Applied Complex Analysis (2021)

1 Lecture 21: Orthogonal polynomials and singular integrals

This lecture we do the following:

1. Cauchy transforms of weighted orthogonal polynomials
   - Three-term recurrence and calculation
   - Hilbert transform of weighted orthogonal polynomials
   - Hilbert transform of weighted Chebyshev polynomials
2. Log transform of weighted *classical* orthogonal polynomials

1.1 Cauchy transforms of orthogonal polynomials

Given a family of orthogonal polynomials $p_k(x)$ with respect to the weight $w(x)$ on (a,b), we always know it satisfies a three-term recurrence:

$$
\begin{aligned}
xp_0(x) &= a_0 p_0(x) + b_0 p_1(x) \\
xp_k(x) &= c_k p_{k-1}(x) + a_k p_k(x) + b_k p_{k+1}(x)
\end{aligned}
$$

Consider now the Cauchy transform of the weighted orthogonal polynomial:

$$
C_k(z) := \mathcal{C}_{[a,b]}[p_k w](z) = \frac{1}{2\pi\mathrm{i}} \int_a^b \frac{p_k(x)w(x)}{x-z} \mathrm{d}x
$$

Theorem (Three-term recurrence Cauchy transform of weighted OPs) $C_k(z)$ satisfies the same recurrence relationship as $p_k(x)$ for $k = 1, 2, \ldots$:

$$
\begin{aligned}
zC_0(z) &= a_0 C_0(z) + b_0 C_1(z) - \frac{1}{2\pi\mathrm{i}} \int_a^b w(x)\mathrm{d}x \\
zC_k(z) &= c_k C_{k-1}(z) + a_k C_k(z) + b_k C_{k+1}(z)
\end{aligned}
$$

Proof

$$
\begin{aligned}
zC_k(z) &= \frac{1}{2\pi\mathrm{i}}\int_a^b \frac{zp_k(x)w(x)}{x-z}\mathrm{d}x = \frac{1}{2\pi\mathrm{i}}\int_a^b \frac{(z-x)p_k(x)w(x)}{x-z}\mathrm{d}x + \int_a^b \frac{xp_k(x)w(x)}{x-z}\mathrm{d}x \\
&= -\frac{1}{2\pi\mathrm{i}}\int_a^b p_k(x)w(x)\mathrm{d}x + \int_a^b \frac{(c_kp_{k-1}(x)+a_kp_k(x)+b_kp_{k+1}(x))w(x)}{x-z}\mathrm{d}x \\
&= -\frac{1}{2\pi\mathrm{i}}\int_a^b p_k(x)w(x)\mathrm{d}x + c_kC_{k-1}(z) + a_kC_k(z) + b_kC_{k+1}(z)
\end{aligned}
$$

when $k > 0$, the integral term disappears. ■

This gives a convenient way to calculate the Cauchy transforms: if we know $C_0(z) = \mathcal{C}w(z)$ and $\int_a^b w(x)\mathrm{d}x$, solve the lower triangular system:

$$
\begin{pmatrix}
1 & & & & \\
a_0 - z & b_0 & & & \\
c_1 & a_1 - z & b_1 & & \\
& c_2 & a_2 - z & b_2 & \\
& & c_3 & a_3 - z & \ddots \\
& & & \ddots & \ddots
\end{pmatrix}
\begin{pmatrix} C_0(z) \\ C_1(z) \\ C_2(z) \\ C_3(z) \\ \vdots \end{pmatrix}
=
\begin{pmatrix} C_0(z) \\ \frac{1}{2\pi\mathrm{i}}\int_a^b w(x)\mathrm{d}x \\ 0 \\ 0 \\ \vdots \end{pmatrix}
$$

Example (Chebyshev Cauchy transform)

Consider the Chebyshev case $w(x) = \frac{1}{\sqrt{1-x^2}}$, which satisfies $\int_{-1}^{1} w(x)\mathrm{d}x = \pi$. Recall that

$$C_0(z) = \mathcal{C}w(z) = \frac{\mathrm{i}}{2\sqrt{z-1}\sqrt{z+1}}$$

Further, we have

$$\begin{aligned} xT_0(x) &= T_1(x) \\ xT_k(x) &= \frac{T_{k-1}(x)}{2} + \frac{T_{k+1}(x)}{2} \end{aligned}$$

hence

$$\begin{aligned} zC_0(z) &= C_1(z) - \frac{1}{2\mathrm{i}} \\ zC_k(z) &= \frac{C_{k-1}(z)}{2} + \frac{C_{k+1}(z)}{2}. \end{aligned}$$

