

Braid monodromy computations using certified path tracking

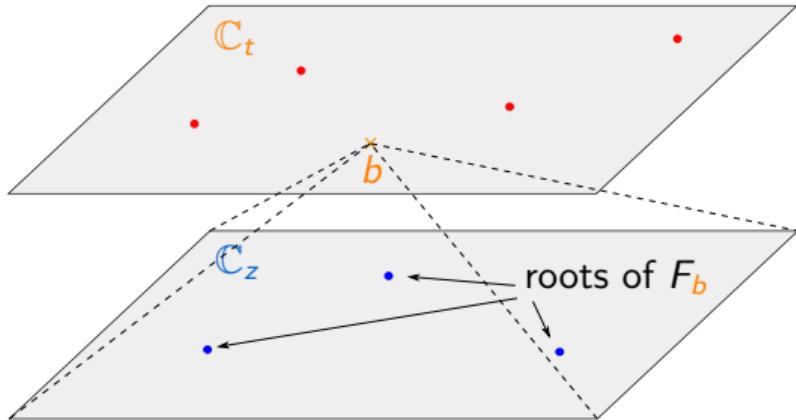
Alexandre Guillemot

Joint work with Pierre Lairez
MATHEXP, Inria, France

Journées de géométrie algorithmique
October 14, 2025 | Roscoff



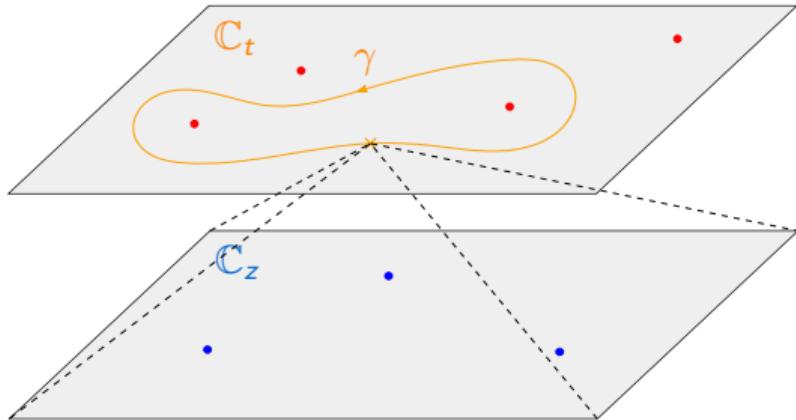
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \setminus \Sigma$ be a base point,

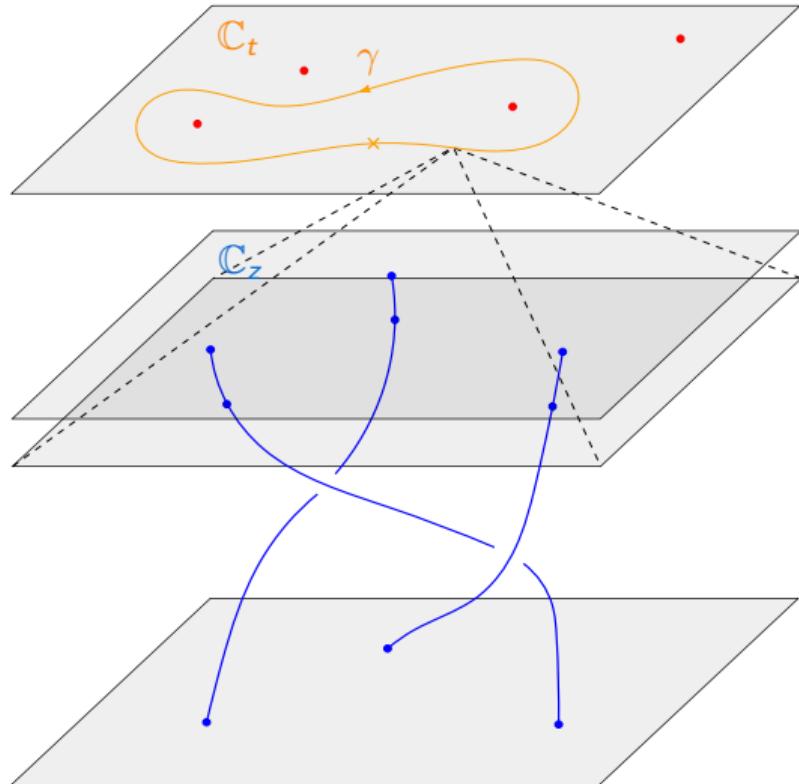
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \setminus \Sigma$ be a base point,
- let $\gamma : [0, 1] \rightarrow \mathbb{C} \setminus \Sigma$ be a loop starting at b .

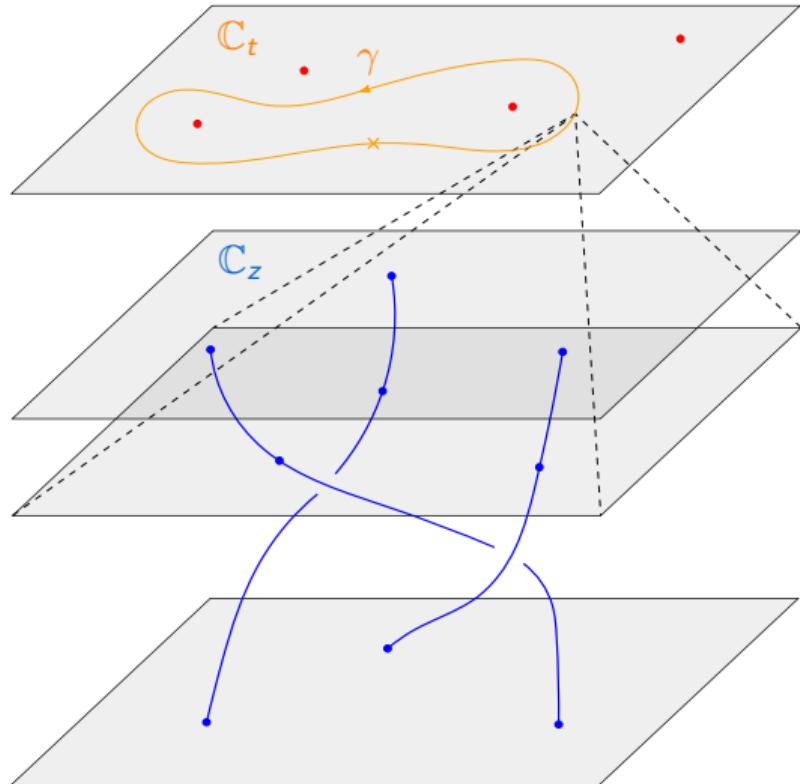
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \setminus \Sigma$ be a base point,
- let $\gamma : [0, 1] \rightarrow \mathbb{C} \setminus \Sigma$ be a loop starting at b .
- The displacement of all **roots** of F_t when t moves along γ defines a braid.

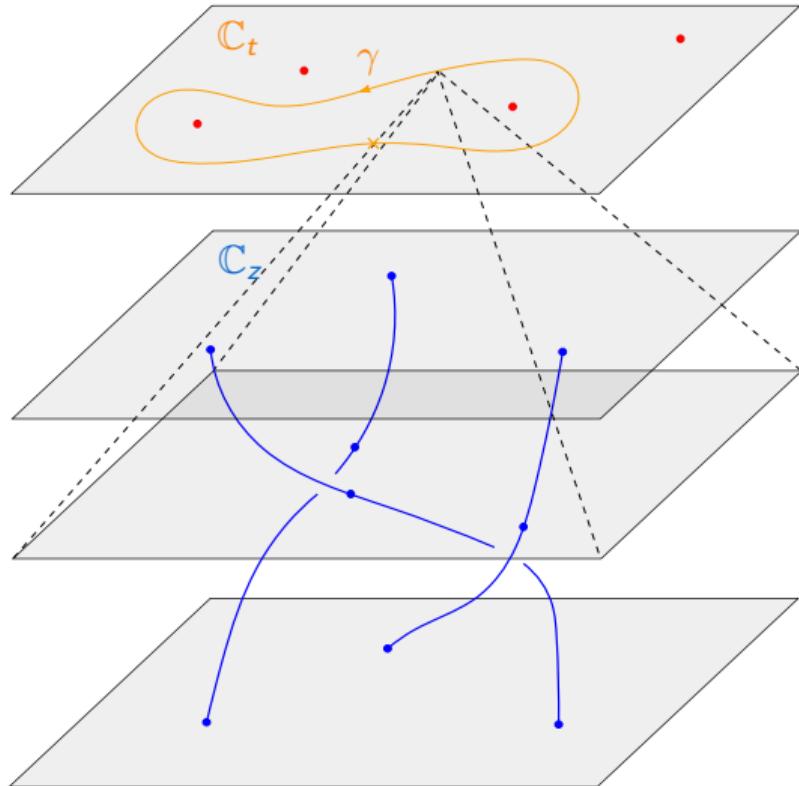
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \setminus \Sigma$ be a base point,
- let $\gamma : [0, 1] \rightarrow \mathbb{C} \setminus \Sigma$ be a loop starting at b .
- The displacement of all **roots** of F_t when t moves along γ defines a braid.

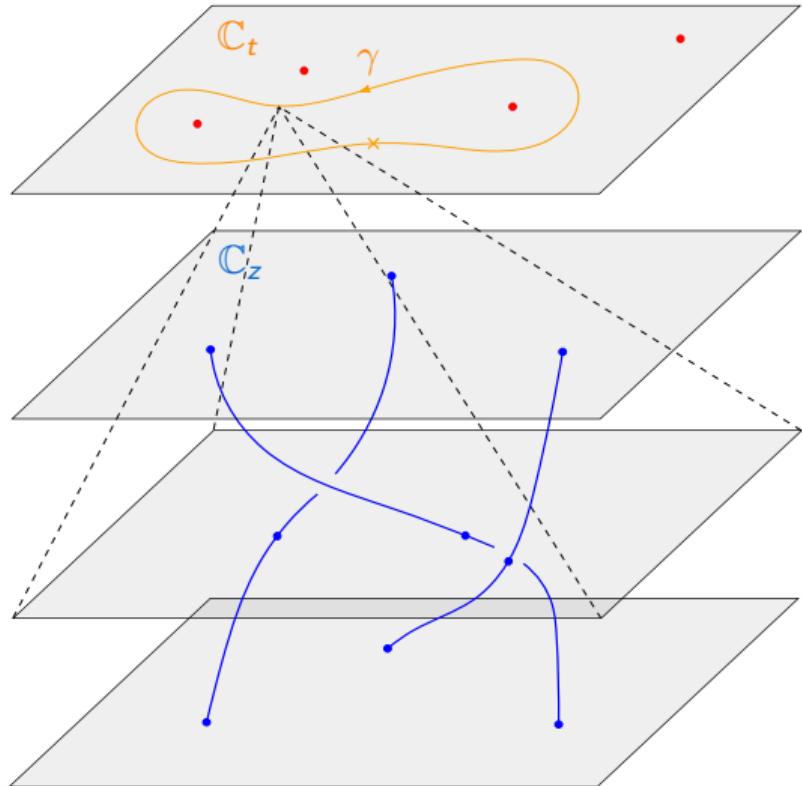
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \setminus \Sigma$ be a base point,
- let $\gamma : [0, 1] \rightarrow \mathbb{C} \setminus \Sigma$ be a loop starting at b .
- The displacement of all **roots** of F_t when t moves along γ defines a braid.

