

# Braid monodromy computations using certified path tracking

---

Alexandre Guillemot

Joint work with Pierre Lairez

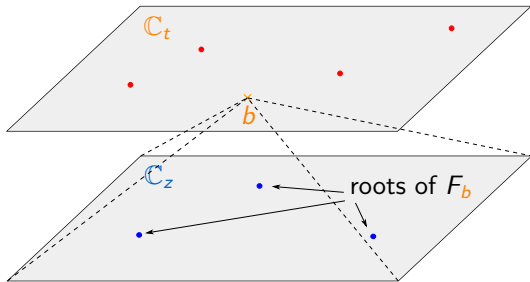
MATHEXP, Inria, France

Journées de géométrie algorithmique

October 14, 2025 | Roscoff



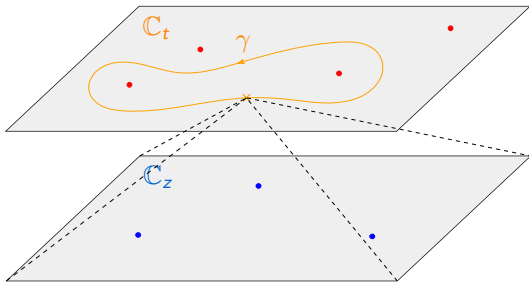
# Motivation



## 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,

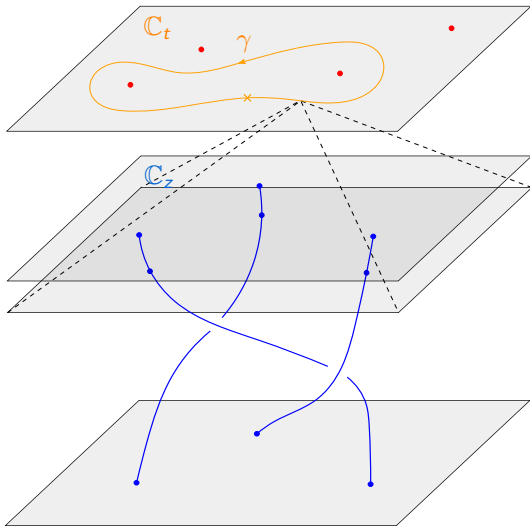
# Motivation



## 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  $b$ .

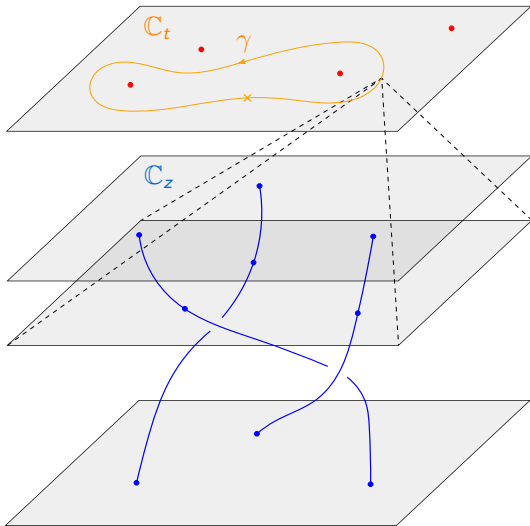
# Motivation



## 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  $\gamma$  defines a braid.

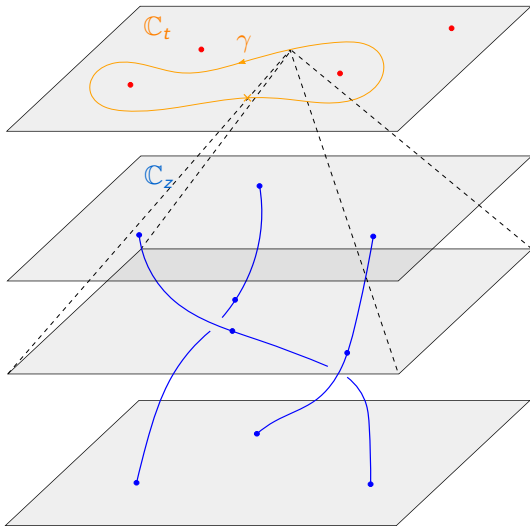
# Motivation



## 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  $\gamma$  defines a braid.

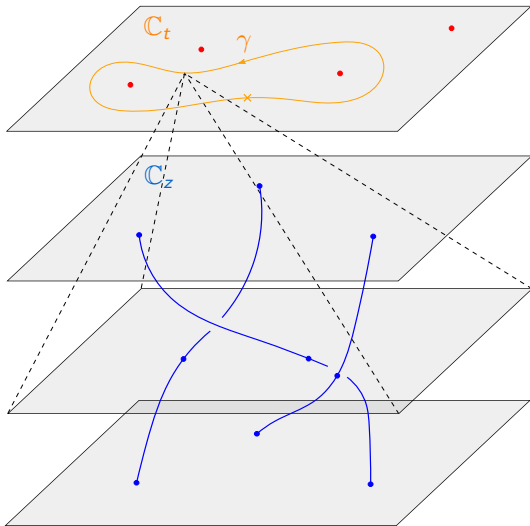
# Motivation



## 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  $\gamma$  defines a braid.

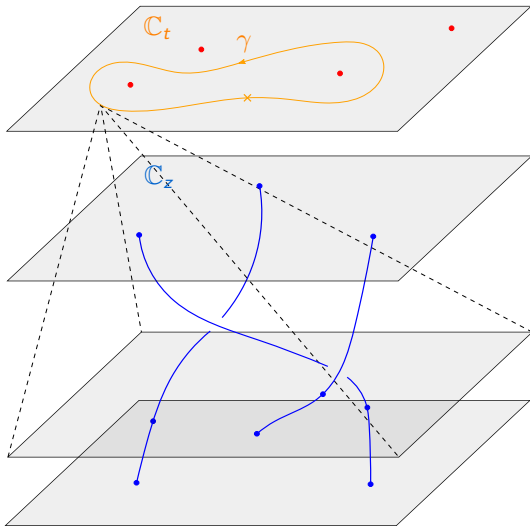
# Motivation



## 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  $\gamma$  defines a braid.

# Motivation

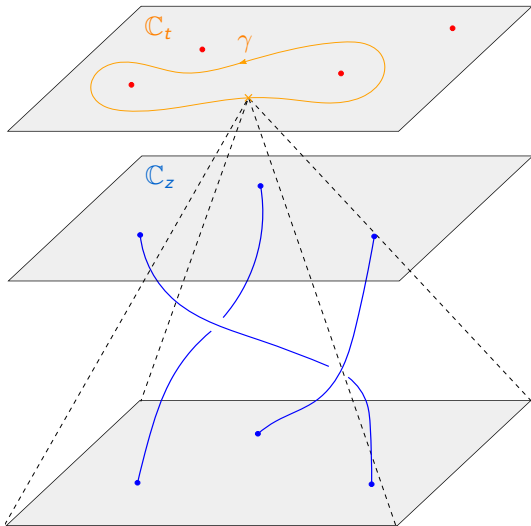


## Setup

- 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  $\gamma$  defines a braid.



