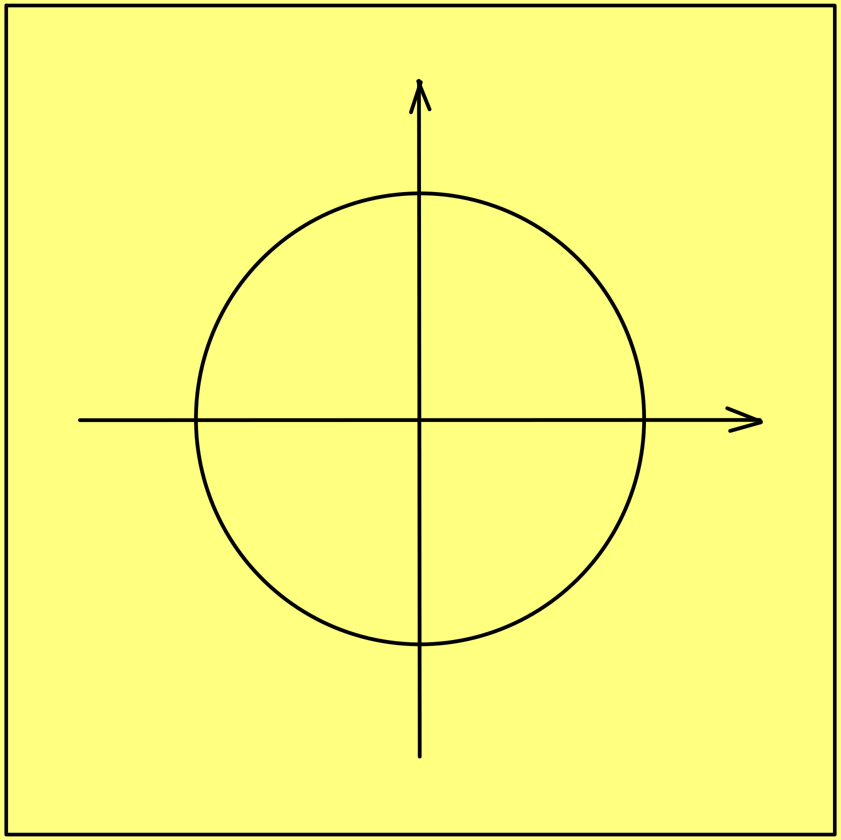
The Persistence of a Self-Map

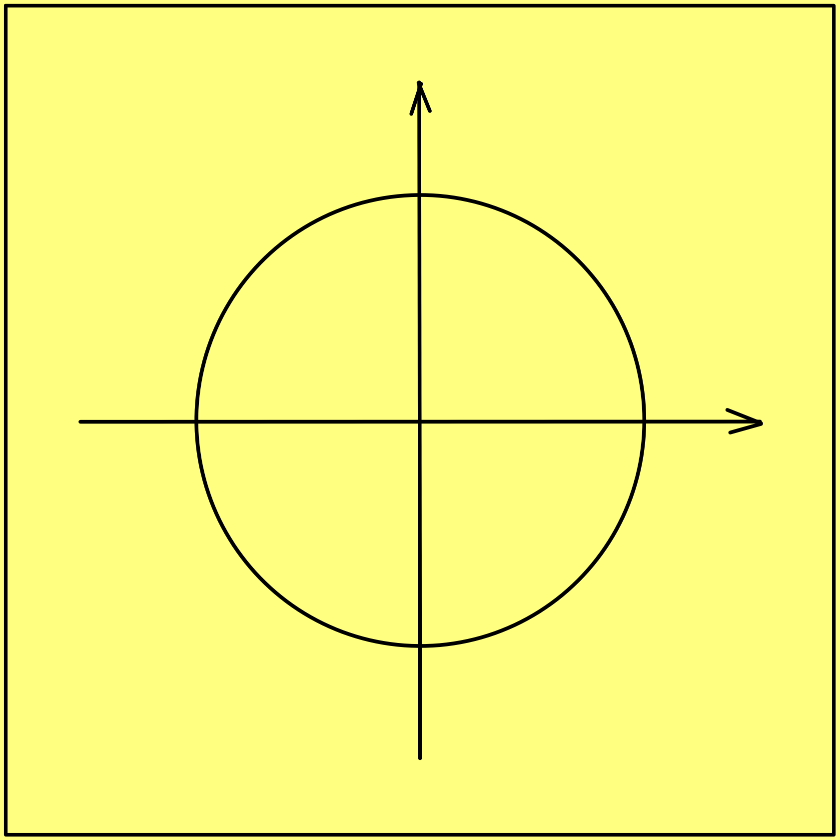