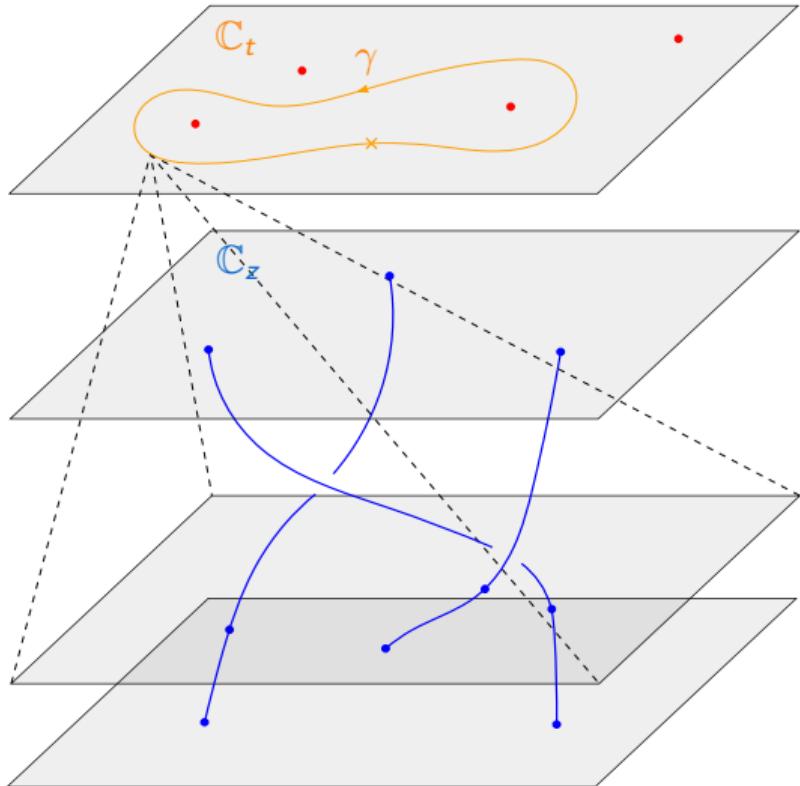
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \setminus \Sigma$ be a base point,
- let $\gamma : [0, 1] \rightarrow \mathbb{C} \setminus \Sigma$ be a loop starting at b .
- The displacement of all **roots** of F_t when t moves along γ defines a braid.

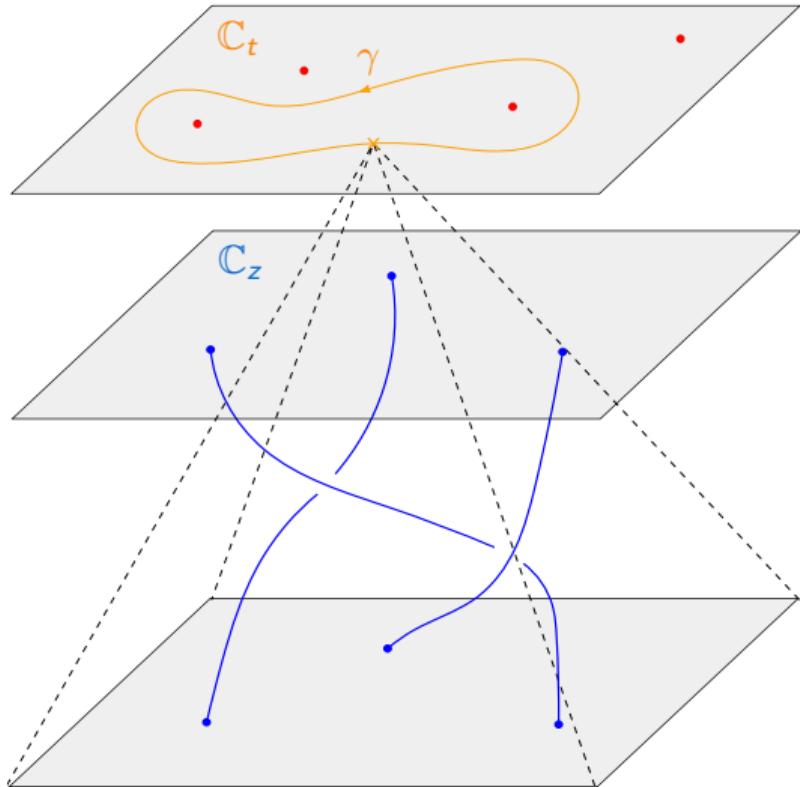
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \setminus \Sigma$ be a base point,
- let $\gamma : [0, 1] \rightarrow \mathbb{C} \setminus \Sigma$ be a loop starting at b .
- The displacement of all roots of F_t when t moves along γ defines a braid.

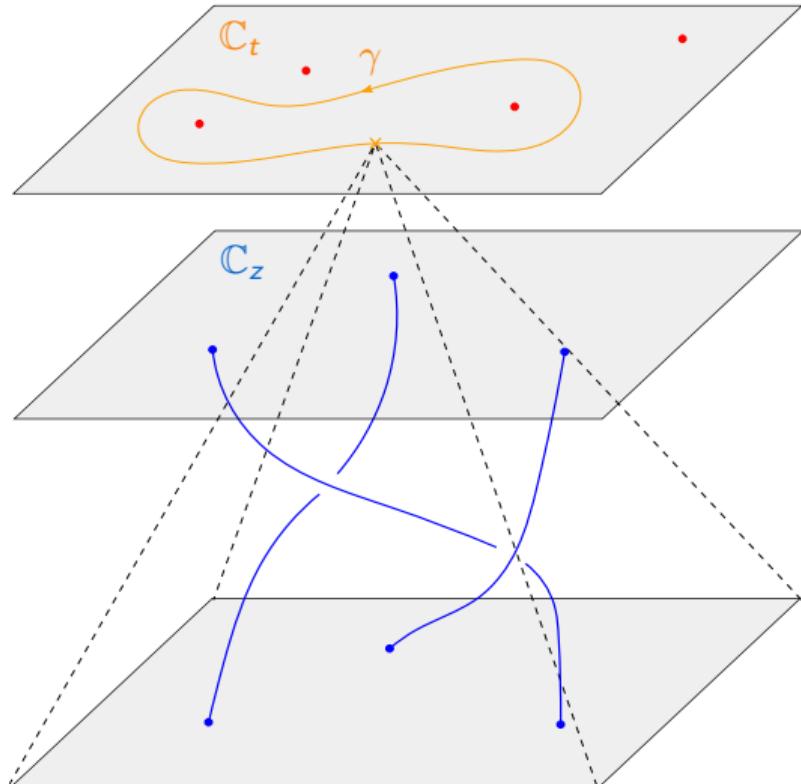
Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \setminus \Sigma$ be a base point,
- let $\gamma : [0, 1] \rightarrow \mathbb{C} \setminus \Sigma$ be a loop starting at b .
- The displacement of all **roots** of F_t when t moves along γ defines a braid.

Motivation



Setup

- Let $g \in \mathbb{C}[t, z]$ ($n = \deg_z(g)$),
- define $F_t(z) = g(t, z)$.
- Let $b \in \mathbb{C} \setminus \Sigma$ be a base point,
- let $\gamma : [0, 1] \rightarrow \mathbb{C} \setminus \Sigma$ be a loop starting at b .
- The displacement of all **roots** of F_t when t moves along γ defines a braid.

Algorithmic goal

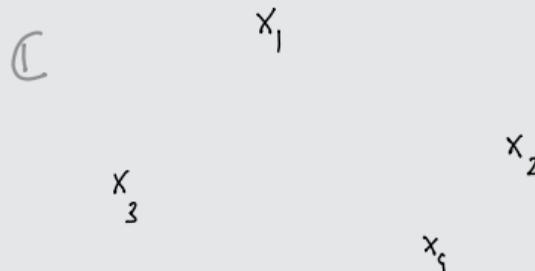
Input: g, γ

Output: the associated braid in terms of Artin's generators

Configurations

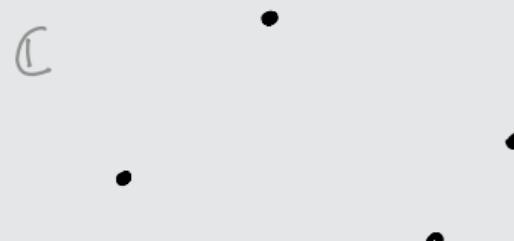
Ordered configurations

$$OC_n = \{(x_1, \dots, x_n) \in \mathbb{C}^n : \forall i \neq j, x_i \neq x_j\}.$$



Configurations

$$C_n = \{\text{subsets of size } n \text{ in } \mathbb{C}\}.$$



“Forget order” projection

$$\begin{aligned} \Phi : \quad OC_n &\rightarrow C_n \\ (x_1, \dots, x_n) &\mapsto \{x_1, \dots, x_n\}. \end{aligned}$$

Braid group

Braid

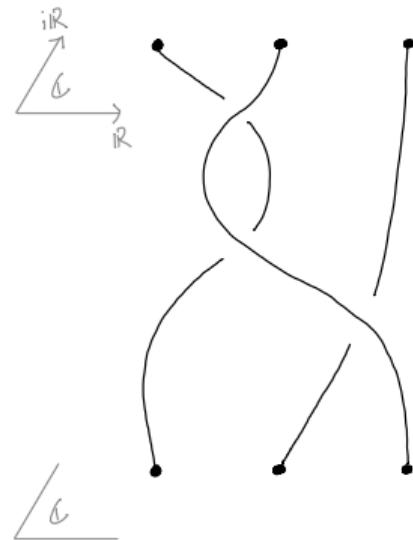
Homotopy class of a path $\beta : [0, 1] \rightarrow C_n$ such that $\beta(0) = \beta(1) = \{1, \dots, n\}$.



Braid group

Braid

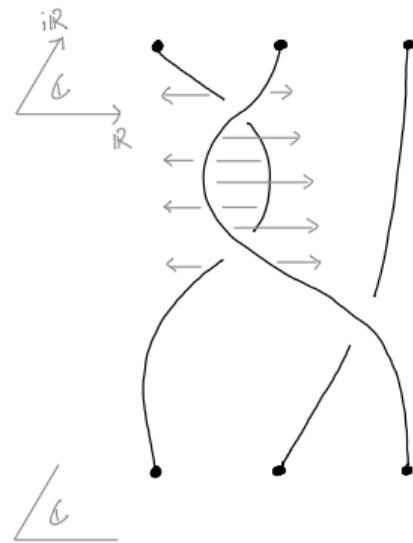
Homotopy class of a path $\beta : [0, 1] \rightarrow C_n$ such that $\beta(0) = \beta(1) = \{1, \dots, n\}$.



Braid group

Braid

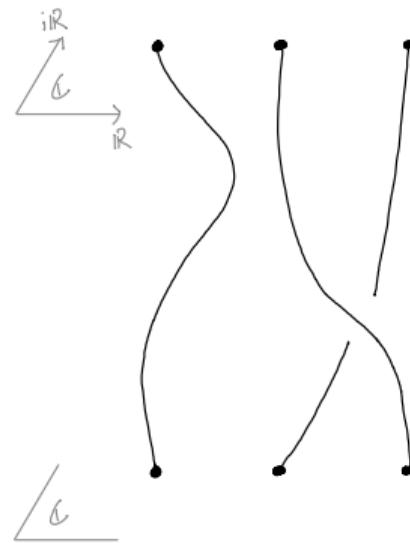
Homotopy class of a path $\beta : [0, 1] \rightarrow C_n$ such that $\beta(0) = \beta(1) = \{1, \dots, n\}$.



Braid group

Braid

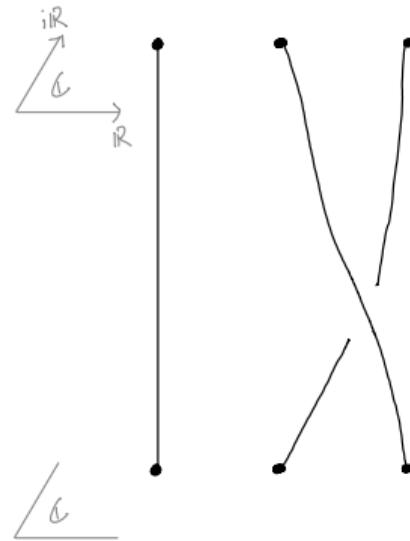
Homotopy class of a path $\beta : [0, 1] \rightarrow C_n$ such that $\beta(0) = \beta(1) = \{1, \dots, n\}$.



Braid group

Braid

Homotopy class of a path $\beta : [0, 1] \rightarrow C_n$ such that $\beta(0) = \beta(1) = \{1, \dots, n\}$.

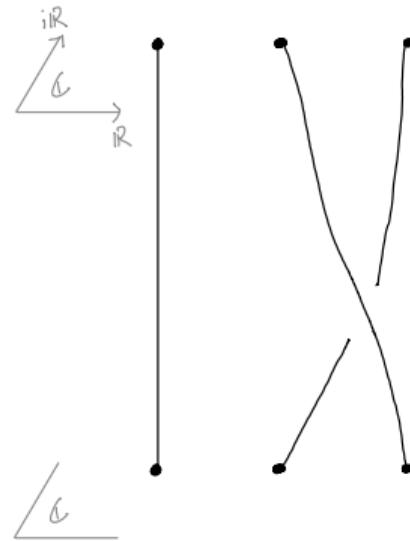


Braid group

Braid

Homotopy class of a path $\beta : [0, 1] \rightarrow C_n$ such that $\beta(0) = \beta(1) = \{1, \dots, n\}$.

