

give me a real way to separate plot schwarzchristoffel transformation in $x(t), y(t)$ form in wx maxima

ChatGPT said:

To plot the **Schwarz-Christoffel transformation** in parametric form $x(t), y(t)$ using **wxMaxima**, you'll need to break the complex function into its real and imaginary parts.

1. General Schwarz-Christoffel Setup

A basic Schwarz-Christoffel transformation maps the upper half-plane to a polygon. The general form is:

$$f(z) = A \int \prod_{k=1}^n (z - z_k)^{\alpha_k - 1} dz + B$$

Let's work with a simple example, say, mapping to a triangle, with:

$$f(z) = \int_0^z (1 - \zeta)^{-1/2} (1 + \zeta)^{-1/2} d\zeta$$

This maps the upper half-plane to a triangle.

2. Define the Parametric Curve

We'll choose a parameter $t \in [a, b]$, and define:

```
maxima
CopyEdit
/* Define the parameter */
tmin: -1$
tmax: 1$
```

Let $z(t) = t + i \cdot 0$ (real line — boundary of upper half-plane):

```
maxima
CopyEdit
z: t + %i*0;
```

Define the integrand:

```
maxima
CopyEdit
integrand: (1 - z)^(-1/2)*(1 + z)^(-1/2);
```

Now integrate numerically from 0 to t:

```
maxima
CopyEdit
f: realpart(rectform(romberg(integrand, t, 0, z)))$
g: imagpart(rectform(romberg(integrand, t, 0, z)))$
```

Here, f and g are $x(t)$ and $y(t)$ respectively.

3. Plotting in wxMaxima

```
maxima
CopyEdit
wxplot2d([f, g], [t, tmin, tmax], [xlabel, "x(t)", [ylabel, "y(t)"], [legend,
"Schwarz-Christoffel"])
```

⚠ Notes:

- `romberg` is a numerical integrator in Maxima. It returns a numeric value, so this approach is suitable for numeric plotting.
 - `rectform` splits complex expressions into real and imaginary parts.
 - If the integrand becomes undefined at endpoints (e.g., square root singularities), you might need to offset `t` or use complex contour integration tricks.
-

Alternative (Symbolic Attempt)

For symbolic plotting (less reliable due to complexity):

```
maxima
CopyEdit
z: t;
integrand: (1 - z)^(-1/2)*(1 + z)^(-1/2);
F: integrate(integrand, z, 0, t);
x: realpart(rectform(F));
y: imagpart(rectform(F));
wxplot2d([x, y], [t, tmin, tmax]);
```

This may fail if Maxima cannot evaluate the integral symbolically. Stick to `romberg` for robust numeric plots.

Would you like a different domain (like unit disk or vertical strip), or a polygon of your choice (e.g., square, rectangle)? I can help you tailor the transformation accordingly.

4o

You said:

`atan2(0,t^2-1)/2` convert to correct way

ChatGPT said:

The expression:

`2\atan2(0,t^2-1)`

is meant to compute **half the angle** whose tangent is t^2-1 , using the `atan2(y, x)` function.

Step-by-step Breakdown

`\atan2(0,t^2-1)`

- When $y=0$ and $x=t^2-1$, `atan2(0, x)` returns:
 - 0 if $x>0$
 - π if $x<0$

- undefined (or 0 conventionally) if $x=0$

✓ Final Piecewise Expression

So:

is undefined but Maxima returns $\frac{1}{2}\text{atan2}(0, t^2-1) = \begin{cases} 0, & t^2-1 > 0 \text{ (or } |t| > 1) \\ \pi/2, & t^2-1 < 0 \text{ (or } |t| < 1) \\ \text{undefined (or 0)}, & t = \pm 1 \end{cases}$ (since $\text{atan2}(0,0)$ is undefined but Maxima returns 0)

In wxMaxima (Maxima Code)

You can implement this using `if` or `piecewise` logic:

```
maxima
CopyEdit
theta(t) := if abs(t) > 1 then 0
            elseif abs(t) < 1 then %pi/2
            else 0; /* or declare as undefined if needed */
```

OR Use Directly:

If using Maxima's `atan2` function:

```
maxima
CopyEdit
theta(t) := atan2(0, t^2 - 1)/2;
```

But remember: `atan2(0, negative number)` returns π , so dividing by 2 gives $\pi/2$.