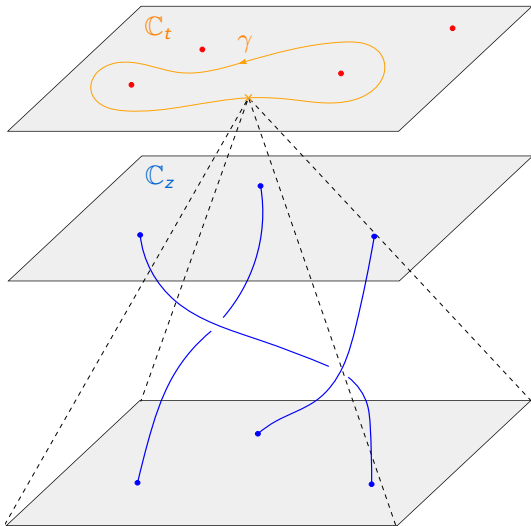
# Motivation



## 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  $\gamma$  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  $\gamma$  defines a braid.

## Algorithmic goal

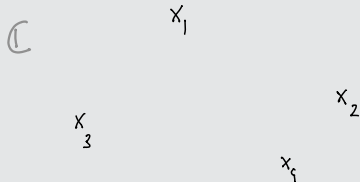
**Input:**  $g, \gamma$

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

# Configurations

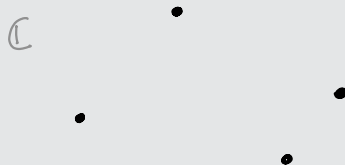
## Ordered configurations

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



## 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

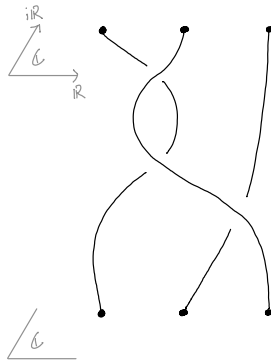
Homotopy class of a path  $\beta : [0, 1] \rightarrow C_n$  such that  $\beta(0) = \beta(1) = \{1, \dots, n\}$ .



# Braid group

## Braid

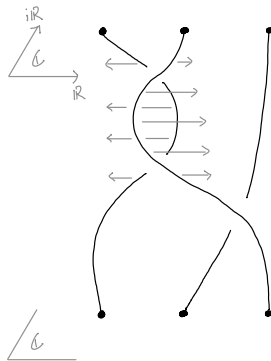
Homotopy class of a path  $\beta : [0, 1] \rightarrow C_n$  such that  $\beta(0) = \beta(1) = \{1, \dots, n\}$ .



# Braid group

## Braid

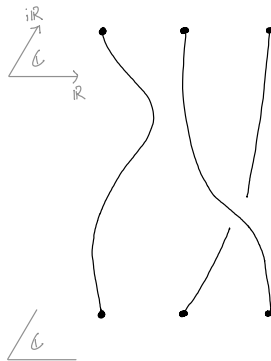
Homotopy class of a path  $\beta : [0, 1] \rightarrow C_n$  such that  $\beta(0) = \beta(1) = \{1, \dots, n\}$ .



# Braid group

## Braid

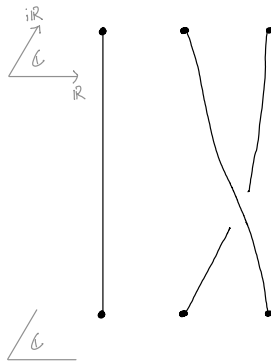
Homotopy class of a path  $\beta : [0, 1] \rightarrow C_n$  such that  $\beta(0) = \beta(1) = \{1, \dots, n\}$ .