In practice, we will manipulate paths in OC_n .



Braid group

Braid

Homotopy class of a path $\beta : [0, 1] \rightarrow C_n$ such that $\beta(0) = \beta(1) = \{1, \dots, n\}$.

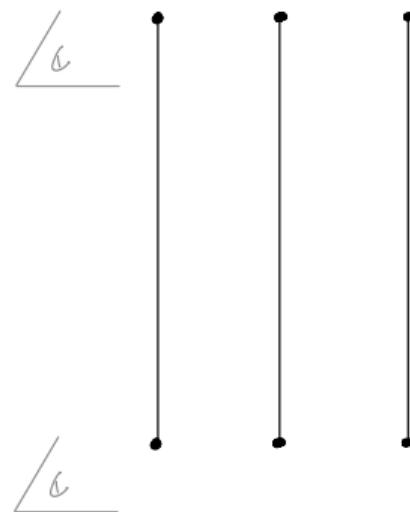
In practice, we will manipulate paths in OC_n .

Braid group B_n

id: class of the constant path equal to $\{1, \dots, n\}$.

Law: $[\beta_1][\beta_2] := [\beta_1 \cdot \beta_2]$.

Rk: this is $\pi_1(C_n, \{1, \dots, n\})$.



id_{B_3}

Braid group

Braid

Homotopy class of a path $\beta : [0, 1] \rightarrow C_n$ such that $\beta(0) = \beta(1) = \{1, \dots, n\}$.

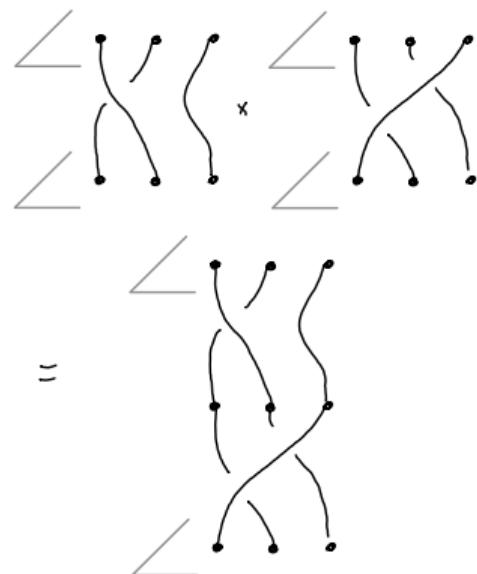
In practice, we will manipulate paths in OC_n .

Braid group B_n

id: class of the constant path equal to $\{1, \dots, n\}$.

Law: $[\beta_1][\beta_2] := [\beta_1 \cdot \beta_2]$.

Rk: this is $\pi_1(C_n, \{1, \dots, n\})$.



Braid group

Braid

Homotopy class of a path $\beta : [0, 1] \rightarrow C_n$ such that $\beta(0) = \beta(1) = \{1, \dots, n\}$.

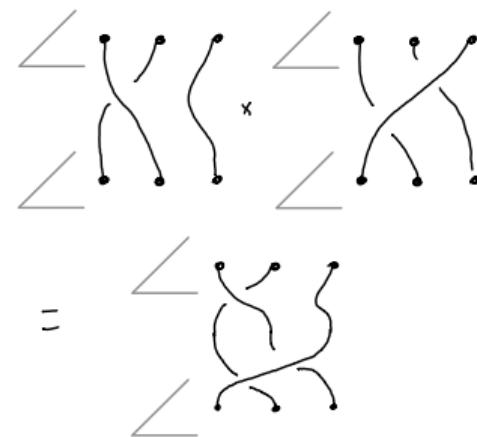
In practice, we will manipulate paths in OC_n .

Braid group B_n

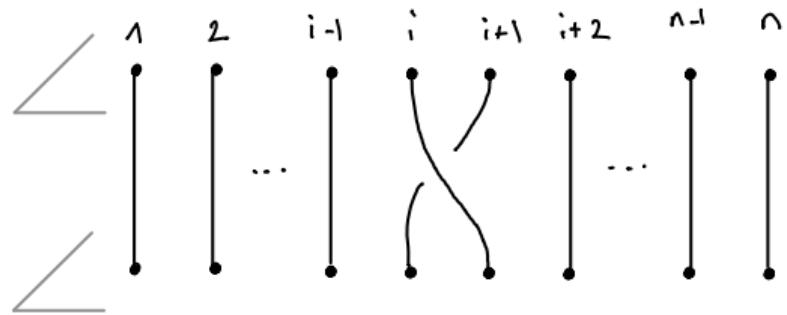
id: class of the constant path equal to $\{1, \dots, n\}$.

Law: $[\beta_1][\beta_2] := [\beta_1 \cdot \beta_2]$.

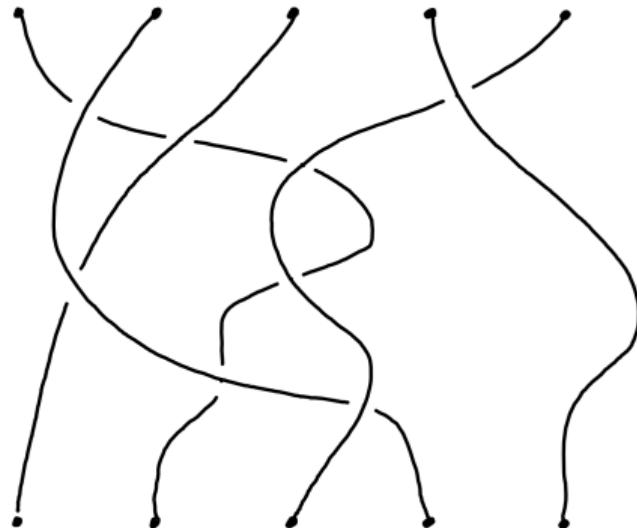
Rk: this is $\pi_1(C_n, \{1, \dots, n\})$.



Artin's theorem



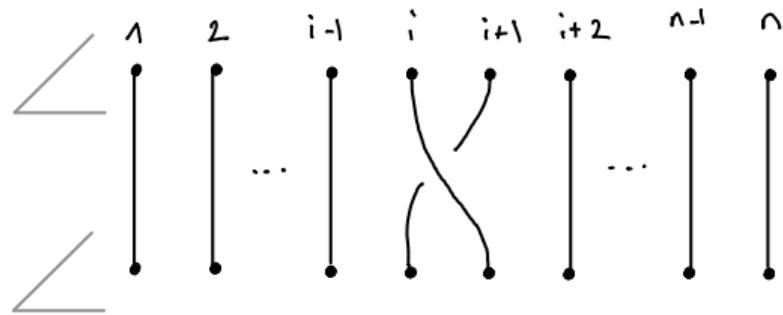
$$\sigma_i \in B_n$$



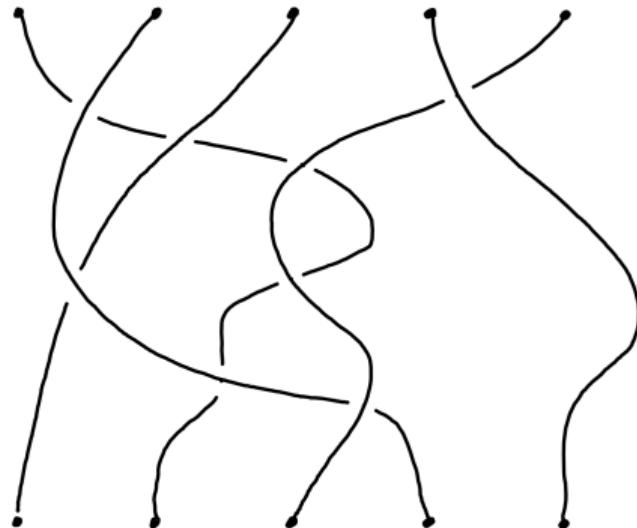
Theorem [Artin, 1947]

The σ_i 's generate B_n (+ explicit relations).

Artin's theorem



$$\sigma_i \in B_n$$



Theorem [Artin, 1947]

The σ_i 's generate B_n (+ explicit relations).

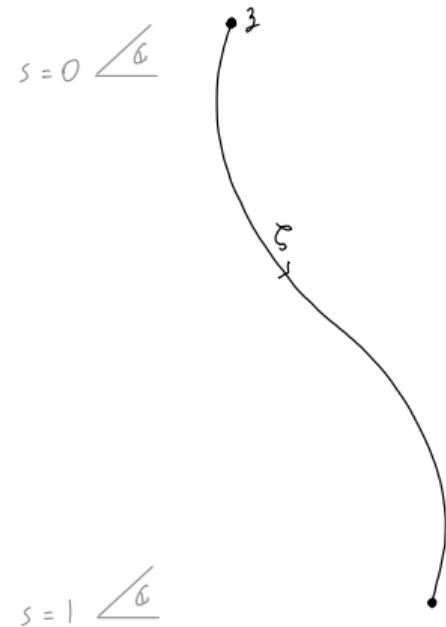
$$\sigma_4\sigma_1^{-1}\sigma_2^{-1}\sigma_3^{-1}\sigma_3\sigma_1\sigma_2\sigma_3^{-1}$$

Main tool

Certified homotopy continuation

Input: $H : [0, 1] \times \mathbb{C}^r \rightarrow \mathbb{C}^r$ and $z \in \mathbb{C}^r$ such that $H(0, z) = 0$.

There exists $\zeta : [0, 1] \rightarrow \mathbb{C}^r$ such that $H(s, \zeta(s)) = 0$ and $\zeta(0) = z$. Assume it is unique.



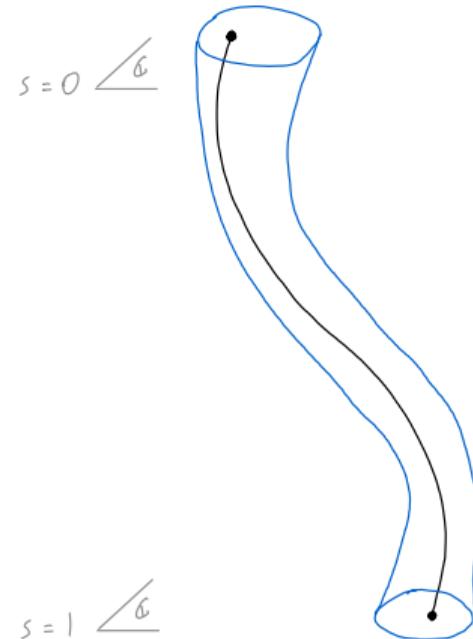
Main tool

Certified homotopy continuation

Input: $H : [0, 1] \times \mathbb{C}^r \rightarrow \mathbb{C}^r$ and $z \in \mathbb{C}^r$ such that $H(0, z) = 0$.

There exists $\zeta : [0, 1] \rightarrow \mathbb{C}^r$ such that $H(s, \zeta(s)) = 0$ and $\zeta(0) = z$. Assume it is unique.

Output: A tubular neighborhood isolating ζ .



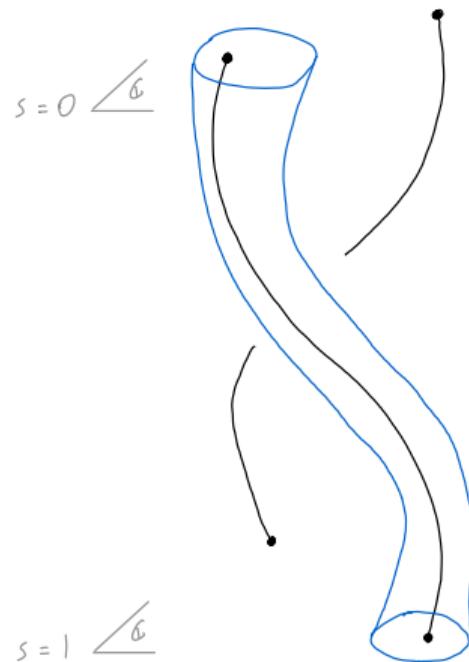
Main tool

Certified homotopy continuation

Input: $H : [0, 1] \times \mathbb{C}^r \rightarrow \mathbb{C}^r$ and $z \in \mathbb{C}^r$ such that $H(0, z) = 0$.

There exists $\zeta : [0, 1] \rightarrow \mathbb{C}^r$ such that $H(s, \zeta(s)) = 0$ and $\zeta(0) = z$. Assume it is unique.

Output: A tubular neighborhood isolating ζ .