In other words, we want to solve

$$\begin{pmatrix} 1 & & & & \\ -z & 1 & & & \\ 1/2 & -z & 1/2 & & \\ & 1/2 & -z & 1/2 & \\ & & 1/2 & -z & \ddots \\ & & & \ddots & \ddots \end{pmatrix} \begin{pmatrix} C_0(z) \\ C_1(z) \\ C_2(z) \\ C_3(z) \\ \vdots \end{pmatrix} = \begin{pmatrix} \frac{\mathrm{i}}{2\sqrt{z-1}\sqrt{z+1}} \\ \frac{1}{2\mathrm{i}} \\ 0 \\ 0 \\ \vdots \end{pmatrix}$$

with forward substitution.

```
using ApproxFun, SingularIntegralEquations, LinearAlgebra, Plots,
ComplexPhasePortrait
x = Fun()
w = 1/sqrt(1-x^2)
z = 0.1+0.1im

n = 10
L = zeros(ComplexF64,n,n)
L[1,1] = 1
L[2,1] = -z
L[2,2] = 1
for k=3:n
    L[k,k-1] = -z
    L[k,k-2] = L[k,k] = 1/2
end

C = L \ [ im/(2sqrt(z-1)sqrt(z+1)); 1/(2im); zeros(n-2)]

T_5 = Fun(Chebyshev(), [zeros(5);1])
cauchy(T_5*w,z) , C[6]
```

```
(0.14734333381379638 - 0.26445831594251407im, 0.1473433381379644 -
0.26445
```

```
83159425141im)
```

Warning This fails for large n or large z:

```
x = Fun()
w = 1/sqrt(1-x^2)
z = 5+6im

n = 100
L = zeros(ComplexF64,n,n)
L[1,1] = 1
L[2,1] = -z
L[2,2] = 1
for k=3:n
    L[k,k-1] = -z
    L[k,k-2] = L[k,k] = 1/2
end
C = L \ [ im/(2sqrt(z-1)sqrt(z+1)); 1/(2im); zeros(n-2)]
T_2_0 = Fun(Chebyshev(), [zeros(20);1])

C[21], cauchy(T_2_0*w, z)
```

```
(-880764.1597963147 - 1.124546144433576e6im, 0.0 + 8.834874115176436
e-18im
)
```

Forward substitution is an unstable algorithm for calculating the $C_k(z)$ because the general solution of the linear recurrence (or linear difference equation) satisfied by the $C_k(z)$ is a superposition (or linear combination) of an exponentially growing solution and an exponentially decaying solution. For large k, the $C_k(z)$ calculated via forward recurrence will pick up the exponentially growing solution because of rounding errors. The computed values of the $C_k(z)$ will then grow exponentially for large k while the actual values the $C_k(z)$ are bounded. Instead, a stable algorithm (e.g., Miller's algorithm or Olver's algorithm) is used to compute the $C_k(z)$. A simple way to compute the C_k more stably is to drop the first row of the recurrence:

```
L[2:end,1:end-1]
C = L[2:end,1:end-1]\ [1/(2im); zeros(n-2)]
C[6]- cauchy(T_5*w, z)
```

```
-3.072062106493749e-17 + 4.368604850944066e-17im
```

This algorithm also becomes unstable for large z. For large z, we can compute the C_k using

$$C_k(z) = \frac{1}{2\pi \mathrm{i}} \int_a^b \frac{p_k(x)w(x)}{x-z} \mathrm{d}x = -\frac{1}{2\pi \mathrm{i} z} \int_a^b p_k(x)w(x) \sum_{n=0}^{\infty} \left(\frac{x}{z}\right)^n \mathrm{d}x = -\frac{1}{2\pi \mathrm{i}} \sum_{n=k}^{\infty} \frac{\mu_n}{z^{n+1}},$$

where μ_n is the n-th moment of the OP p_k:

$$\mu_n = \int_a^b x^n p_k(x) w(x) \mathrm{d}x.$$

1.2 Hilbert transform of weighted orthogonal polynomials

Now consider the Hilbert transform of weighted orthogonal polynomials:

$$H_k(x) = \mathcal{H}_{(a,b)}[p_k w](x) = \frac{1}{\pi} \int_a^b \frac{p_k(t) w(t)}{x - t} \mathrm{d}t$$