# Braid group

## Braid

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



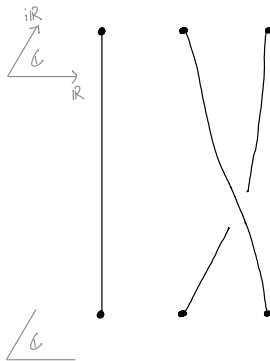


# Braid group

## 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  $\beta(0) = \beta(1) = \{1, \dots, n\}$ .

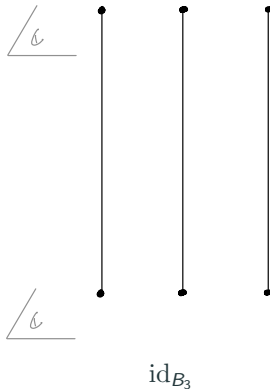
In practice, we will manipulate paths in  $OC_n$ .

## 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  $\pi_1(C_n, \{1, \dots, n\})$ .



# Braid group

## Braid

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

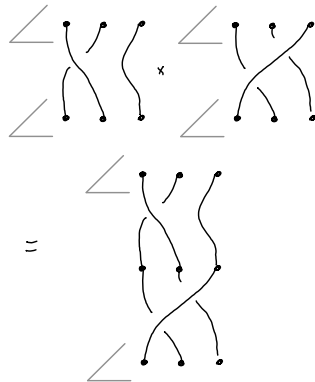
In practice, we will manipulate paths in  $OC_n$ .

## 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  $\beta(0) = \beta(1) = \{1, \dots, n\}$ .

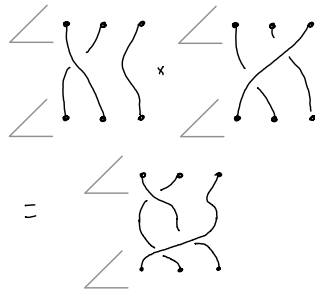
In practice, we will manipulate paths in  $OC_n$ .

## Braid group $B_n$

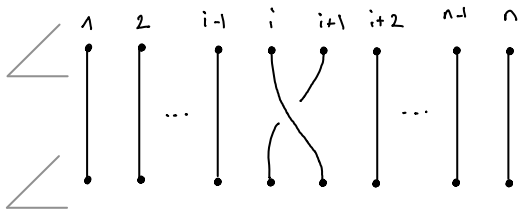
id: class of the constant path equal to  $\{1, \dots, n\}$ .

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

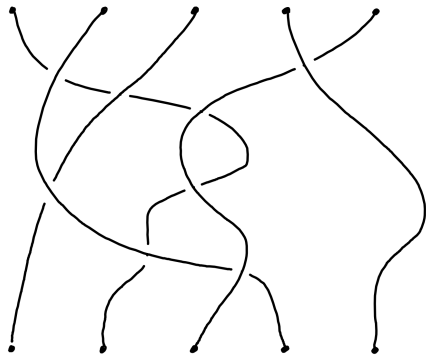
Rk: this is  $\pi_1(C_n, \{1, \dots, n\})$ .



## Artin's theorem



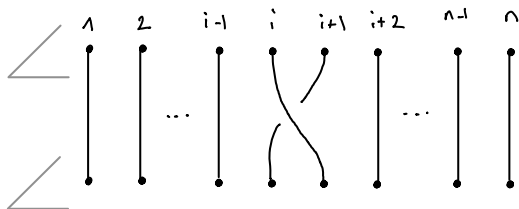
$$\sigma_i \in B_n$$



## Theorem [Artin, 1947]

The  $\sigma_i$ 's generate  $B_n$  (+ explicit relations).

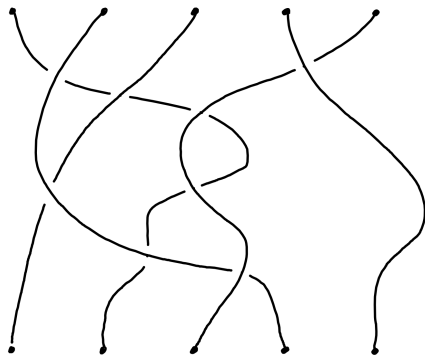
# Artin's theorem



$$\sigma_i \in B_n$$

## Theorem [Artin, 1947]

The  $\sigma_i$ 's generate  $B_n$  (+ explicit relations).



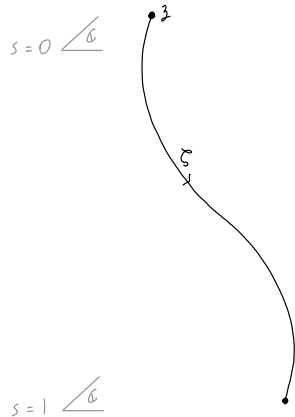
$$\sigma_4 \sigma_1^{-1} \sigma_2^{-1} \sigma_3^{-1} \sigma_3 \sigma_1 \sigma_2 \sigma_3^{-1}$$