Main tool

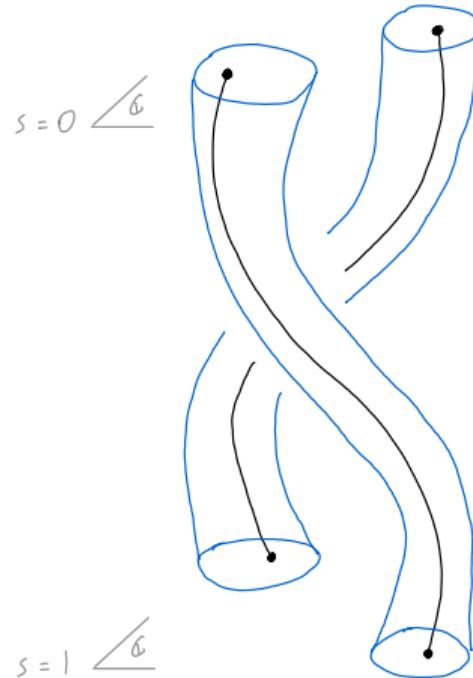
Certified homotopy continuation

Input: $H : [0, 1] \times \mathbb{C}^r \rightarrow \mathbb{C}^r$ and $z \in \mathbb{C}^r$ such that $H(0, z) = 0$.

There exists $\zeta : [0, 1] \rightarrow \mathbb{C}^r$ such that $H(s, \zeta(s)) = 0$ and $\zeta(0) = z$. Assume it is unique.

Output: A tubular neighborhood isolating ζ .

We can do that for every solution at $s = 0$.



Main tool

Certified homotopy continuation

Input: $H : [0, 1] \times \mathbb{C}^r \rightarrow \mathbb{C}^r$ and $z \in \mathbb{C}^r$ such that $H(0, z) = 0$.

There exists $\zeta : [0, 1] \rightarrow \mathbb{C}^r$ such that $H(s, \zeta(s)) = 0$ and $\zeta(0) = z$. Assume it is unique.

Output: A tubular neighborhood isolating ζ .

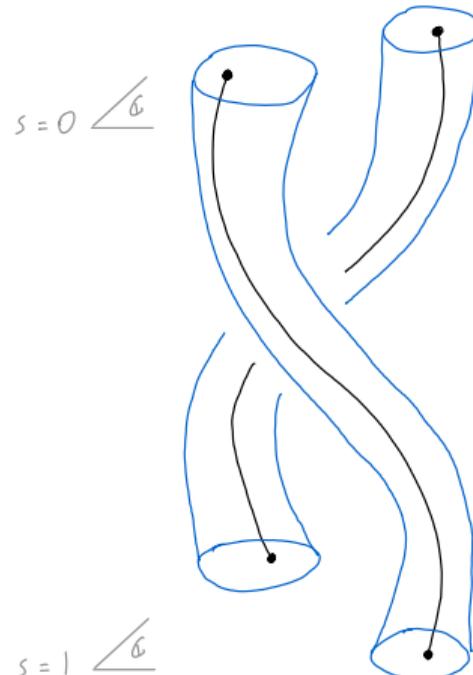
We can do that for every solution at $s = 0$.

Application

Recall $g \in \mathbb{C}[t, z]$ and $\gamma : [0, 1] \rightarrow \mathbb{C} \setminus \Sigma$ from first slide.

Apply certified homotopy continuation to

$H(s, z) = g(\gamma(s), z)$.



Today's goal

We now assume $\zeta = (\zeta_1, \dots, \zeta_n) : [0, 1] \rightarrow OC_n$ inducing a loop in C_n i.e. $\Phi(\zeta(0)) = \Phi(\zeta(1))$.

Goal

Input : ζ (n disjoint tubular neighborhoods around ζ_1, \dots, ζ_n).

Output : A decomposition in standard generators of the braid induced by ζ_1, \dots, ζ_n .

Today's goal

We now assume $\zeta = (\zeta_1, \dots, \zeta_n) : [0, 1] \rightarrow OC_n$ inducing a loop in C_n i.e. $\Phi(\zeta(0)) = \Phi(\zeta(1))$.

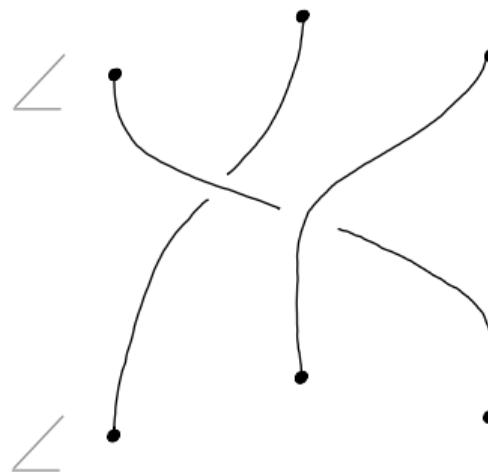
Goal

Input : ζ (n disjoint tubular neighborhoods around ζ_1, \dots, ζ_n).

Output : A decomposition in standard generators of the braid induced by ζ_1, \dots, ζ_n .

Overall strategy

! We do not have access to ζ , not even to $\zeta(0)$.



Today's goal

We now assume $\zeta = (\zeta_1, \dots, \zeta_n) : [0, 1] \rightarrow OC_n$ inducing a loop in C_n i.e. $\Phi(\zeta(0)) = \Phi(\zeta(1))$.

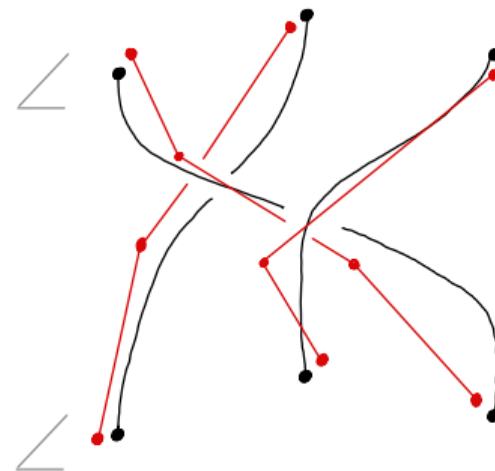
Goal

Input : ζ (n disjoint tubular neighborhoods around ζ_1, \dots, ζ_n).

Output : A decomposition in standard generators of the braid induced by ζ_1, \dots, ζ_n .

Overall strategy

- ! We do not have access to ζ , not even to $\zeta(0)$.
- 1) Find a path $\tilde{\zeta}$ that has same associated braid.



Today's goal

We now assume $\zeta = (\zeta_1, \dots, \zeta_n) : [0, 1] \rightarrow OC_n$ inducing a loop in C_n i.e. $\Phi(\zeta(0)) = \Phi(\zeta(1))$.

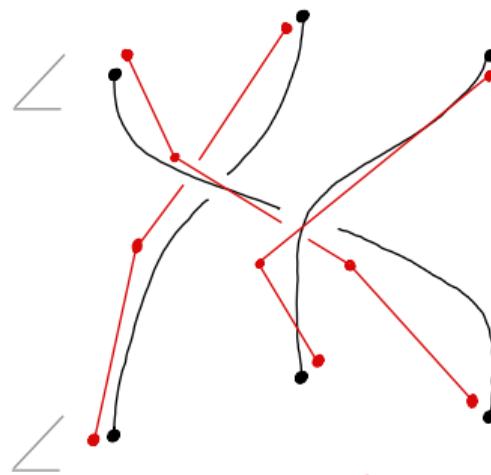
Goal

Input : ζ (n disjoint tubular neighborhoods around ζ_1, \dots, ζ_n).

Output : A decomposition in standard generators of the braid induced by ζ_1, \dots, ζ_n .

Overall strategy

- ! We do not have access to ζ , not even to $\zeta(0)$.
- 1) Find a path $\tilde{\zeta}$ that has same associated braid.
- 2) Decompose $\tilde{\zeta}$.

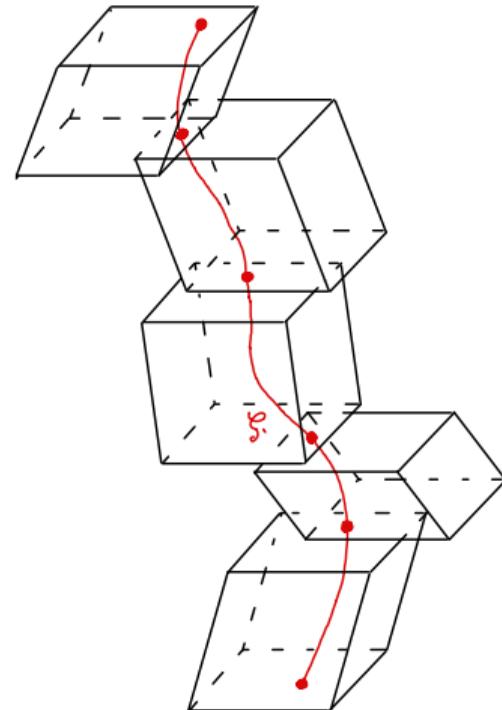


$$\sigma_1 \sigma_2^{-1}$$

Related work

SIROCCO [Marco-Buzunariz and Rodríguez, 2016]

- Tubular neighborhoods are piecewise **linear**.
- For each strand ζ_i , computes a piecewise linear path in the tube.
- “Intuitive” (! non generic cases) algorithm on the braid with piecewise linear strands.



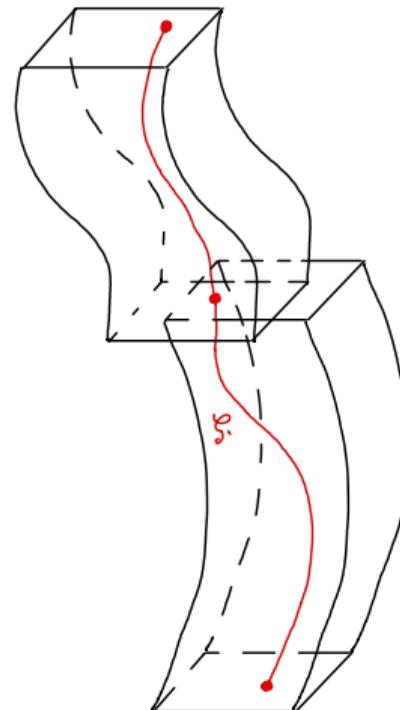
Related work

SIROCCO [Marco-Buzunariz and Rodríguez, 2016]

- Tubular neighborhoods are piecewise **linear**.
- For each strand ζ_i , computes a piecewise linear path in the tube.
- “Intuitive” (! non generic cases) algorithm on the braid with piecewise linear strands.

Algpath [G. and Lairez, 2024]

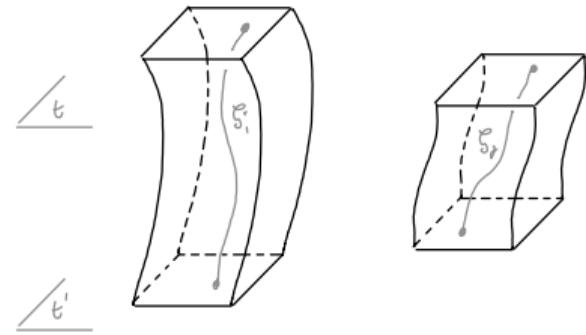
- Tubular neighborhoods are piecewise **cubic**.
- + Faster than SIROCCO.
- ! Finding a piecewise linear path in the tube requires additional work.



Strand separation interface

We assume a function $\text{sep}(i, j, t)$ that returns $t' \in (t, 1]$ and a symbol in $\star \in \{\rightarrow, \leftarrow, \rightarrow, \leftarrow\}$, such that for all $s \in [t, t']$,

- $\text{Re}(\zeta_i(s)) < \text{Re}(\zeta_j(s))$ if $\star = \rightarrow$,
- $\text{Re}(\zeta_i(s)) > \text{Re}(\zeta_j(s))$ if $\star = \leftarrow$,
- $\text{Im}(\zeta_i(s)) < \text{Im}(\zeta_j(s))$ if $\star = \rightarrow$,
- $\text{Im}(\zeta_i(s)) > \text{Im}(\zeta_j(s))$ if $\star = \leftarrow$.



$$\text{sep}(i, j, t) = (t', \rightarrow)$$

Cells

Recall: $OC_n = \{(x_1, \dots, x_n) \in \mathbb{C}^n : \forall i \neq j, x_i \neq x_j\}$.

Definition

A cell is a pair $c = (R, I)$ of relations on $\{1, \dots, n\}$.