✓ Interpretation:

This expression is useful for computing phase angles or selecting branches of multivalued functions like the logarithm or square roots in complex analysis, often used in **Schwarz-Christoffel** or **conformal mappings**.

Let me know if you'd like to see it plotted over a range or used inside another transformation.

40

You said:

$$F = \frac{-((2\%i \log(\sqrt{t^2-1}) + t) - \%pi)/2}{x - ((-2*\text{atan2}(\sin(\text{atan2}(0, t^2-1)/2)*\sqrt{\text{abs}(t^2-1)}), \cos(\text{atan2}(0, t^2-1)/2)*\sqrt{\text{abs}(t^2-1)}) + t) - \%pi)/2} y = \frac{(\log(\sin(\text{atan2}(0, t^2-1)/2)^2*\text{abs}(t^2-1) + (\cos(\text{atan2}(0, t^2-1)/2)*\sqrt{\text{abs}(t^2-1)}) + t)^2)}{2}$$
 ChatGPT said: The expression: $\frac{1}{2}\text{atan2}(0, t^2-1)$ is meant to compute half the angle whose tangent is $\frac{0}{t^2-1}$, using the `atan2(y, x)` function. Step-by-step Breakdown $\frac{1}{2}\text{atan2}(0, t^2-1)$ When $y=0$ and $x = t^2-1$, `atan2(0, x)` returns: 0 if $x > 0$, π if $x < 0$, undefined (or 0 conventionally) if $x = 0$. ✓ Final Piecewise Expression So: $\frac{1}{2}\text{atan2}(0, t^2-1) = \begin{cases} 0, & t^2-1 > 0 \text{ (or } |t| > 1) \\ \pi/2, & t^2-1 < 0 \text{ (or } |t| < 1) \\ 0, & t = \pm 1 \end{cases}$

is undefined but Maxima returns 0) $\frac{1}{2} \operatorname{atan2}(0, t^2 - 1) = \begin{cases} 0, & t^2 - 1 > 0 \text{ (or } |t| > 1) \\ \pi/2, & t^2 - 1 < 0 \text{ (or } |t| < 1) \end{cases}$ $t = \pm 1$ (since $\operatorname{atan2}(0, 0)$ is undefined but Maxima returns 0) In wxMaxima (Maxima Code) You can implement this using if or piecewise logic: maxima Copy Edit $\theta(t) := \text{if } \text{abs}(t) > 1 \text{ then } 0 \text{ elseif } \text{abs}(t) < 1 \text{ then } \%pi/2 \text{ else } 0; /* or declare as undefined if needed */$ OR Use Directly: If using Maxima's atan2 function: maxima Copy Edit $\theta(t) := \operatorname{atan2}(0, t^2 - 1)/2$; But remember: $\operatorname{atan2}(0, \text{negative number})$ returns π , so dividing by 2 gives $\pi/2$. ✓ Interpretation: This expression is useful for computing phase angles or selecting branches of multivalued functions like the logarithm or square roots in complex analysis, often used in Schwarz-Christoffel or conformal mappings. Let me know if you'd like to see it plotted over a range or used inside another transformation. convert this correct way

ChatGPT said:

You're working with a **Schwarz-Christoffel transformation** involving a complex logarithmic expression, and want to **separate and correct it** into a real-valued parametric form $x(t), y(t)$ in wxMaxima.

You provided:

- $F: -(22i \log(t^2 - 1 + t) - \pi)$
- $x(t)$ and $y(t)$ are derived from the real and imaginary parts, but expressed in a complex and branch-sensitive way.

✓ Goal: Correct and Simplify the Parametric Form

Step 1: Correct the Complex Function

Given:

$$F(t) = -(22i \log(t^2 - 1 + t) - \pi)$$

Let's simplify:

$$F(t) = -i \log(t^2 - 1 + t) + 2\pi$$

So,

$$F(t) = x(t) + iy(t)$$

This gives us:

- $x(t) = \operatorname{Re} F(t)$
- $y(t) = \operatorname{Im} F(t)$

Step 2: Extract Real and Imaginary Parts

Let's define:

$$w(t) = \log(t^2 - 1 + t)$$

Use the identity for complex logarithm:

$$\log(a+ib)=\log(a^2+b^2)+i\operatorname{atan2}(b,a)$$

Let:

$$u(t)=t^2-1+t$$

Then:

- $\Re(w(t))=\log(|u(t)|)$
- or $\Im(w(t))=\operatorname{atan2}(0,u(t))=0$ or π , depending on the sign of $u(t)$

