing the trapezoidal rule for the last integral we divide t into N equally spaced subintervals with integration points $t_n = \frac{n}{N}$ where $n = 1, \ldots, N$. Then

$$\int_0^1 R_{z(t)}(A)\, z'(t) dt \approx \frac{1}{N} \sum_{n=1}^N R_{z(t_n)}(A) z'(t_n)$$

(a) Compute $z'(t)$ and implement a function `projector(centre, radius; N=30)` which computes the projector integral using a contour of `radius` centred around `centre` and using `N` points in the trapezoidal rule.

(a) solution

```
projector (generic function with 1 method)
```

```julia
function projector(centre, radius; N=30)
    accu = zeros(ComplexF64, size(A)...)

    z(t) = centre + radius * exp(2π * im * t)
    for n in 1:N
        t = n / N
        dz = radius * exp(2π * im * t) * 2π * im
        accu += resolvent(z(t)) * dz
    end
    (-1/(2π * im)) * accu / N  # Could also cancel the prefactor against dz above.
end
```

(b) Targeting the projector of λ_1, how do we need to choose z_0 and r ?

(b) solution

We can choose $z_0 = \lambda_1$ and the radius r must be chosen smaller than the gap to the rest of the spectrum.

(c) Compute the projector P_1 and verify that it is indeed a projector and that it keeps `X1` invariant.

(c) solution

```
P1 = 3×3 Matrix{ComplexF64}:
      0.996252-6.89101e-17im     0.0114212-7.46345e-19im     -0.0600302+5.10951e-18im
     0.0114212-8.38375e-19im   0.000130936+5.65051e-18im   -0.000688199+2.81151e-20im
    -0.0600302+3.26891e-18im  -0.000688199+2.81151e-20im     0.00361718-7.60649e-19im
```

```julia
P1 = projector(λ1, 0.1)
```

```
3×3 Matrix{ComplexF64}:
   0.996252-1.37825e-16im     0.0114212-1.47135e-18im     -0.0600302+9.29203e-18im
  0.0114212-1.56178e-18im   0.000130936-1.66585e-20im   -0.000688199+1.05425e-19im
 -0.0600302+7.45173e-18im  -0.000688199+7.88785e-20im     0.00361718-5.085e-19im
```

```julia
P1^2
```

```
[0.998124-6.90967e-17im, 0.0114427-7.73836e-19im, -0.060143+3.30885e-18im]
```

```julia
P1*X1
```

```
5.400965884114423e-16
```

```julia
norm(X1 - P1 * X1)
```

(d) Compute the other two projectors P_2 and P_3 and verify Theorem 1 of the notes.

```
P =
  [3×3 Matrix{ComplexF64}:                                                        , 3×3
      0.996252-6.89101e-17im      0.0114212-7.46345e-19im     -0.0600302+5.10951e-18im   0
```

```julia
P = [projector(λi, 0.1) for λi in eigvals(A)]
```

```julia
# idempotence
for i in 1:3
    @show norm(P[i]^2 - P[i])
end
```

```
norm(P[i] ^ 2 - P[i]) = 2.335255721356238e-16
norm(P[i] ^ 2 - P[i]) = 1.7205171771564808e-17
norm(P[i] ^ 2 - P[i]) = 1.1616538482465167e-16
```

```julia
# commutativity
for i in 1:3
    for j in (i+1):3
        @show norm(P[i] * P[j] - P[j] * P[i])
    end
end
```

```
norm(P[i] * P[j] - P[j] * P[i]) = 2.424490735266833e-18
norm(P[i] * P[j] - P[j] * P[i]) = 3.728581358427086e-17
norm(P[i] * P[j] - P[j] * P[i]) = 1.4470073262587477e-17
```

```
3.455345009537121e-16
```

```julia
# completeness
@show norm((P[1] + P[2] + P[3]) - I)
```

```
norm((P[1] + P[2] + P[3]) - I) = 3.455345009537121e-16
```

(e) If X_1 is the eigenvector corresponding to λ_1, then $X_1 X_1^H$ is a projector onto the eigenspace of λ_1 (Why ?). Verify that P_1 is equal to $X_1 X_1^H$, thus that P_1 indeed projects onto the eigenspace of λ_1.

```
3×3 Matrix{ComplexF64}:
   0.996252-6.89101e-17im      0.0114212-7.46345e-19im     -0.0600302+5.10951e-18im
  0.0114212-8.38375e-19im    0.000130936+5.65051e-18im   -0.000688199+2.81151e-20im
 -0.0600302+3.26891e-18im   -0.000688199+2.81151e-20im     0.00361718-7.60649e-19im
```

```julia
P1
```

```
3×3 Matrix{Float64}:
  0.996252     0.0114212    -0.0600302
  0.0114212    0.000130936  -0.000688199
 -0.0600302   -0.000688199   0.00361718
```

```julia
X1 * X1'
```

```
7.691287833879081e-16
```

```julia
norm(P1 - X1 * X1')
```

```
Enter cell code...
```