We associate to it a topological space $|c| \subseteq OC_n$
whose points are $(x_1, \dots, x_n) \in OC_n$ such that

- for all $(i, j) \in R$, $\operatorname{Re}(x_i) < \operatorname{Re}(x_j)$,
- for all $(i, j) \in I$, $\operatorname{Im}(x_i) < \operatorname{Im}(x_j)$,

Notation

- $i \xrightarrow{c} j \iff (i, j) \in R$
- $i \xrightarrow{c} j \iff (i, j) \in I$

Cells

Recall: $OC_n = \{(x_1, \dots, x_n) \in \mathbb{C}^n : \forall i \neq j, x_i \neq x_j\}$.

Definition

A cell is a pair $c = (R, I)$ of relations on $\{1, \dots, n\}$.

We associate to it a topological space $|c| \subseteq OC_n$ whose points are $(x_1, \dots, x_n) \in OC_n$ such that

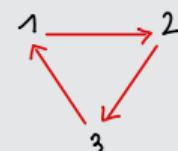
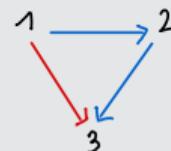
- for all $(i, j) \in R$, $\operatorname{Re}(x_i) < \operatorname{Re}(x_j)$,
- for all $(i, j) \in I$, $\operatorname{Im}(x_i) < \operatorname{Im}(x_j)$,

Notation

- $i \xrightarrow{c} j \iff (i, j) \in R$
- $i \xrightarrow{c} j \iff (i, j) \in I$

Examples

$c = (\emptyset, \emptyset)$: $|c| = OC_n$,



$(1, 2, 3 + i) \in |c| \quad |c| = \emptyset$

Properties of cells

Empty cells

A cell is empty if and only if there is a cycle in R or in I .

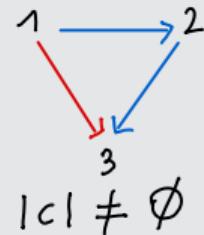
Convex cells

A (non-empty) cell is convex if and only if for all $i, j \in \{1, \dots, n\}$, either $i \rightarrow^* j$ or $j \rightarrow^* i$ or $i \rightarrow^* j$ or $j \rightarrow^* i$. We call this graph property "monochromatic semi-connectedness" (m.s.c. for short).

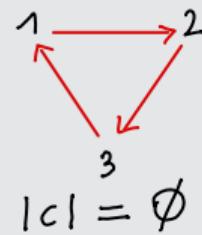
Intersection of cells

Given $c = (R, I)$ and $c' = (R', I')$ two cells, the space associated to $(R \cup R', I \cup I')$ is $|c| \cap |c'|$.

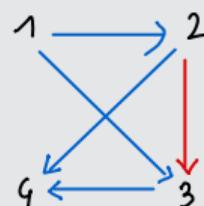
Examples



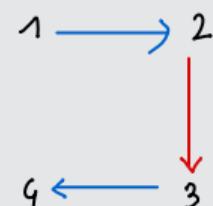
$$|c| \neq \emptyset$$



$$|c| = \emptyset$$



$|c|$ convex

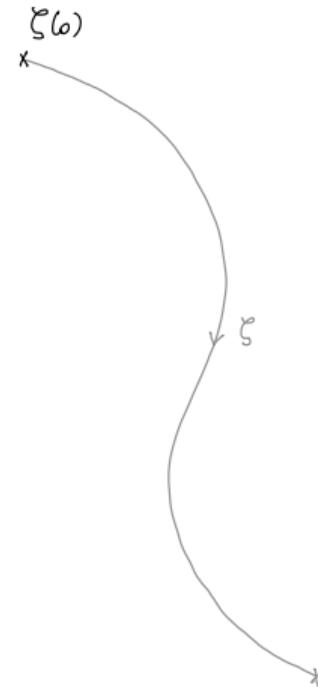


$|c|$ not convex

Step 1: compute a sequence of cells

Path to cells

Input: ζ (represented by tubular neighborhoods).



Step 1: compute a sequence of cells

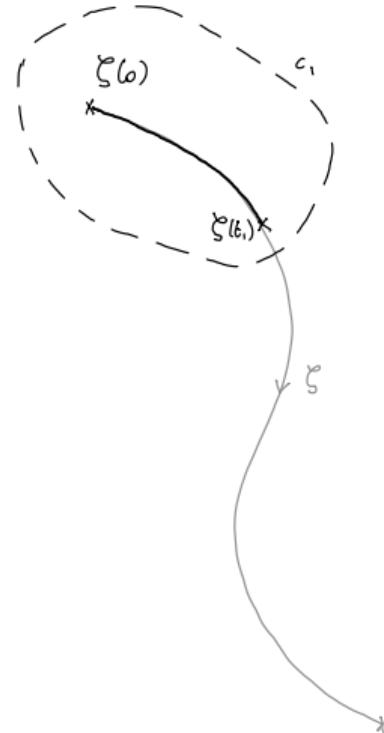
Path to cells

Input: ζ (represented by tubular neighborhoods).

Output: a sequence of convex cells c_1, \dots, c_r such that there exists $0 = t_0 < \dots < t_r = 1$ and for any $s \in [t_{i-1}, t_i]$, $\zeta(s) \in c_i$.

Idea:

- Start with an initial convex cell c containing $\zeta(0)$.
- Associate to each edge a time of validity.
- When a relation expires, update it using sep and repair convexity.
- Repeat.



Step 1: compute a sequence of cells

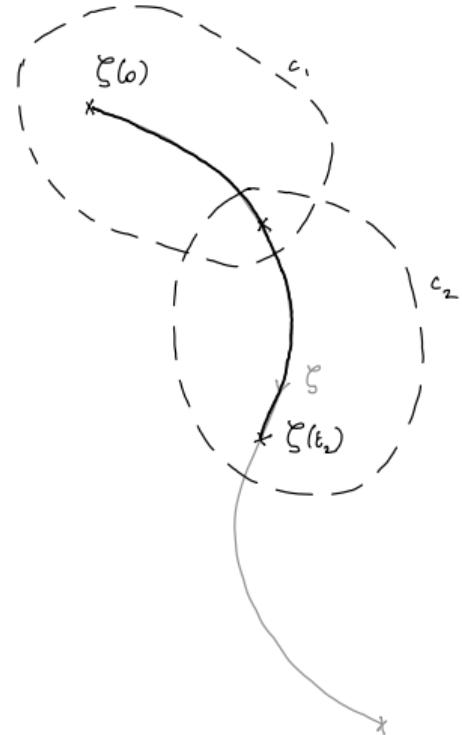
Path to cells

Input: ζ (represented by tubular neighborhoods).

Output: a sequence of convex cells c_1, \dots, c_r such that there exists $0 = t_0 < \dots < t_r = 1$ and for any $s \in [t_{i-1}, t_i]$, $\zeta(s) \in c_i$.

Idea:

- Start with an initial convex cell c containing $\zeta(0)$.
- Associate to each edge a time of validity.
- When a relation expires, update it using sep and repair convexity.
- Repeat.



Step 1: compute a sequence of cells

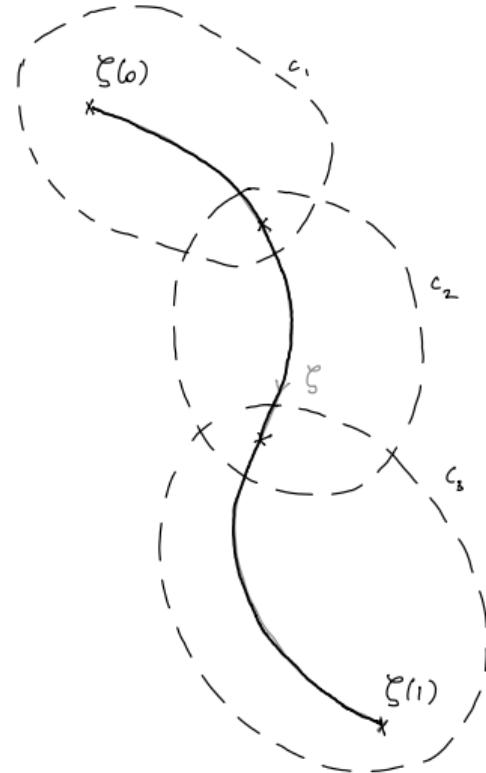
Path to cells

Input: ζ (represented by tubular neighborhoods).

Output: a sequence of convex cells c_1, \dots, c_r such that there exists $0 = t_0 < \dots < t_r = 1$ and for any $s \in [t_{i-1}, t_i]$, $\zeta(s) \in c_i$.

Idea:

- Start with an initial convex cell c containing $\zeta(0)$.
- Associate to each edge a time of validity.
- When a relation expires, update it using sep and repair convexity.
- Repeat.



Step 2: linearize ζ

Definition

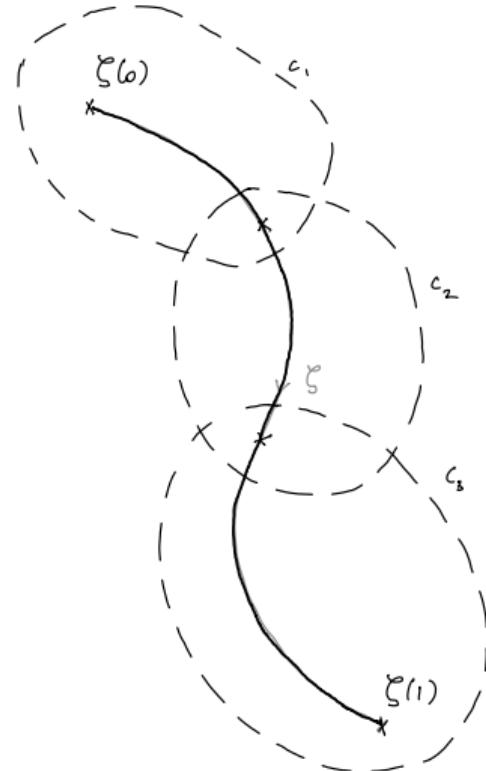
Let $\pi, \varphi \in \mathfrak{S}_n$. We define

$$p_{\pi, \varphi} = (\pi(1) + i\varphi(1), \dots, \pi(n) + i\varphi(n)) \in OC_n.$$

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{pmatrix}$$

$$\varphi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 1 & 3 \end{pmatrix}$$

$$P_{\pi, \varphi} e : \quad \begin{matrix} \cdot & \bullet_1 & \cdot & \cdot \\ \cdot & \cdot & \bullet_4 & \cdot \\ \bullet_2 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \bullet_3 \end{matrix}$$



Step 2: linearize ζ

Definition

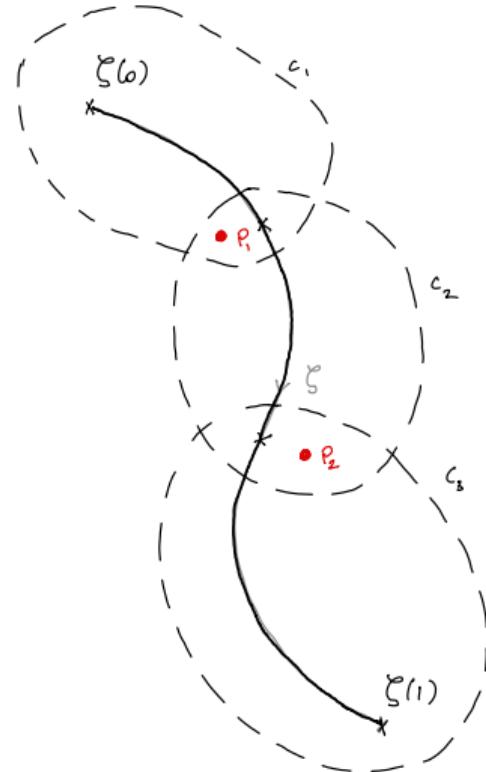
Let $\pi, \varphi \in \mathfrak{S}_n$. We define

$$p_{\pi, \varphi} = (\pi(1) + i\varphi(1), \dots, \pi(n) + i\varphi(n)) \in OC_n.$$

$$\begin{array}{l} \pi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{pmatrix} \quad p_{\pi, e} : \begin{matrix} \cdot & \bullet_1 & \cdot & \cdot \\ \cdot & \cdot & \bullet_4 & \cdot \\ \bullet_2 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \bullet_3 \end{matrix} \\ e = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 1 & 3 \end{pmatrix} \end{array}$$

Linearization of ζ

For each c_i, c_{i+1} , we compute π, φ such that $p_i = p_{\pi, \varphi}$ lies in the intersection $c_i \cap c_{i+1}$ (Hint: total order extending R and I).