Just like Cauchy transforms, the Hilbert transforms have

Corollary (Hilbert transform recurrence)

$$xH_0(x) = a_0 H_0(x) + b_0 H_1(x) + \frac{1}{\pi} \int_a^b w(x) \mathrm{d}x$$

$$xH_k(x) = c_k H_{k-1}(x) + a_k H_k(x) + b_k H_{k+1}(x)$$

Proof Recall

$$\mathcal{C}^+ f(x) + \mathcal{C}^- f(x) = \mathrm{i}\mathcal{H} f(x)$$

Therefore, we have

$$C_k^+(x) + C_k^-(x) = \mathrm{i}\mathcal{H}[wp_k](x)$$

hence we have

$$\begin{aligned}
xH_0(x) &= -\mathrm{i}x(C_0^+(x) + C_0^-(x)) \\
&= -\mathrm{i}\left[a_0(C_0^+(x) + C_0^-(x)) + b_0(C_1^+(x) + C_1^-(x)) - \frac{1}{\pi\mathrm{i}}\int_a^b w(x)\mathrm{d}x\right] \\
&= a_0 H_0(x) + b_0 H_1(x) + \frac{1}{\pi}\int_a^b w(x)\mathrm{d}x
\end{aligned}$$

Other k follows by a similar argument.

■

1.2.1 Example 1: weighted Chebyshev T

For

$$H_k(x) := \mathcal{H}[T_k/\sqrt{1-x^2}](x) = \frac{1}{\pi}\int_{-1}^{1} \frac{T_k(t)}{(x-t)\sqrt{1-t^2}} \mathrm{d}t$$

The recurrence gives us

$$\begin{aligned} xH_0(x) &= H_1(x) + 1 \\ xH_k(x) &= \frac{H_{k-1}(x)}{2} + \frac{H_k(x)}{2} \end{aligned}$$

In this case, we have $H_0(x) = \mathcal{H}[w](x) = 0$. Therefore, we can rewrite this recurrence as

$$\begin{aligned} H_1(x) = -1,\ xH_1(x) &= \frac{H_2(x)}{2} \\ xH_k(x) &= \frac{H_{k-1}(x)}{2} + \frac{H_{k+1}(x)}{2} \end{aligned}$$

This is precisely the three-term recurrence satisfied by $-U_{k-1}$! We therefore have

$$H_k(x) = -U_{k-1}(x)$$

This gives a very easy way to compute Hilbert transforms: if

$$f(x) = \sum_{k=0}^{\infty} f_k T_k(x)$$

then

$$\mathcal{H}\left[\frac{f}{\sqrt{1-\diamond^2}}\right](x) = -\sum_{k=0}^{\infty} f_{k+1} U_k(x)$$

```julia
x = 0.1
T = Fun(Chebyshev(),[zeros(n);1])
H(f,x) = -hilbert(f,x) # Fix normalisation
H(w*T,x) , -Fun(Ultraspherical(1), [zeros(n-1);1])(x)
```

```
(-0.5608031061203765, -0.5608031061203765)
```

1.2.2 Example 2: weighted Chebyshev U

For

$$H_k(x) := \mathcal{H}[U_k\sqrt{1-x^2}](x) = \frac{1}{\pi}\int_{-1}^{1}\frac{U_k(t)\sqrt{1-t^2}}{x-t}\mathrm{d}t$$

The recurrence gives us

$$\begin{aligned}
xH_0(x) &= \frac{H_1(x)}{2} + 1/2 \\
xH_k(x) &= \frac{H_{k-1}(x)}{2} + \frac{H_k(x)}{2}
\end{aligned}$$