Therefore:

$$F(t)=-i\log(|u(t)|)-\operatorname{atan2}(0,u(t))+2\pi$$

Step 3: Define in wxMaxima

Here's the corrected version in **wxMaxima**:

```
maxima
CopyEdit
/* Define u(t) */
u(t) := sqrt(t^2 - 1) + t;

/* Real part of F(t) */
x(t) := -imagpart(log(u(t))) + %pi/2;

/* Imaginary part of F(t) */
y(t) := -realpart(log(u(t)));
```

Alternative using explicit functions:

```
maxima
CopyEdit
x(t) := -atan2(0, u(t)) + %pi/2;
y(t) := -log(abs(u(t)));
```

Step 4: Plot x(t), y(t)

```
maxima
CopyEdit
wxplot2d([parametric, x(t), y(t), [t, -2, 2]],
          [xlabel, "x(t)", [ylabel, "y(t)",
          [legend, "SC mapping"]]);
```

✓ Summary

You started from:

$$F(t)=-(22i\log(t^2-1+t)-\pi)$$

And we correctly derived:

$$x(t)=-\operatorname{atan2}(0,t^2-1+t)+2\pi \quad y(t)=-\log(|t^2-1+t|)$$

This form correctly reflects the conformal mapping for the SC transform using a square-root logarithmic branch. Let me know if you want to extend this to a polygon map or animate it.

4o

You said:

logic operators symbol

ChatGPT said:

Here are the **logical operators and their symbols** used in most programming and mathematical environments like **Maxima, C, Python**, and **general logic**:

Logical Operators (with Symbols)

Operator	Symbol(s)	Meaning / Description
AND	and, &&, \wedge	True if both operands are true
OR	or, \vee	
NOT	not, !, \neg	Inverts the truth value
XOR (Exclusive OR)	xor, \oplus	True if operands are different
Implication	->, \Rightarrow	"If A, then B"
Bi-implication	<->, \Leftrightarrow	"A if and only if B"
NAND	!(A and B), \uparrow	NOT AND
NOR	!(A or B), \downarrow	NOT OR

In wxMaxima / Maxima

Operator	Symbol	Example
AND	and	true and false
OR	or	true or false
NOT	not	not true
XOR	xor	true xor false
Implies	implies	A implies B
Equal	=	A = B
Not Equal	#	A # B

Unicode Logic Symbols (for docs/math)

Symbol	Name	Unicode
\wedge	Logical AND	U+2227
\vee	Logical OR	U+2228
\neg	Logical NOT	U+00AC
\Rightarrow	Implies	U+21D2
\Leftrightarrow	If and only if	U+21D4
\oplus	Exclusive OR	U+2295

Symbol	Name	Unicode
↑	NAND	U+2191
↓	NOR	U+2193

Would you like a printable table image or code snippet to test these in Maxima or another language?

4o

You said:

for or straight line in keyyboard

ChatGPT said:

The **OR** operator using a **straight line** on the keyboard is:

CopyEdit
|

This is called the **pipe** symbol and usually represents **bitwise OR** or **logical OR** in many programming languages.

Details:

- On most keyboards, the **pipe symbol |** is typed by pressing:
 - **Shift + Backslash key (\)**
(The backslash key is usually above the Enter key or near the right Shift key.)
- For **logical OR** in many languages, you use **double pipes ||**:

```
c
CopyEdit
if (a || b) {
    // True if a OR b is true
}
```

Summary:

Operator	Symbol on Keyboard	Usage
OR	`	` (pipe)
Logical OR		

If you want the **single vertical bar** character for **OR** just press **Shift + ** key once.

Need help finding the key on your keyboard? Which keyboard layout are you using?