Herbert Edelsbrunner
IST Austria

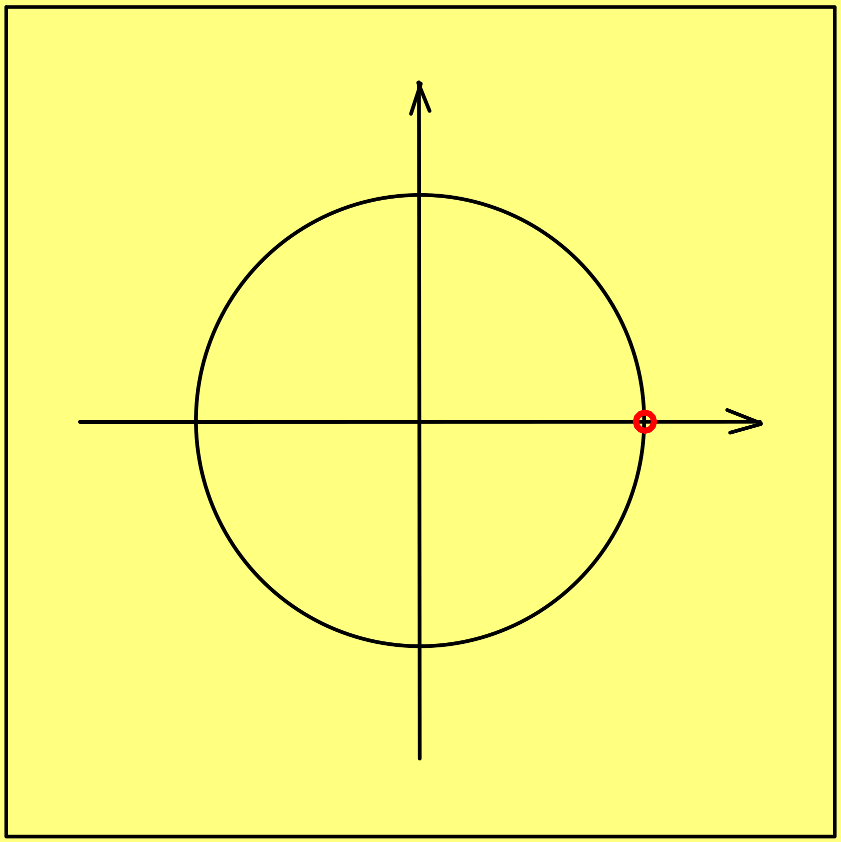
The Persistence of a Self-Map

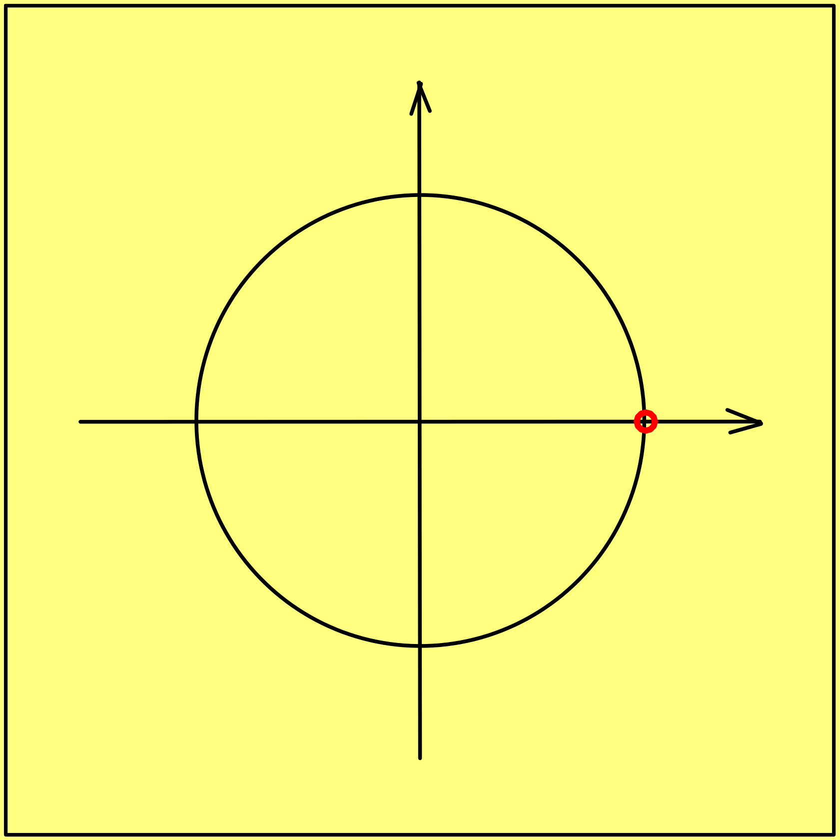
