Summary of the PhD Defense Presentation

Vijay Jung Kunwar

Notation:

∂	$\frac{d}{dx}$
L_{inp}	Input differential operator; a second order linear differential operator
$H^{a,b}_{c,x}$	Gauss hypergeometric differential operator
$S(x) = {}_2F_1(a, b; c \mid x)$	Gauss hypergeometric function; a solution of $H^{a,b}_{c,x}$
(e_0, e_1, e_∞)	Exponent differences of $H^{a,b}_{c,x}$ at $(0, 1, \infty)$
$\xrightarrow{f}_C$	Change of variables
$\xrightarrow{r_0,r_1}_G$	Gauge transformation
$\xrightarrow{r}_E$	Exponential product
$[g_0, g_1, \ldots, g_k]$	k-constellation; $g_i \in S_n$ (details in the slides)

Given:

A second order linear differential operator L_{inp} with rational function coefficients (see slides for motivation);

1. irreducible and has no Liouvillian solutions,
2. has five regular singularities where at least one of the singularities is *logarithmic*, and
3. has arithmetic monodromy group.

Goal:

Solve L_{inp} in terms of ${}_2F_1(a, b; c \mid f)$, i.e, find a solution y of the following form (also called the *closed form solution*):

$$y = \exp(\int r\, dx) \left(r_0 S(f) + r_1 S(f)'\right) \neq 0$$

where $S(x) = {}_2F_1(a, b; c \mid x)$, $r, r_0, r_1, f \in \mathbb{C}(x)$.

Problem Discussion:

For the solver program to be complete we need the complete tables of Belyi, Belyi-1 and Belyi-2 maps. We use the following correspondence to prove the completeness:

Belyi maps	$\longleftrightarrow$	dessins, i.e, equivalence classes of 3-constellations $[g_0, g_1, g_\infty]$ mod conjugation
Belyi-1 maps	$\longleftrightarrow$	near dessins, i.e, equivalence classes of 4-constellations $[g_0, g_1, g_t, g_\infty]$ mod conjugation
Belyi-2 maps	$\longleftrightarrow$	we have algorithms to compute such maps

- We compute Belyi, Belyi-1 maps solving polynomial equations and using other techniques.
- We compute dessins, near dessins using combinatorial search plus various techniques to prevent computational explosion.