In this case, we have $H_0(x) = \mathcal{H}[w](x) = x$. Therefore, we can rewrite this recurrence as

$$\begin{aligned}
H_{-1}(x) &:= 1 \\
xH_{-1}(x) &= H_0(x) \\
xH_0(x) &= \frac{H_{-1}(x)}{2} + \frac{H_1(x)}{2} \\
xH_k(x) &= \frac{H_{k-1}(x)}{2} + \frac{H_{k+1}(x)}{2}
\end{aligned}$$

This is precisely the three-term recurrence satisfied by T_{k+1}! We therefore have

$$H_k(x) = T_{k+1}(x)$$

1.3 Log transforms of weighted orthogonal polynomials

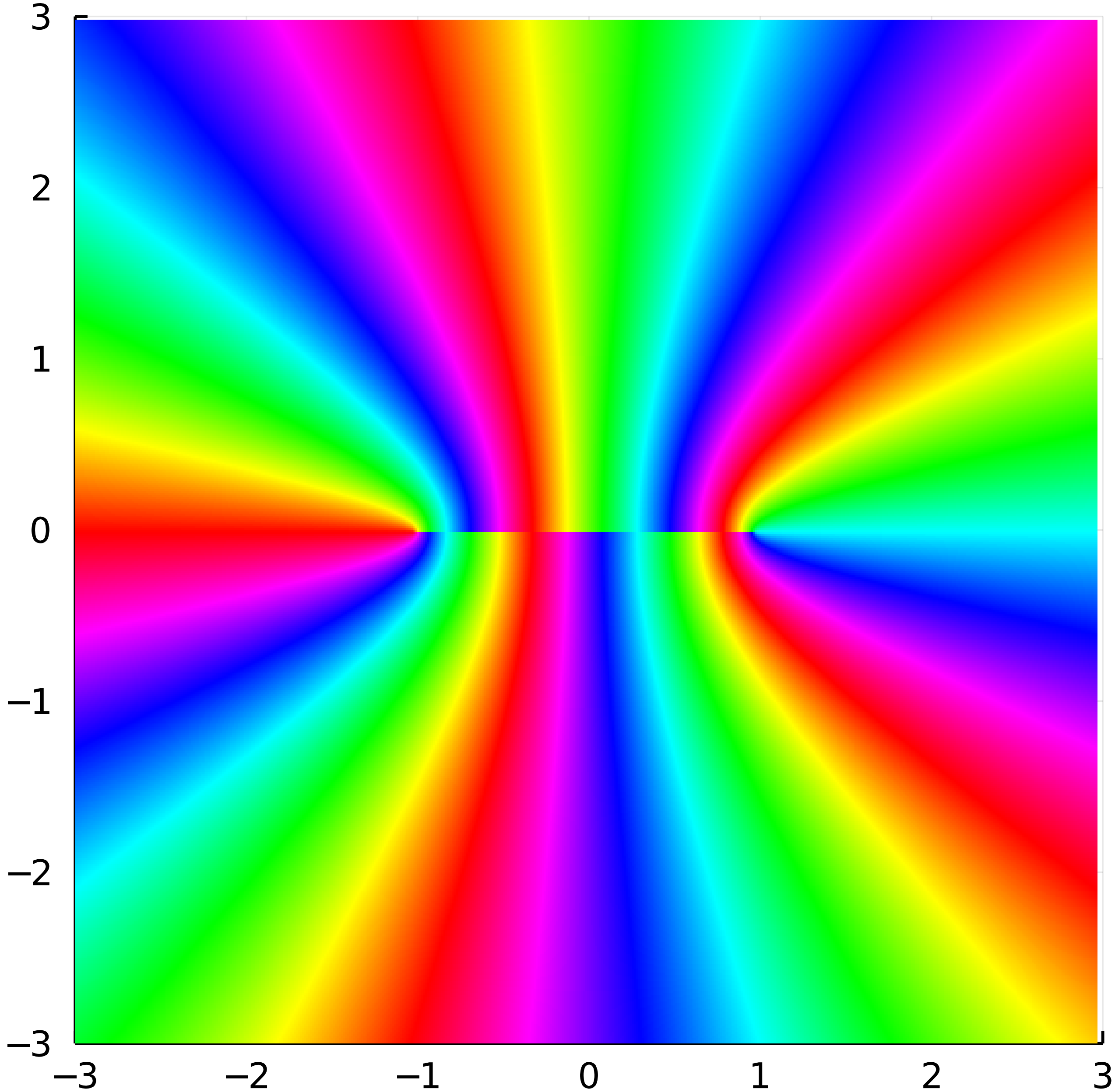
Now consider $\frac{1}{\pi}\int_a^b p_k(x)w(x)\log|z-x|\mathrm{d}x$, which we write in terms of the real part of