4.1-mini

You said:

```
z: t; integrand: (1 - z)^(-1/2)*(1 + z)^(-1/2); F: integrate(integrand, z, 0, t); x: realpart(rectform(F));
y: imagpart(rectform(F)); wxplot2d([x, y], [t, tmin, tmax]); for an 10 hexogonal coordinate
```

ChatGPT said:

Got it! You want to:

- Define $z=t$
 - Integrate the function $(1-z)^{-1/2}(1+z)^{-1/2}$ from 0 to t
 - Extract the real and imaginary parts of the integral result (converted to rectangular form)
 - Plot the parametric curve $(x(t),y(t))$ for t over a certain range
 - And you want to do this for a 10-sided hexagonal coordinate (probably meaning a polygon with 10 vertices; maybe you want the Schwarz-Christoffel map for a 10-gon?)
-

Let's break down the Maxima code with your inputs:

```
maxima
CopyEdit
/* Define the variable */
z : 't;

/* Define the integrand */
integrand : (1 - z)^(-1/2) * (1 + z)^(-1/2);

/* Compute the definite integral from 0 to t */
F : integrate(integrand, z, 0, t);

/* Convert to rectangular form to separate real and imaginary parts */
F_rect : rectform(F);

/* Extract real and imaginary parts */
x : realpart(F_rect);
y : imagpart(F_rect);

/* Plot parametric curve (x(t), y(t)) over t from tmin to tmax */
wxplot2d([parametric, x, y, [t, tmin, tmax]]);
```

Comments / Notes:

- The integrand $(1-z)^{-1/2}(1+z)^{-1/2}$ corresponds to the Schwarz-Christoffel mapping integrand for a **line segment** or a polygon with two right-angled vertices (or a special case).
 - For a polygon with **10 vertices**, the Schwarz-Christoffel mapping integrand would have factors like $(z-z_k)^{\alpha_k-1}$ for each vertex z_k , where α_k depends on the interior angle at vertex k .
 - Your current integrand is for a very simple case (two singularities at ± 1).
 - If you want the 10-sided polygon, the integrand will be more complex — involving 10 terms.
-

Example for a 10-sided polygon (decagon)

For a regular decagon, the interior angle is:

$$\theta = 10(10-2) \times \pi = 108\pi = 54\pi$$

The exterior angle (turning angle) is:

$$\alpha = 1 - \pi\theta = 1 - 54 = 51$$

So each vertex contributes a factor $(z - z_k)^{\alpha - 1} = (z - z_k)^{-4/5}$ in the integrand.

The integrand is then roughly:

$$k = 1 \prod_{k=1}^{10} (z - z_k)^{-4/5}$$

where z_k are prevertices on the real axis (or unit circle).

To Do a proper 10-gon Schwarz-Christoffel map, you need:

1. The locations of the prevertices z_k (usually on the real axis).
 2. The turning angles α_k .
 3. Construct the integrand as a product of $(z - z_k)^{\alpha_k - 1}$.
 4. Integrate from one prevertex to t .
 5. Separate into $x(t), y(t)$.
 6. Plot over your range.
-

A simplified example for your current integral on $[-1, 1]$:

```
maxima
CopyEdit
/* Set t range */
tmin : -0.99;
tmax : 0.99;

/* Define variable */
z : 't;

/* Define integrand */
integrand : (1 - z)^(-1/2) * (1 + z)^(-1/2);

/* Compute definite integral */
F : integrate(integrand, z, 0, t);

/* Convert to rectangular form */
F_rect : rectform(F);

/* Extract real and imaginary parts */
x : realpart(F_rect);
y : imagpart(F_rect);

/* Plot parametric curve */
wxplot2d([parametric, x, y, [t, tmin, tmax]], [xlabel, "x(t)"], [ylabel,
"y(t)"]);
```

For a 10-gon, you need the full integrand like:

```
maxima
CopyEdit
/* Example: define prevertices on real axis */
z_k : makelist(cos(2*pi*k/10), k, 1, 10);

/* Interior angles (for regular polygon) */
alpha_k : makelist(1 - 4/5, k, 1, 10);

/* Build integrand product */
integrand_10gon : 1;
for k:1 thru 10 do
  integrand_10gon : integrand_10gon * (t - z_k[k])^(alpha_k[k] - 1);

/* Compute integral */
F_10gon : integrate(integrand_10gon, t, z_k[1], t);

/* Then extract real/imag and plot similarly */
```

If you want me to write full Maxima code for the **Schwarz-Christoffel transformation** of a **regular 10-gon** including numerical evaluation and plotting, just say! I can help step-by-step.

TL;DR:

- Your current code integrates a simple function related to a 2-vertex polygon.
 - To do a 10-sided polygon, you need a more complex integrand involving all vertices.
 - The process is: build integrand → integrate → separate real/imag → plot parametric.
-

Want me to generate the 10-gon Schwarz-Christoffel integrand and plotting code for wxMaxima?

4.1-mini

Tools

ChatGPT can make mistakes. Check important info. See [Cookie Preferences](#).