Step 2: linearize ζ

Definition

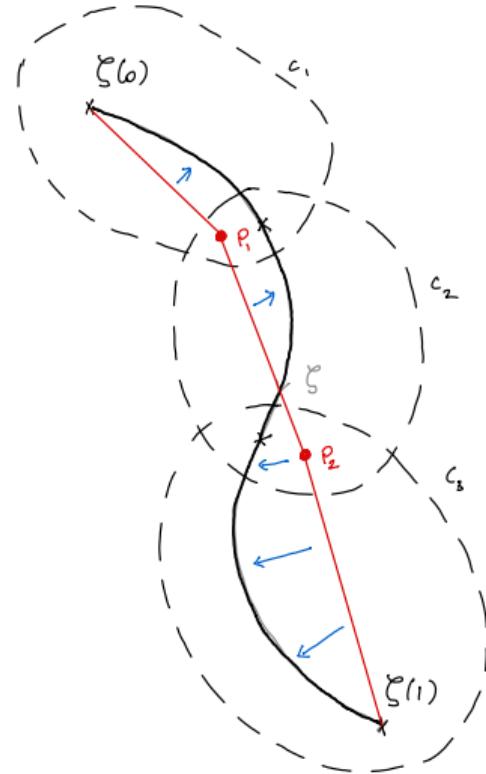
Let $\pi, \varphi \in \mathfrak{S}_n$. We define

$$p_{\pi, \varphi} = (\pi(1) + i\varphi(1), \dots, \pi(n) + i\varphi(n)) \in OC_n.$$

$$\begin{aligned}\pi &= \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{pmatrix} & p_{\pi, \varphi} : & \begin{array}{c} \cdot \quad \bullet_1 \quad \cdot \quad \cdot \\ \cdot \quad \cdot \quad \cdot \quad \bullet_4 \\ \bullet_2 \quad \cdot \quad \cdot \quad \cdot \\ \cdot \quad \cdot \quad \cdot \quad \bullet_3 \end{array} \\ \varphi &= \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 1 & 3 \end{pmatrix}\end{aligned}$$

Linearization of ζ

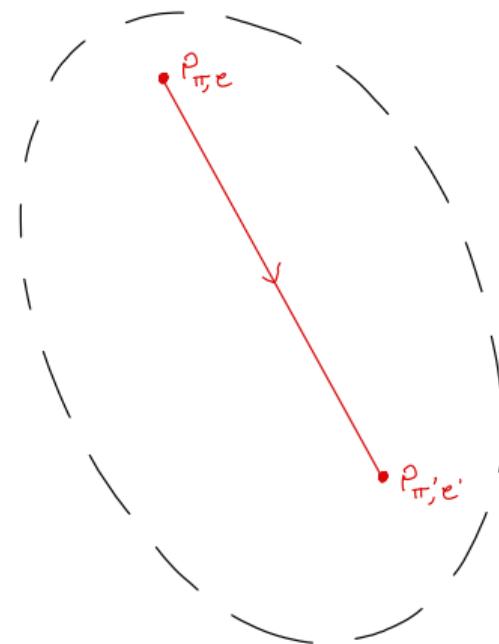
For each c_i, c_{i+1} , we compute π, φ such that $p_i = p_{\pi, \varphi}$ lies in the intersection $c_i \cap c_{i+1}$ (Hint: total order extending R and I). The linear interpolation of the p_i is homotopic to ζ . Why? cells are convex!



Step 3: decomposition of the linearization

Reduction

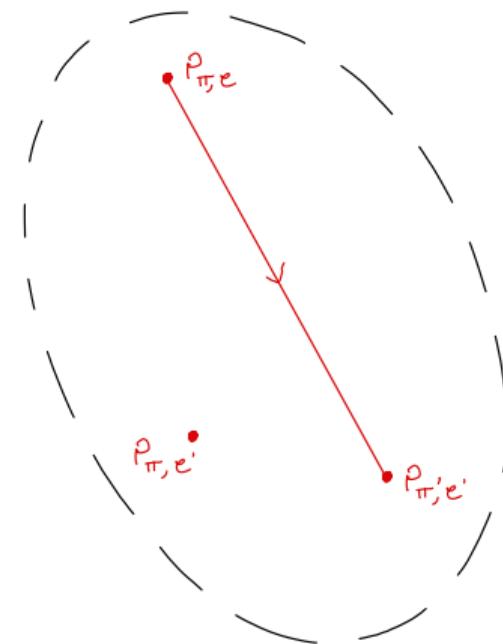
- Computing the braid associated to the whole linearization or to each piece and concatenating the results is equivalent.



Step 3: decomposition of the linearization

Reduction

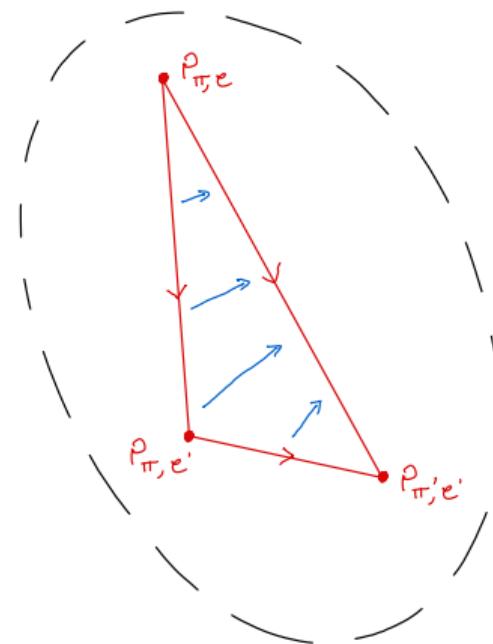
- Computing the braid associated to the whole linearization or to each piece and concatenating the results is equivalent.
- Assume $p_{\pi,\varphi}$ and $p_{\pi',\varphi'}$ both lie in a convex cell $c = (R, I)$. It means that π, π' extend R and φ, φ' extend I . **So $p_{\pi,\varphi'}$ also lies in c !**



Step 3: decomposition of the linearization

Reduction

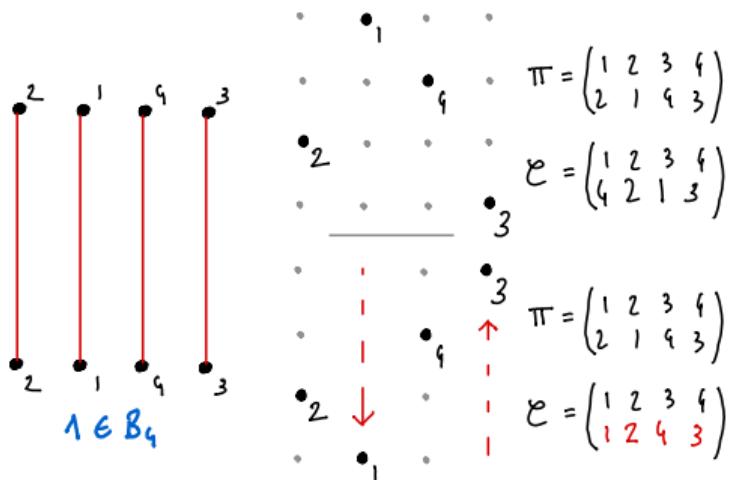
- Computing the braid associated to the whole linearization or to each piece and concatenating the results is equivalent.
- Assume $p_{\pi,\varphi}$ and $p_{\pi',\varphi'}$ both lie in a convex cell $c = (R, I)$. It means that π, π' extend R and φ, φ' extend I . **So $p_{\pi,\varphi'}$ also lies in c !**
- We compute the braid of $p_{\pi,\varphi} \rightarrow p_{\pi,\varphi'}$ then the braid of $p_{\pi,\varphi'} \rightarrow p_{\pi',\varphi'}$.



Step 3: decomposition of the linearization

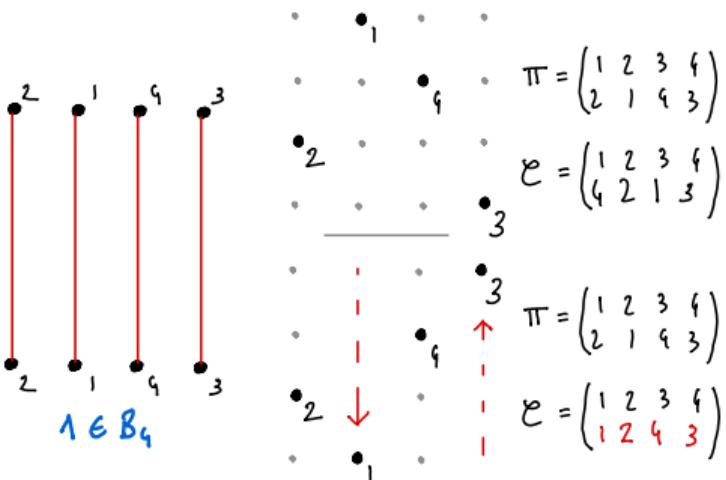
$$\begin{array}{cccc} \bullet_2 & \bullet_1 & \bullet_4 & \bullet_3 \\ \bullet_2 & \bullet_1 & \bullet_4 & \bullet_3 \\ \hline \bullet_2 & \bullet_1 & \bullet_4 & \bullet_3 \\ \bullet_2 & \bullet_1 & \bullet_4 & \bullet_3 \\ \hline \bullet_2 & \bullet_1 & \bullet_4 & \bullet_3 \end{array}$$
$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{pmatrix}$$
$$e = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 1 & 3 \end{pmatrix}$$
$$\begin{array}{cccc} \bullet_2 & \bullet_1 & \bullet_4 & \bullet_3 \\ \bullet_2 & \bullet_1 & \bullet_4 & \bullet_3 \\ \hline \bullet_2 & \bullet_1 & \bullet_4 & \bullet_3 \end{array}$$
$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{pmatrix}$$
$$e = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 4 & 3 \end{pmatrix}$$

Step 3: decomposition of the linearization

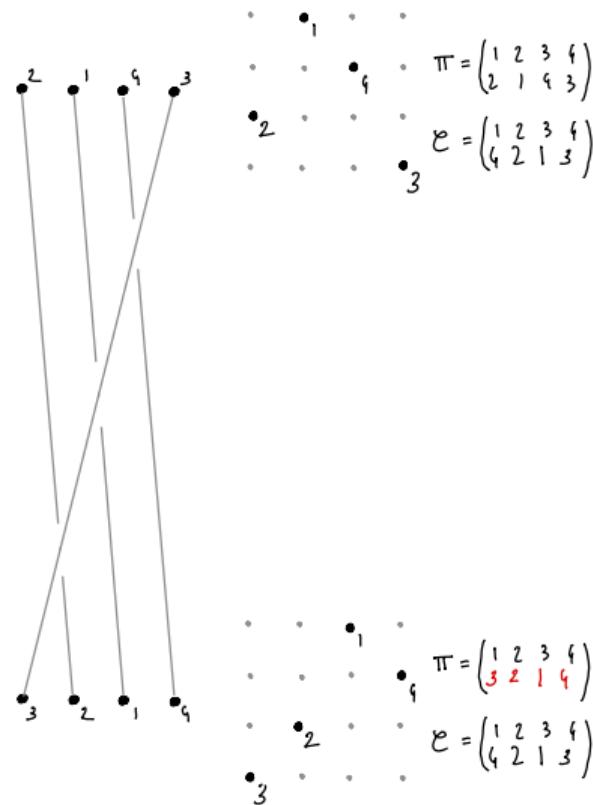


$p_{\pi, \varphi} \rightarrow p_{\pi, \varphi'}$: trivial braid.