# 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.*



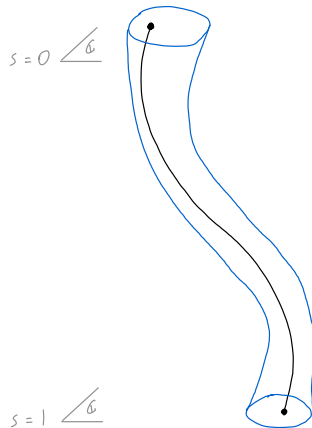
# Main tool

## 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  $\zeta$ .





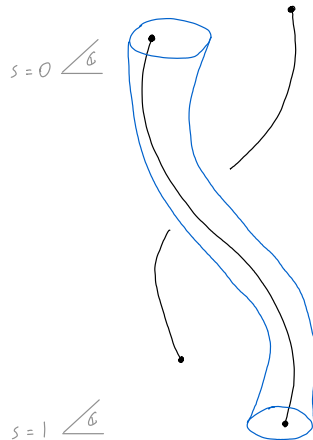
# Main tool

## 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  $\zeta$ .



# Main tool

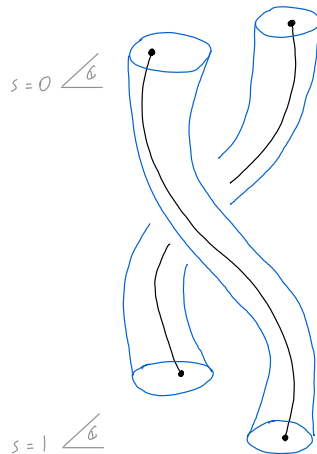
## Certified homotopy continuation

**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  $\zeta$ .

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  $\zeta$ .

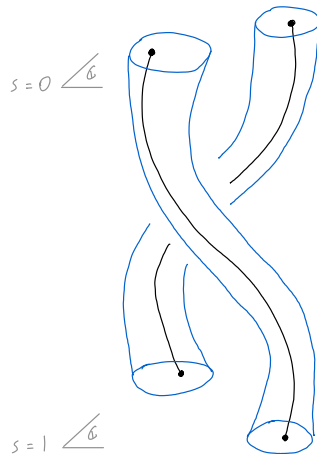
We can do that for every solution at  $s = 0$ .

## 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 :**  $\zeta$  ( $n$  disjoint tubular neighborhoods around  $\zeta_1, \dots, \zeta_n$ ).

**Output :** A decomposition in standard generators of the braid induced by  $\zeta_1, \dots, \zeta_n$ .

# 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.  $\Phi(\zeta(0)) = \Phi(\zeta(1))$ .

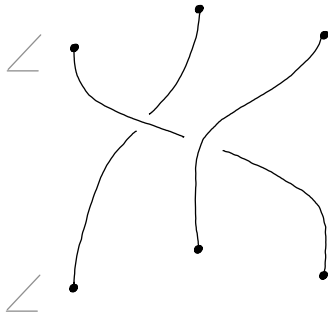
## Goal

**Input :**  $\zeta$  ( $n$  disjoint tubular neighborhoods around  $\zeta_1, \dots, \zeta_n$ ).

**Output :** A decomposition in standard generators of the braid induced by  $\zeta_1, \dots, \zeta_n$ .

## Overall strategy

! We do not have access to  $\zeta$ , 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

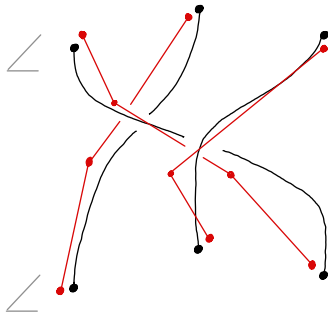
**Input :**  $\zeta$  ( $n$  disjoint tubular neighborhoods around  $\zeta_1, \dots, \zeta_n$ ).

**Output :** A decomposition in standard generators of the braid induced by  $\zeta_1, \dots, \zeta_n$ .

## Overall strategy

! We do not have access to  $\zeta$ , 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

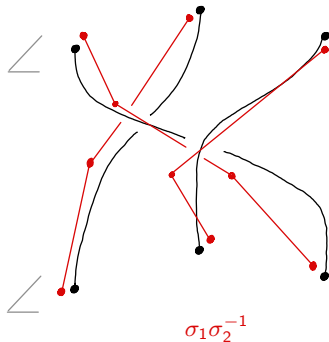
**Input :**  $\zeta$  ( $n$  disjoint tubular neighborhoods around  $\zeta_1, \dots, \zeta_n$ ).

**Output :** A decomposition in standard generators of the braid induced by  $\zeta_1, \dots, \zeta_n$ .

## Overall strategy

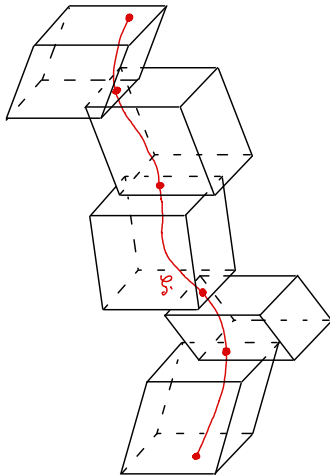
! We do not have access to  $\zeta$ , not even to  $\zeta(0)$ .

- 1) Find a path  $\tilde{\zeta}$  that has same associated braid.
- 2) Decompose  $\tilde{\zeta}$ .