Herbert Edelsbrunner
IST Austria

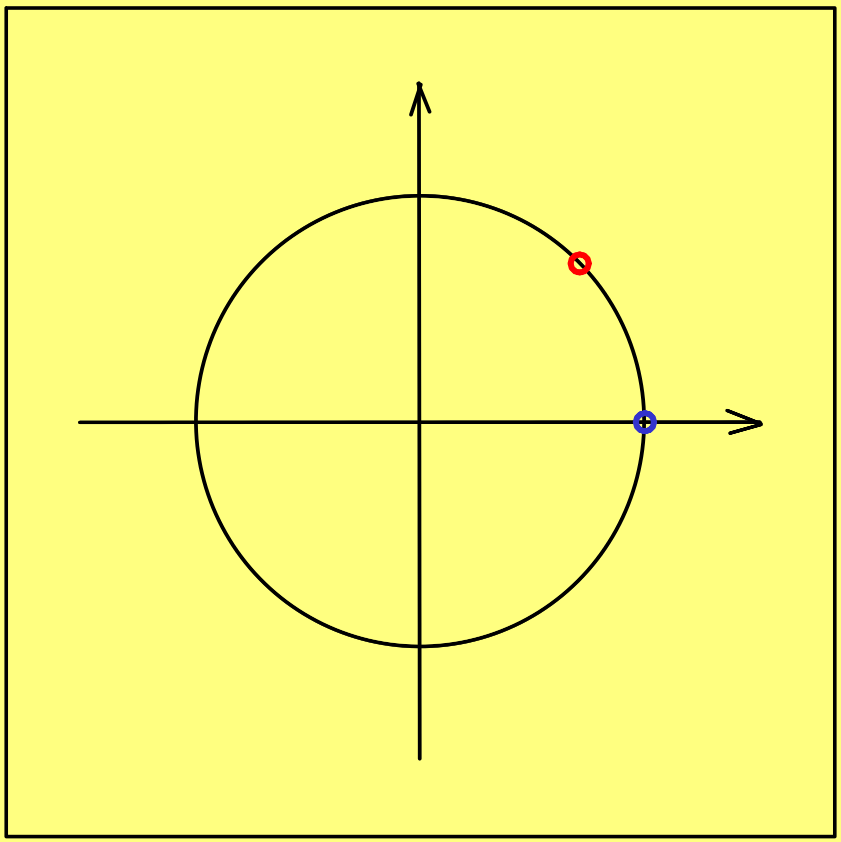
with Grzegorz Jabłoński and Marian Mrozek