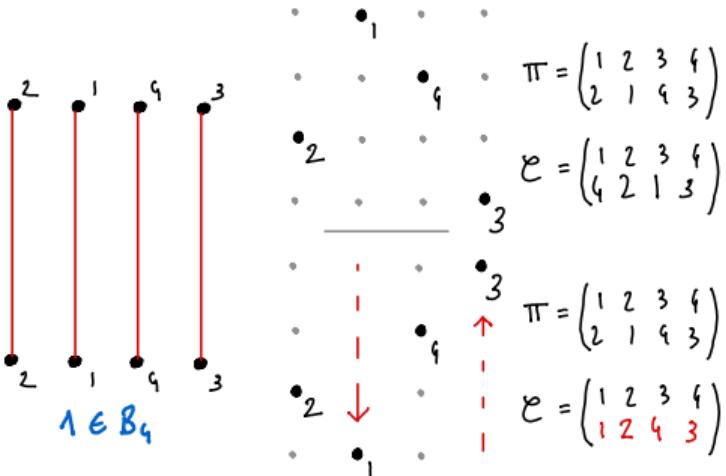
Step 3: decomposition of the linearization



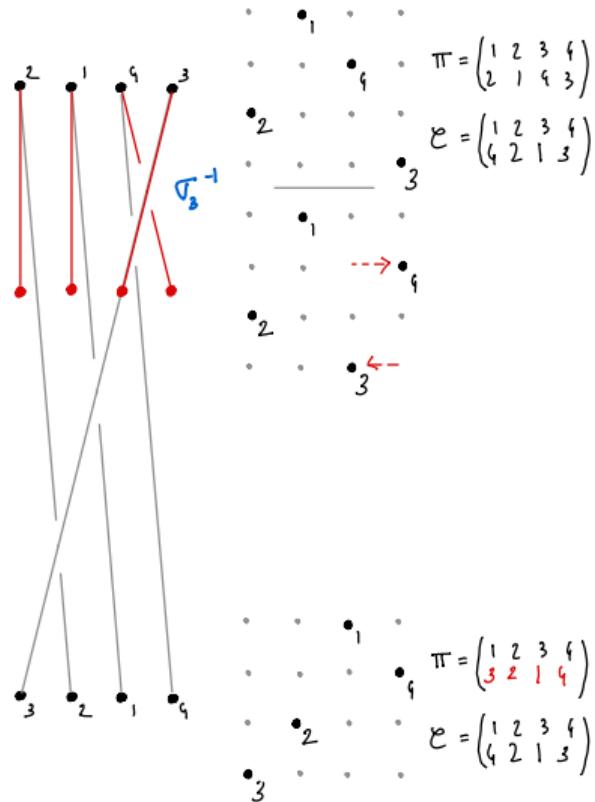
$p_{\pi, \varphi} \rightarrow p_{\pi, \varphi'}$: trivial braid.



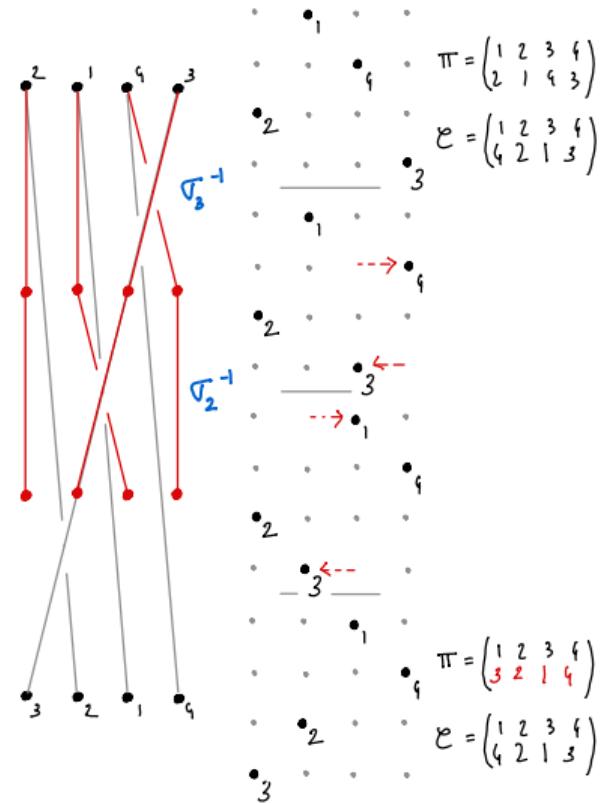
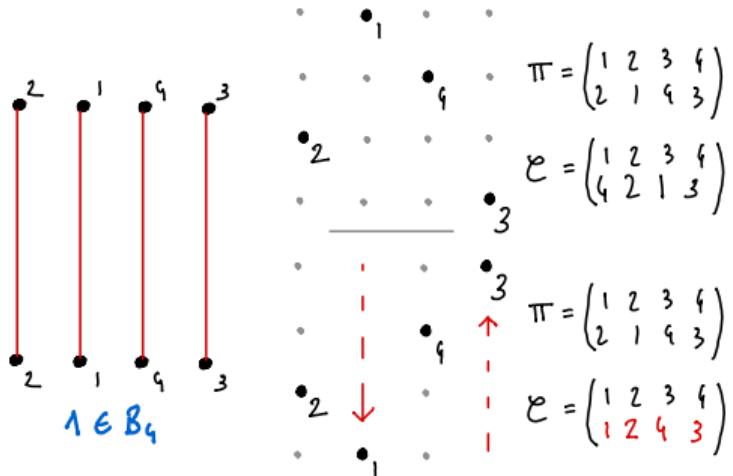
Step 3: decomposition of the linearization



$p_{\pi, \varphi} \rightarrow p_{\pi, \varphi'}$: trivial braid.

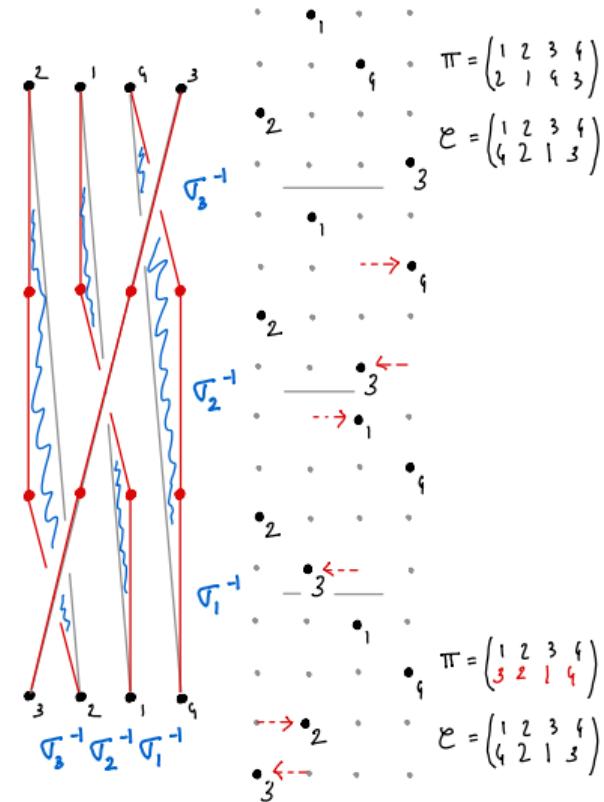
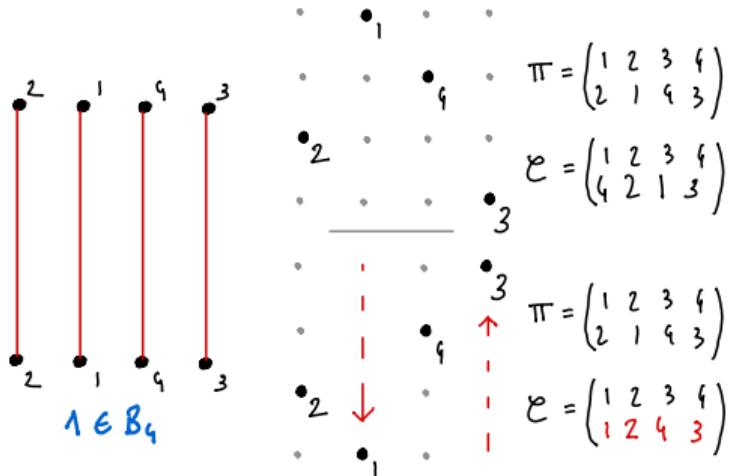


Step 3: decomposition of the linearization



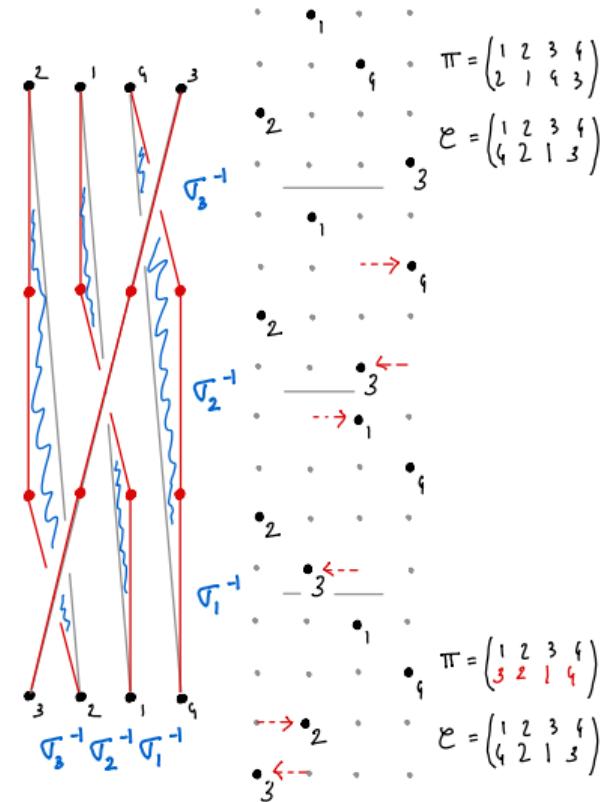
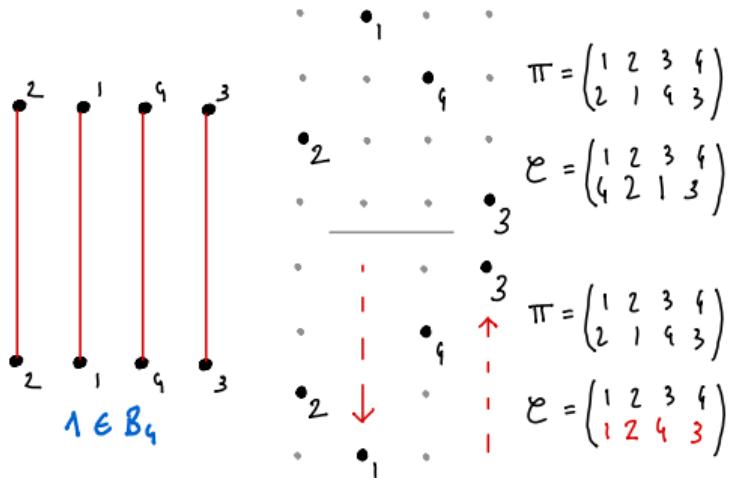
$p_{\pi, \varphi} \rightarrow p_{\pi, \varphi'}$: trivial braid.

Step 3: decomposition of the linearization



$p_{\pi, \varphi} \rightarrow p_{\pi, \varphi'}$: trivial braid.

Step 3: decomposition of the linearization



$p_{\pi, \varphi} \rightarrow p_{\pi, \varphi'}$: trivial braid.

$p_{\pi, \varphi'} \rightarrow p_{\pi', \varphi'}$: Decompose $\pi' \pi^{-1} = s_{i_1} \cdots s_{i_r}$ in elementary transpositions. Output $\sigma_{i_1}^{\varepsilon_1} \cdots \sigma_{i_r}^{\varepsilon_r}$ with $\varepsilon_1, \dots, \varepsilon_r \in \{\pm 1\}$ computed using φ' .