## SIROCCO [Marco-Buzunariz and Rodríguez, 2016]

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



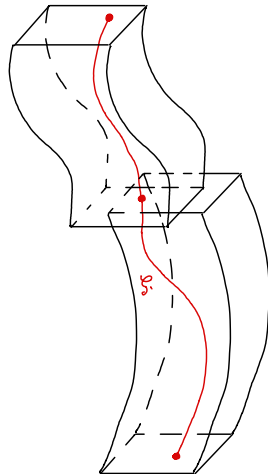


## SIROCCO [Marco-Buzunariz and Rodríguez, 2016]

- Tubular neighborhoods are piecewise **linear**.
- For each strand  $\zeta_i$ , 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]

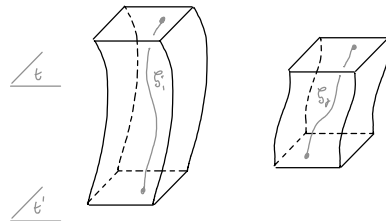
- Tubular neighborhoods are piecewise **cubic**.
- ⊕ Faster than SIROCCO.
- ! Finding a piecewise linear path in the tube requires additional work.



## 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  $\star = \leftarrow$ .



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

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

- $i \xrightarrow{\text{blue}}_c j \iff (i, j) \in R$
- $i \xrightarrow{\text{red}}_c j \iff (i, j) \in I$

# 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

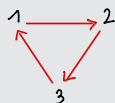
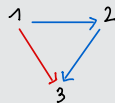
- for all  $(i, j) \in R$ ,  $\operatorname{Re}(x_i) < \operatorname{Re}(x_j)$ ,
- for all  $(i, j) \in I$ ,  $\operatorname{Im}(x_i) < \operatorname{Im}(x_j)$ ,

## Notation

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

## Examples

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



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

# Properties of cells

## Empty cells

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

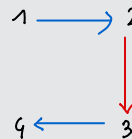
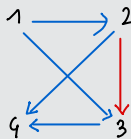
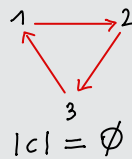
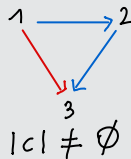
## Convex cells

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  $|c| \cap |c'|$ .

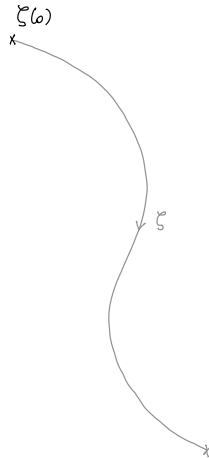
## Examples



## Step 1: compute a sequence of cells

### Path to cells

**Input:**  $\zeta$  (represented by tubular neighborhoods).



# Step 1: compute a sequence of cells

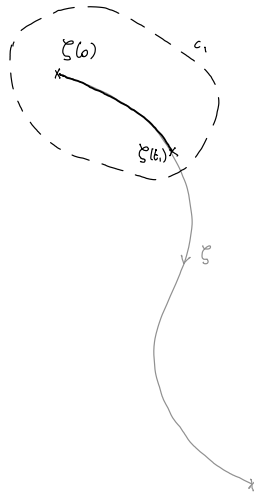
## Path to cells

**Input:**  $\zeta$  (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

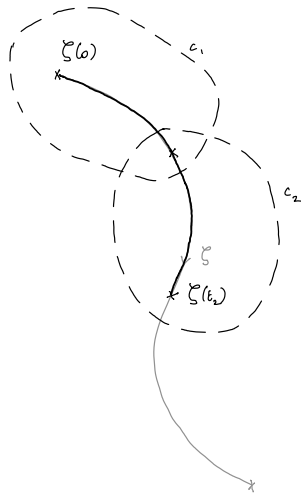
## Path to cells

**Input:**  $\zeta$  (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

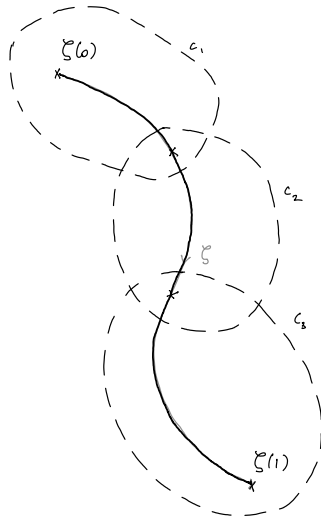
### Path to cells

**Input:**  $\zeta$  (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 $\zeta$

### Definition

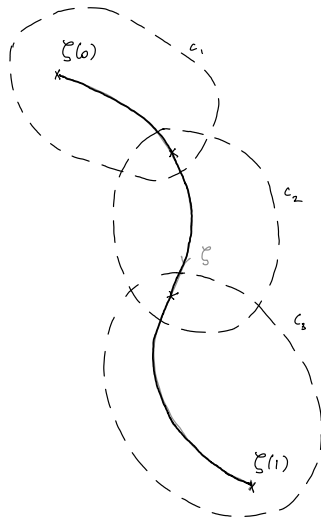
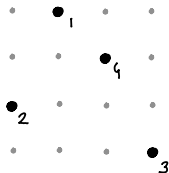
Let  $\pi, \varphi \in \mathfrak{S}_n$ . We define

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

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

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

$p_{\pi, e} :$



## Step 2: linearize $\zeta$

### Definition

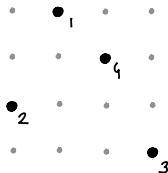
Let  $\pi, \varphi \in \mathfrak{S}_n$ . We define