$$\begin{aligned}M_k(z) = M[p_k w](z) &= \frac{1}{\pi}\int_a^b p_k(x)w(x)\log(z-x)\mathrm{d}x \\ &= \frac{\log(z-a)}{\pi}\int_a^b p_k(x)w(x)\mathrm{d}x + 2\mathrm{i}\mathcal{C}_{[a,b]}F(z)\end{aligned}$$

where $F(x) = \int_x^1 p_k(t)w(t)\mathrm{d}t$. For $k>0$ we have $\int_a^b p_k(x)w(x)dx = 0$ due to orthogonality, and hence we actually have no branch cut on $(-\infty,a)$:

```
x = Fun()
T_5 = Fun(Chebyshev(), [zeros(5);1])
w = 1/sqrt(1-x^2)
M_5 = z-> -2im*cauchyintegral(w*T_5, z) # cauchyintegral computes
an indefinite integral of the Cauchy transform

phaseplot(-3..3, -3..3, M_5)
```


3
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2
3

1.4 Weighted Chebyshev log transform

For classical orthogonal polynomials we can go a step further and relate the indefinite integrals to other orthogonal polynomials.

For example, recall that

$$\frac{\mathrm{d}}{\mathrm{d}x}[\sqrt{1-x^2}U_n(x)] = -\frac{n+1}{\sqrt{1-x^2}}T_{n+1}(x)$$

in other words,

$$\int_x^1 \frac{T_k(t)}{\sqrt{1-t^2}}\mathrm{d}t = \frac{\sqrt{1-x^2}U_{k-1}(x)}{k}$$

Thus for $k = 1, 2, \ldots$,

$$M_k(z) = \frac{1}{\pi}\int_{-1}^1 \frac{T_k(x)}{\sqrt{1-x^2}}\log(z-x)\mathrm{d}x = \frac{2\mathrm{i}}{k}\mathcal{C}[\sqrt{1-\diamond^2}U_{k-1}](z)$$

and for $k = 0$,

$$M_0(z) = 2\log(\sqrt{z-1}+\sqrt{z+1}) - 2\log 2$$

As we saw above, Cauchy transforms of weighted OPs satisfy simple recurrences, and this relationship renders log transforms equally calculable.

```
T_5 = Fun(Chebyshev(), [zeros(5);1])
U_4 = Fun(Ultraspherical(1), [zeros(4);1])
x = Fun()
M_5 = z->sum(T_5/sqrt(1-x^2) * log(z-x))/π
M_5(z), 2im*cauchy(sqrt(1-x^2)*U_4,z)/5
```

```
(6.576121814966234e-8 - 2.0389718362285198e-7im, 6.576121814966217e-8
 - 2.0389718362285196e-7im)
```

For $z = x \in [-1,1]$ and $k > 0$

$$
\begin{aligned}
L_k(x) &= \frac{1}{\pi}\int_{-1}^{1} \frac{T_k(t)\log|t-x|}{\sqrt{1-t^2}}\mathrm{d}t \\
&= \Re M_k^+(x) \\
&= -\frac{2}{k}\Im\mathcal{C}^+[\sqrt{1-\diamond^2}U_{k-1}](x) \\
&= -\frac{\mathcal{H}[\sqrt{1-\diamond^2}U_{k-1}](x)}{k} \\
&= -\frac{T_k(x)}{k}
\end{aligned}
$$

and

$$
L_0(x) = \Re M_0^+(x) = -2\log 2
$$

```
x_0 = 1/sqrt(2)
L_5(x) = logkernel(w*T_5,x)
L_5(x_0),-T_5(x_0)/5
```

```
(0.1414213562373094̲5, 0.14142135623730942)
```