Conclusion

```
~/2025/code/braid_group cargo run --release
```

```
Finished `release` profile [optimized] target(s) in 0.08s
Running `target/release/braid_group`
```

```
05 7 0 1 0 2 5 0 2 9 0 4 7 0 5 1 0 5 5 0 6 1 0 6 3 0 8 3 0 3 0 5 0 4 1 0 3 1 0 3 5 0 0 - 1 0 1 0 0 0 3 7 0 7 3 0 9 7 0 9 8 0 2 7 0 4 9 0 8 7 0 9 7 - 1 0 5 9 0 1 5 0 2 0 3 - 1 0 9 6 - 1 0 7 7 0 9 2 - 1 0 9 3 - 1 0 8 1 0 9 2 0 2 4 0 7 2 - 1 0 8 4 0 1 8 - 1 0 2 6 0 9 4 - 1 0 9 5 - 1 0 1 0 0 0 8 8 0 8 7 - 1 0 1 7 - 1 0 2 4 - 1 0 1 6 - 1 0 2 5 0 2 4 0 8 2 0 1 5 - 1 0 6 3 - 1 0 8 6 - 1 0 1 3 0 1 4 - 1 0 1 3 - 1 0 6 - 1 0 8 1 - 1 0 6 5 0 8 7 - 1 0 1 2 - 1 0 7 3 - 1 0 1 1 - 1 0 8 8 - 1 0 9 6 - 1 0 9 3 0 9 4 0 4 0 5 - 1 0 2 7 - 1 0 1 0 - 1 0 6 6 0 6 2 0 7 1 0 7 4 - 1 0 6 - 1 0 7 0 6 0 4 - 1 0 1 5 0 9 - 1 0 9 3 0 2 3 0 1 6 - 1 0 9 2 - 1 0 8 - 1 0 2 0 - 1 0 7 - 1 0 7 5 - 1 0 6 - 1 0 5 - 1 0 4 - 1 0 9 1 0 2 2 0 2 1 0 8 9 - 1 0 2 0 0 7 3 0 8 5 0 9 0 - 1 0 2 2 0 9 1 - 1 0 3 - 1 0 9 2 - 1 0 6 0 0 9 3 - 1 0 8 2 - 1 0 8 3 - 1 0 2 - 1 0 8 2 0 9 4 - 1 0 9 5 - 1 0 1 2 - 1 0 6 7 0 0 - 1 0 1 - 1 0 0 - 1 0 9 6 - 1 0 3 5 - 1 0 6 8 0 7 6 - 1 0 7 7 - 1 0 9 7 - 1 0 1 4 0 1 5 - 1 0 1 4 - 1 0 9 8 - 1 0 6 0 0 9 4 0 9 2 - 1 0 9 1 0 7 8 - 1 0 1 3 - 1 0 1 4 0 7 0 - 1 0 6 9 0 7 0 0 5 9 - 1 0 2 1 0 8 3 0 7 9 - 1 0 8 0 - 1 0 9 2 0 7 1 0 1 5 0 2 3 - 1 0 1 7 - 1 0 9 - 1 0 1 8 0 1 0 0 1 9 0 8 1 - 1 0 1 8 0 9 3 0 9 2 - 1 0 1 7 0 8 4 0 8 3 - 1 0 8 2 - 1 0 8 3 - 1 0 8 4 - 1 0 1 6 0 7 2 0 7 9 - 1 0 1 2 - 1 0 7 6 - 1 0 1 3 0 8 5 - 1 0 7 3 0 8 6 - 1 0 3 6 0 7 4 0 8 1 0 8 7 - 1 0 1 5 0 8 9 - 1 0 1 4 0 8 7 0 1 3 0 1 2 0 7 - 1 0 1 8 0 2 8 0 9 0 - 1 0 9 1 - 1 0 7 8 0 9 2 - 1 0 9 3 - 1 0 9 4 - 1 0 9 5 - 1 0 9 8 0 0 9 7 0 9 2 - 1 0 9 6 - 1 0 2 4 0 9 8 - 1 0 2 4 0 9 8 - 1 0 9 4 0 7 5 0 2 9 - 1 0 1 5 0 7 6 0 8 8 0 8 7 - 1 0 3 3 0 9 5 0 9 4 0 9 3 0 9 2 0 1 1 0 7 7 0 7 8 0 6 8 0 6 7 - 1 0 9 1 0 9 0 0 7 9 0 8 0 1 0 0 0 9 1 0 9 1 - 1 0 8 9 0 4 0 8 1 0 8 2 0 4 7 0 8 8 0 2 0 0 2 1 - 1 0 8 3 0 8 1 0 8 4 0 8 5 0 7 6 - 1 0 8 6 0 2 0 1 0 9 4 0 8 7 0 7 7 - 1 0 7 5 0 7 6 0 8 8 0 8 9 0 7 5 - 1 0 9 0 0 9 1 0 6 6 0 6 5 - 1 0 2 2 - 1 0 8 5 - 1 0 8 4 0 9 2 0 9 3 0 9 4 0 9 5 0 2 3 0 9 6 0 9 7 0 2 4 0 8 2 - 1 0 8 3 0 8 2 0 1 0 9 0 2 0 - 1 0 3 4 - 1 0 3 5 0 3 6 0 8 1 - 1 0 7 4 - 1 0 7 3 0 7 2 0 2 5 0 6 3 0 6 4 0 6 3 - 1 0 7 3 - 1 0 7 4 0 2 6 - 1 0 3 7 - 1 0 3 8 - 1 0 3 9 0 6 2 0 6 1 0 6 0 - 1 0 5 9 - 1 0 6 0 0 6 1 - 1 0 2 3 - 1 0 1 2 - 1 0 1 3 0 5 8 0 5 7 - 1 0 4 0 0 4 1 - 1 0 2 2 0 1 4 0 1 5 - 1 0 4 2 - 1 0 4 3 0 5 6 0 5 5 - 1 0 5 4 - 1 0 5 3 0 4 4 - 1 0 7 2 - 1 0 7 3 0 4 5 0 2 6 0 2 5 - 1 0 8 4 - 1 0 7 4 - 1 0 7 5 0 8 6 0 7 6 - 1 0 7 7 0 5 2 0 5 1 - 1 0 3 8 0 8 5 0 5 0 0 4 9 - 1 0 2 4 - 1 0 2 3 0 4 6 - 1 0 4 7 0 7 8 - 1 0 7 9 0 9 0 8 0 0 8 4 0 4 8 0 7 8 0 8 2 - 1 0 8 0 0 3 0 - 1 0 3 4 0 5 2 - 1 0 7 4 0 7 6 0 9 0 - 1 0 2 3 - 1 0 2 4 0 3 6 - 1 0 7 9 0 7 8 - 1 0 4 - 1 0 2 1 0 4 6 0 4 7 0 7 0 - 1 0 4 6 - 1 0 2 8 - 1 0 6 8 - 1 0 6 9 0 2 0 6 0 2 3 - 1 0 5 4 0 6 6 - 1 0 8 8 - 1 0 9 4 - 1 0 4 2 0 3 2 - 1 0 4 9 0 5 0 - 1 0 3 8 - 1 0 1 0 0 2 6 - 1 0 5 6 - 1 0 5 8 - 1 0 7 2 0 4 0 - 1 0 4 4 0 5 1 - 1 0 8 5 0 6 0 - 1 0 5 2 0 1 2 0 9 6 - 1 0 4 5 0 4 4 - 1 0 2 4 0 2 3 - 1 0 1 7 0 5 3 0 5 4 - 1 0 2 3 0 2 4 - 1 0 7 7 0 7 6 - 1 0 8 6 0 7 5 0 8 4 - 1 0 4 2 0 4 3 0 4 2 - 1 0 5 5 - 1 0 5 6 0 6 2 - 1 0 7 4 - 1 0 2 5 - 1 0 2 6 0 7 3 0 7 2 - 1 0 5 7 - 1 0 5 8 0 3 8 0 6 0 0 6 1 0 6 0 - 1 0 5 9 - 1 0 2 2 0 4 9 - 1 0 4 1 - 1 0 1 5 - 1 0 1 4 0 2 3 - 1 0 6 0 0 6 1 - 1 0 4 0 0 1 3 0 1 2 - 1 0 8 5 - 1 0 6 2 0 3 9 0 3 8 - 1 0 7 4 0 7 3 - 1 0 3 7 - 1 0 6 0 - 1 0 6 3 0 6 4 - 1 0 3 6 0 3 5 0 3 4 - 1 0 8 1 - 1 0 6 5 - 1 0 6 6 0 7 2 0 2 6 - 1 0 2 5 0 8 2 0 8 3 0 2 4 0 2 5 - 1 0 2 6 0 8 2 - 1 0 8 4 - 1 0 7 3 0 7 4 - 1 0 3 2 0 3 3 0 3 2 - 1 0 3 3 - 1 0 2 3 0 2 2 - 1 0 1 9 - 1 0 2 0 - 1 0 1 9 0 9 7 0 9 6 0 3 1 - 1 0 9 5 0 9 4 0 9 3 0 9 2 0 8 5 0 3 0 6 7 - 1 0 6 8 0 8 - 1 0 2 1 - 1 0 2 0 0 9 1 0 9 0 0 7 5 - 1 0 7 6 0 7 5 0 8 7 0 7 7 - 1 0 8 8 0 7 6 - 1 0 8 9 0 8 8 0 8 7 0 8 6 0 8 5 0 8 1 0 4 7 - 1 0 9 8 - 1 0 9 7 - 1 0 2 9 - 1 0 2 8 0 9 6 - 1 0 8 4 0 9 4 0 9 5 - 1 0 9 4 - 1 0 9 3 - 1 0 8 3 0 8 2 0 8 1 0 9 1 0 9 2 - 1 0 9 1 - 1 0 9 0 - 1 0 7 8 0 8 8 0 8 9 - 1 0 8 8 - 1 0 8 7 - 1 0 8 2 - 1 0 1 0 2 0 5 9 - 1 0 6 0 0 7 7 0 1 8 0 1 7 - 1 0 8 5 - 1 0 8 4 - 1 0 8 3 - 1 0 8 2 - 1 0 7 8 - 1 0 3 0 4 0 8 8 0 8 0 0 8 9 - 1 0 1 8 - 1 0 1 9 - 1 0 1 8 0 1 9 0 6 9 - 1 0 7 9 0 8 4 0 1 6 - 1 0 7 8 0 9 6 0 7 7 0 7 0 0 8 1 - 1 0 5 0 6 0 8 0 - 1 0 7 0 8 0 0 - 1 0 1 - 1 0 9 4 - 1 0 7 9 - 1 0 7 5 - 1 0 7 6 0 7 5 0 7 8 - 1 0 1 7 - 1 0 1 5 0 2 - 1 0 3 - 1 0 1 1 0 1 0 - 1 0 9 0 - 1 0 7 6 - 1 0 1 1 0 1 6 0 4 - 1 0 5 - 1 0 2 2 - 1 0 6 - 1 0 7 - 1 0 7 4 0 7 3 0 1 4 0 1 3 - 1 0 1 4 - 1 0 1 5 - 1 0 1 4 0 8 - 1 0 7 5 - 1 0 7 4 - 1 0 9 - 1 0 1 0 - 1 0 2 3 - 1 0 7 2 0 2 1 0 2 6 - 1 0 1 2 0 1 3 0 7 1 0 2 2 0 7 3 - 1 0 6 7 0 7 0 0 6 9 0 7 0 2 5 0 8 9 - 1 0 7 2 - 1 0 1 4 0 1 5 0 9 3 - 1 0 9 2 - 1 0 9 4 0 9 3 0 9 2 0 7 1 - 1 0 1 6 0 7 0 - 1 0 1 7 0 8 8 - 1 0 9 5 0 9 6 - 1 0 1 1 - 1 0 1 2 - 1 0 1 5 - 1 0 2 7 - 1 0 1 3 - 1 0 1 4 - 1 0 1 5 - 1 0 1 6 - 1 0 9 4 0 9 5 - 1 0 1 8 0 1 9 0 8 6 0 1 5 - 1 0 2 6 0 6 8 0 6 7 0 1 1 - 1 0 1 3 0 8 5 0 6 9 - 1 0 8 7 - 1 0 8 3 0 1 7 - 1 0 8 6 - 1 0 1 8 - 1 0 8 2 - 1 0 8 3 0 4 - 1 0 5 - 1 0 6 - 1 0 9 4 - 1 0 8 1 - 1 0 8 2 0 4 0 5 0 6 0 7 0 9 1 0 9 3 - 1 0 1 0 0 9 2 - 1 0 8 8 0 2 1 - 1 0 8 9 - 1 0 2 3 0 2 0 0 2 1 0 2 2 0 2 1 0 2 0 0 8 7 0 8 8 - 1 0 2 3 0 1 9 - 1 0 6 - 1 0 2 0 - 1 0 2 1 - 1 0 2 2 - 1 0 5 - 1 0 4 0 9 3 - 1 0 2 0 0 1 9 - 1 0 2 0 - 1 0 9 4 0 6 5 0 1 3 0 9 5 0 9 6 - 1 0 2 0 3 0 2 - 1 0 1 0 - 1 0 3 - 1 0 7 - 1 0 9 - 1 0 1 3 - 1 0 1 7 - 1 0 1 9 - 1 0 2 1 - 1 0 2 3 - 1 0 3 3 - 1 0 3 5 - 1 0 3 9 - 1 0 4 3 - 1 0 4 5 - 1 0 5 3 - 1 0 6 3 - 1 0 6 5 - 1 0 6 7 - 1 0 6 9 - 1 0 7 1 - 1 0 7 5 - 1 0 8 5 - 1 0 8 7 - 1 0 8 9 - 1 0 9 1 - 1 0 9 5 - 1 0 9 7 - 1 0 9 8
```