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

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

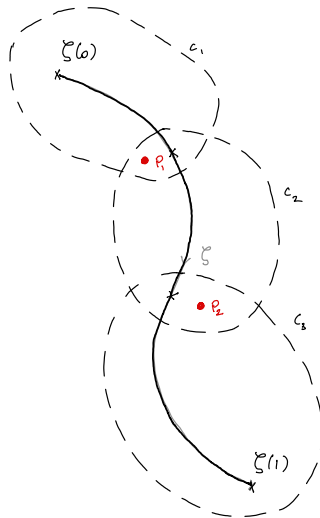
$$\varphi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 1 & 3 \end{pmatrix}$$

$p_{\pi, \varphi} :$



### Linearization of $\zeta$

For each  $c_i, c_{i+1}$ , we compute  $\pi, \varphi$  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 $\zeta$

### Definition

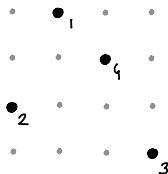
Let  $\pi, \varphi \in \mathfrak{S}_n$ . We define

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

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

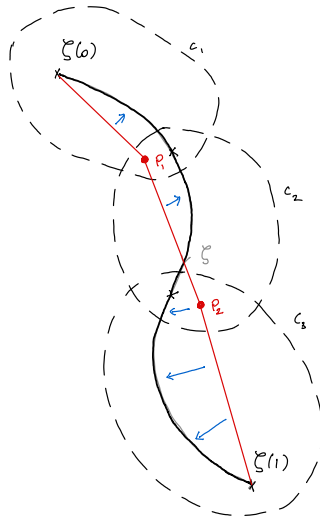
$$\varphi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 1 & 3 \end{pmatrix}$$

$p_{\pi, \varphi} :$



### Linearization of $\zeta$

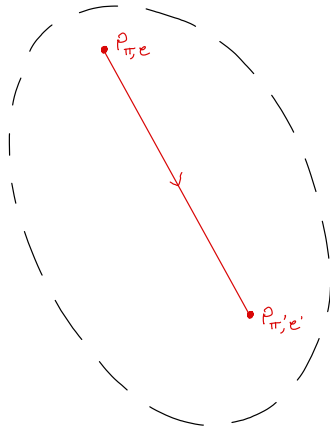
For each  $c_i, c_{i+1}$ , we compute  $\pi, \varphi$  such that  $p_i = p_{\pi, \varphi}$  lies in the intersection  $c_i \cap c_{i+1}$  (Hint: total order extending  $R$  and  $I$ ). The linear interpolation of the  $p_i$  is homotopic to  $\zeta$ . Why? cells are convex!



## Step 3: decomposition of the linearization

### Reduction

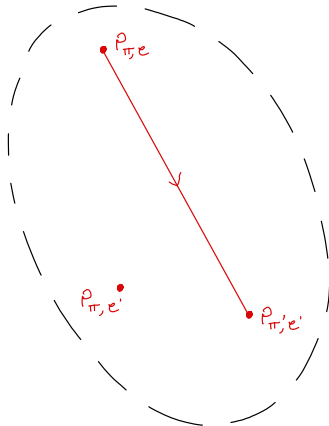
- Computing the braid associated to the whole linearization or to each piece and concatenating the results is equivalent.



## Step 3: decomposition of the linearization

### Reduction

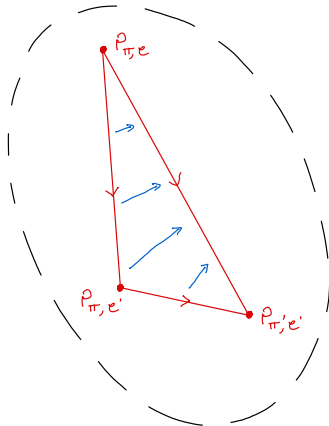
- Computing the braid associated to the whole linearization or to each piece and concatenating the results is equivalent.
- Assume  $p_{\pi,\varphi}$  and  $p_{\pi',\varphi'}$  both lie in a convex cell  $c = (R, I)$ . It means that  $\pi, \pi'$  extend  $R$  and  $\varphi, \varphi'$  extend  $I$ . **So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



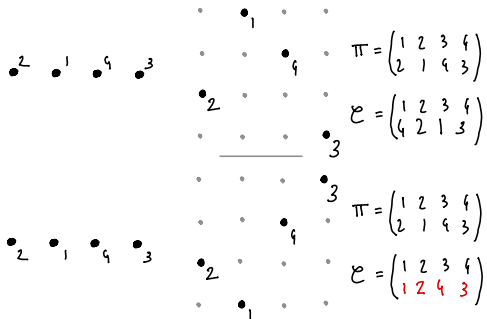
## Step 3: decomposition of the linearization

### 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  $\pi, \pi'$  extend  $R$  and  $\varphi, \varphi'$  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  $p_{\pi,\varphi'} \rightarrow p_{\pi',\varphi'}$ .