Proof: Compare!

```
> f:= -(x^2-3) / (3*x^2-1)*x^4;
```

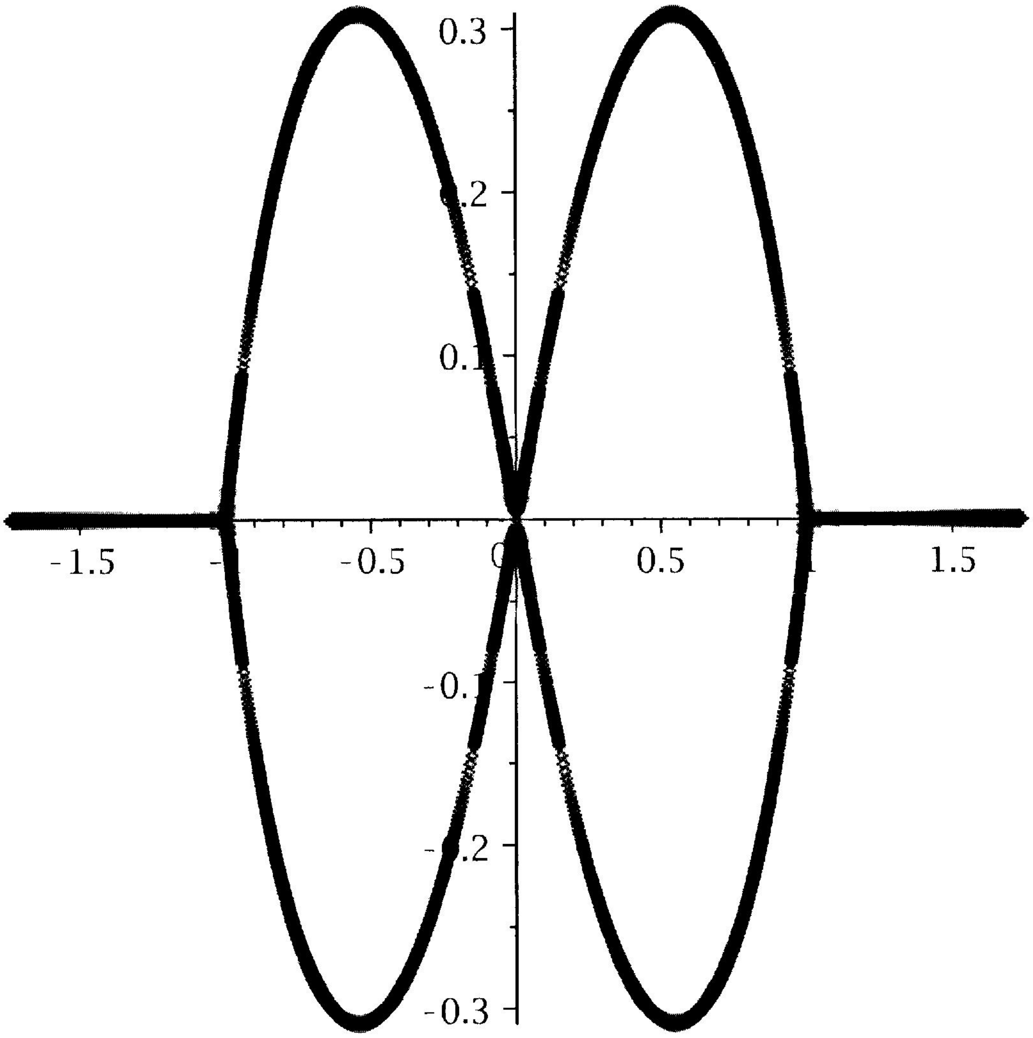
$$f := -\frac{(x^2-3)\,x^4}{3\,x^2-1} \tag{1}$$

```
> factor(1 - f);
```

$$\frac{(x-1)^3\,(x+1)^3}{3\,x^2-1} \tag{2}$$

> **read** *PlotDessin*: *plots*[*pointplot*](*pts*); # *plots* f^{-1}([0, 1]).

\>

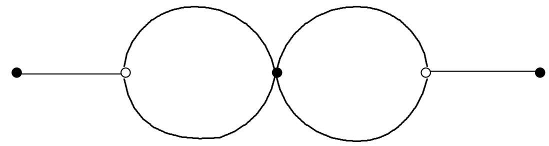



```
> algcurves[monodromy]( numer(f-y), y, x)[-1];
```

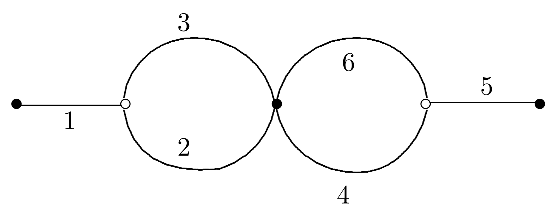
[[0., [[2, 3, 5, 4]]], [1., [[1, 3, 2], [4, 5, 6]]], [∞, [[1, 4, 6, 3]]]] **(3)**

0: cycles of length 4, 1, 1
1: cycles of length 3, 3
inf: cycles of length 4, 1, 1 (1-cycles are not printed)

Dessin:



"Labelled dessin", i.e, 3-constellation:



Read off permutations:

black vertices:	$x = -\sqrt{3}, x = 0, x = \sqrt{3}$	$g_0 = (1)\,(2\,4\,6\,3)\,(5)$
white vertices:	$x = -1, x = 1$	$g_1 = (1\,2\,3)\,(4\,5\,6)$
faces:	$x = -\frac{1}{\sqrt{3}}, x = \frac{1}{\sqrt{3}}, x = \infty$	$g_\infty = (2)\,(6)\,(1\,3\,5\,4)$ (follow outside face, $x = \infty$, clockwise)

Note:

Since the labelling was done arbitrarily, we should expect the result from "`algcurves[monodromy]`" to match only up to conjugation (same dessin, not the same 3-constellation). We have a program that can find the conjugation.