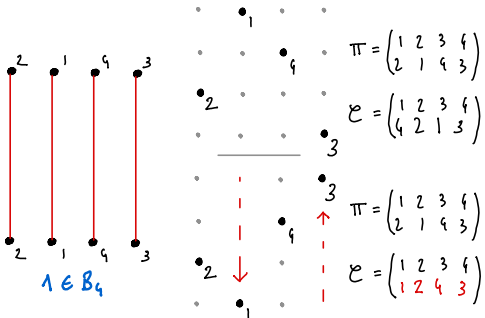


## Step 3: decomposition of the linearization



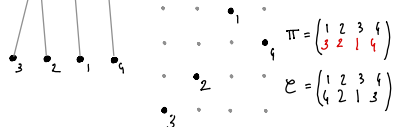
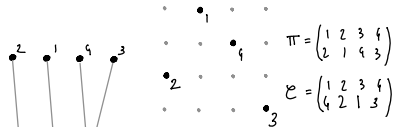
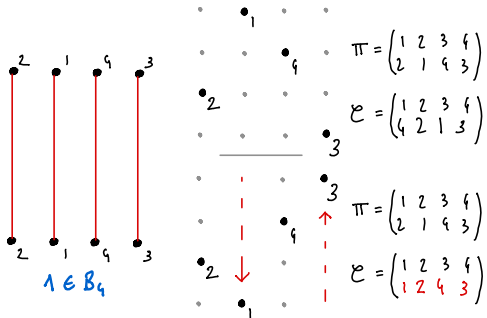


## Step 3: decomposition of the linearization



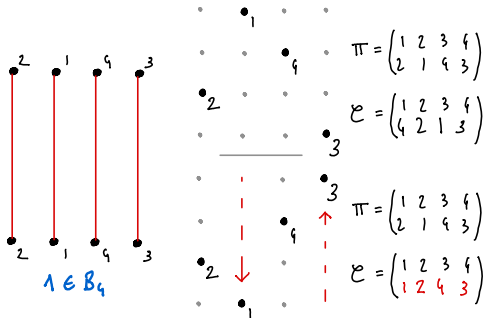
$p_{\pi, \varphi} \rightarrow p_{\pi, \varphi'}: \text{trivial braid.}$

# Step 3: decomposition of the linearization

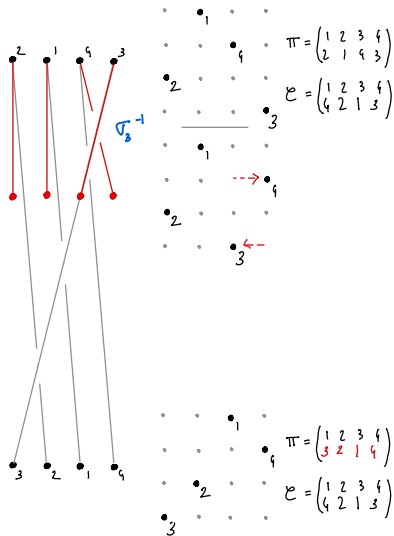


$p_{\pi, \varphi} \rightarrow p_{\pi, \varphi'}: \text{trivial braid.}$

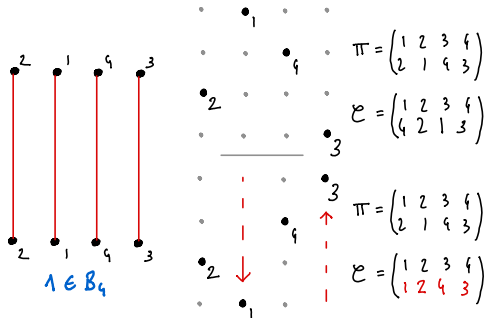
# Step 3: decomposition of the linearization



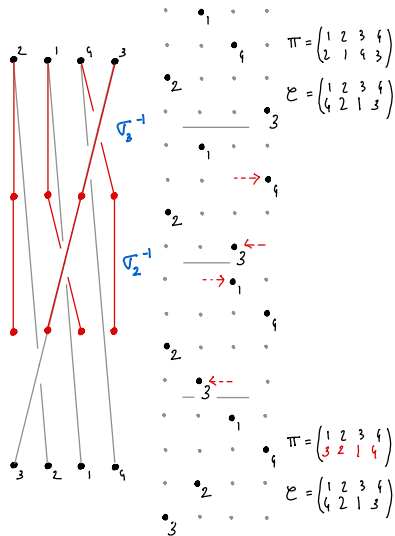
$p_{\pi, \varphi} \rightarrow p_{\pi, \varphi'}: \text{trivial braid.}$



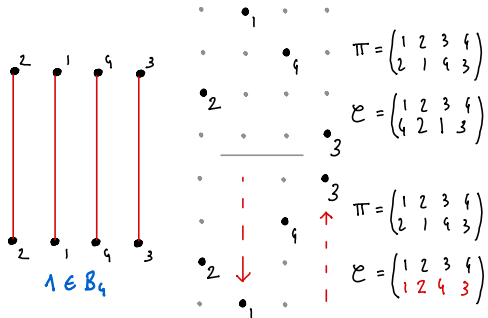
# Step 3: decomposition of the linearization



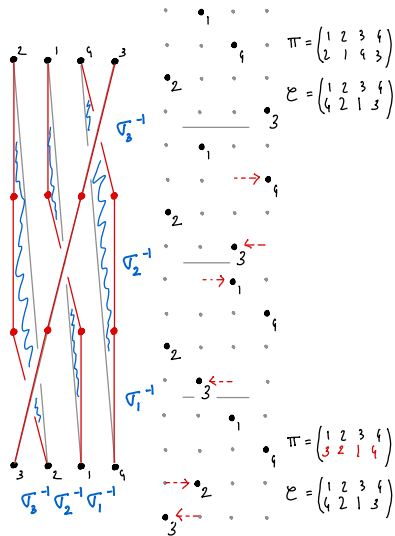
$p_{\pi, \varphi} \rightarrow p_{\pi, \varphi'}: \text{trivial braid.}$



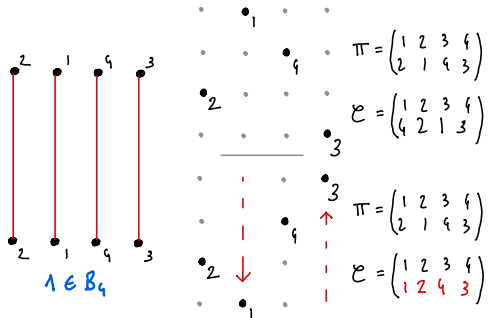
# Step 3: decomposition of the linearization



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

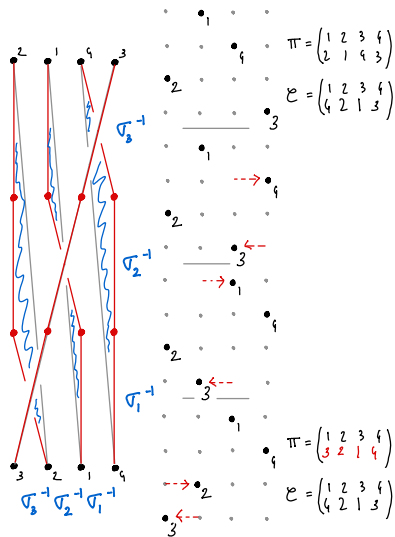


### Step 3: decomposition of the linearization



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



# Conclusion

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

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

```
Running `target/release/braid_group`
```

```
057011025029047051055061063083030504103103500-10100037073097098027049087097-10590150203-1096-1077092
-1093-1081092024072-1084018-1026094-1095-1010088087-1017-1024-1016-1025024082015-1063-1086-1013014-1
013-106-1081-1065087-1012-1073-1011-1088-1096-10930940405-1027-1010-1066062071074-106-1070604-101509
-1093023016-1092-108-1020-107-1075-106-105-104-1091022021089-1020073085090-1022091-103-1092-106093-1
082-1083-102-1082094-1095-1012-106700-101-100-1096-1035-1068076-1077-1097-1014015-1014-1098-10600940
92-1091078-1013-1014070-1069070059-1021083079-1080-1092071015023-1017-109-1018010019081-1018093092-1
017084083-1082-1083-1084-1016072079-1012-1076-1013085-1073086-1036074081087-1088-1015089-10140870130
12017-1018028090-1091-1078092-1093-1094-1095-1098097092-1096-1097-1024098-1094075029-1015076088087-1
033095094093092011077078068067-109109007908001009019-1079-108407-10807030078060805091031-10890408108
2047-103088020021-1083081084085076-1032033-10860201094087077-1075076088089075-1090091066065-1022-108
5-1084092093094095023096097024082-1083082019020-1034-1035036081-1074-1073072025063064063-1073-107402
6-1037-1038-1039062061060-1059-1060061-1023-1012-1013058057-1040041-1022014015-1042-1043056055-1054-
1053044-1072-1073045026025-1084-1074-1075086076-1077052051-1038085050049-1024-1023046-1047078-107909
800084048078082-1080030-1034052-1074076090-1023-1024036-1079078-104-1021046047070-1046-1028-1068-108
609206023054066-1088-1094-102032-1049050-1038-1010026-1056-1058-1072040-1044051-1085060-1052012096-1
045044-1024023-1017053054-1023024-1077076-1086075084-1042043042-1055-1056062-1074-1025-1026073072-10
57-1058038060061060-1059-1022049-1041-1014-1015-1014023-1060061-1040013012-1085-1062039038-1074073-1
037-1060-1063064-1036035034-1081-1065-1066072026-1025082083024025-1026082-1084-1073074-1032033032-10
33-1023022-1019-1020-1019097096031-1095094093092085030067-106808-1021-1020091090075-1076075087077-10
88076-1089088087086085081047-1098-1097-1029-1028096-1084094095-1094-1093-1083082081091092-1091-1090-
1078088089-1088-1079-1087-1082-1086-10102059-1060077018017-1085-1084-1083-1082-1078-1030408808089-1
018-1019-1018019069-1079084016-1078096077070081-10506080-1070800-101-1094-1079-1075-1076075078-1017-
101502-103-1011010-1091010077-1090-1076-101101604-105-1022-106-107-1074073014013-1014-1015-101408-107
5-1074-109-1010-1023-1072021026-1012013071022073-1067070069070025089-1072-1014015093-1092-1094093092
071-1016070-1017088-1095096-1011-1012-1015-1027-1013-1014-1015-1016-1094095-1018019086015-1026068067
011-1085069-1013014068-1087-1033017-1086-1018-1082-108304-105-106-1094-1081-108204050607091093-10100
92-1088021-1089-1023020021022021020087088-1023019-106-1020-1021-1022-105-104093-1020019-1020-1094065
013095096-1020302-10100-101-103-107-109-1013-1017-1019-1021-1023-1033-1035-1039-1043-1045-1053-1063-
1065-1067-1069-1071-1075-1079-1085-1087-1089-1091-1095-1097-1098
```