Jagiellonian University, Kraków

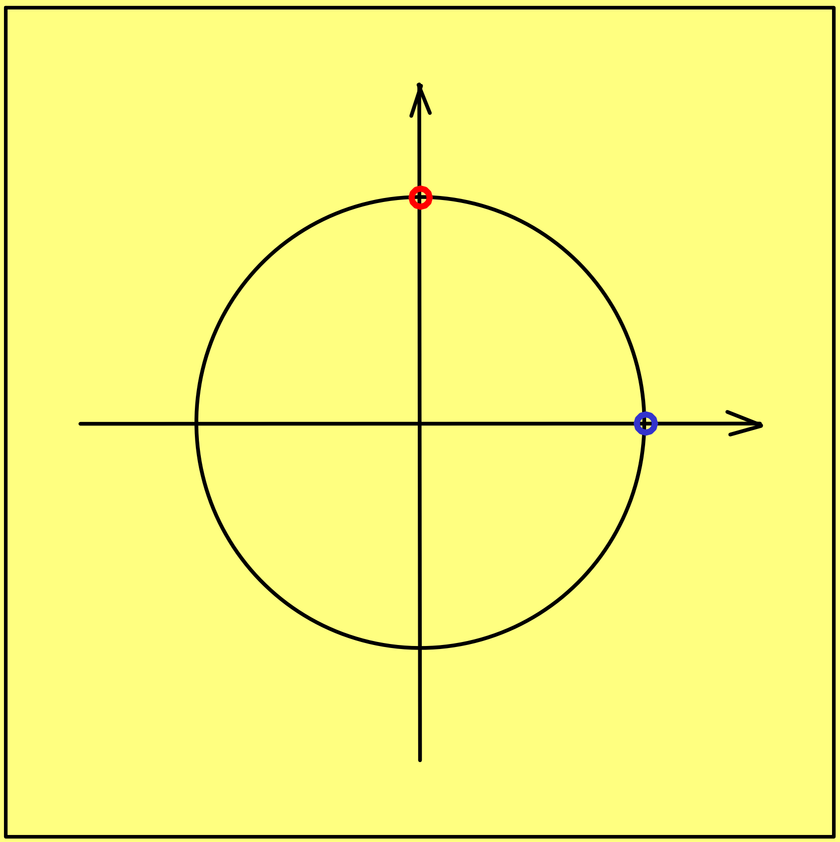


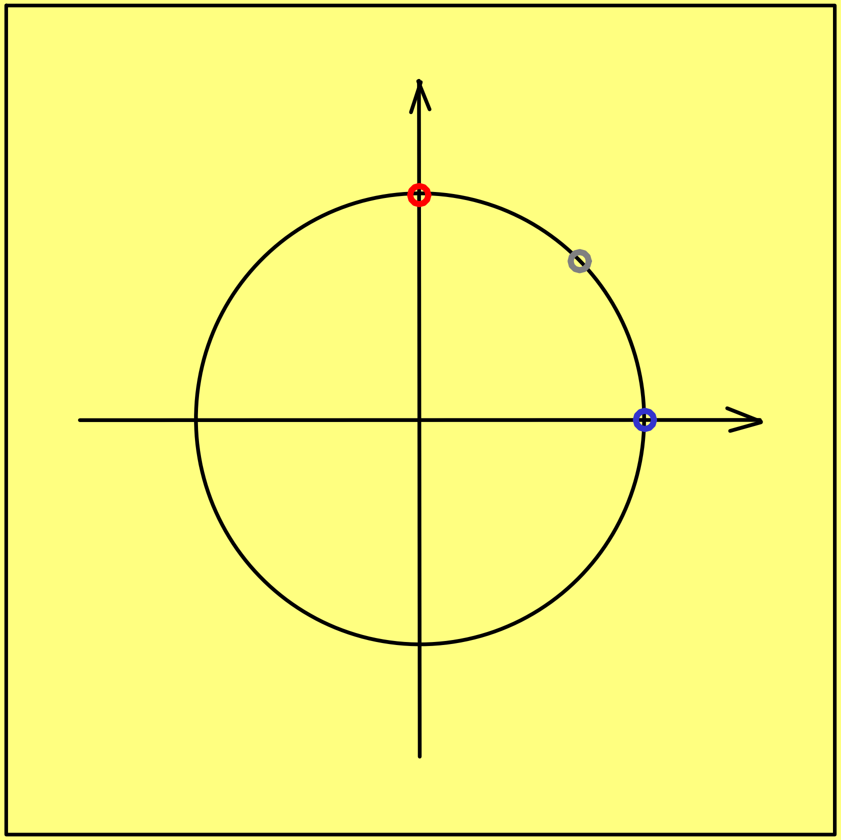
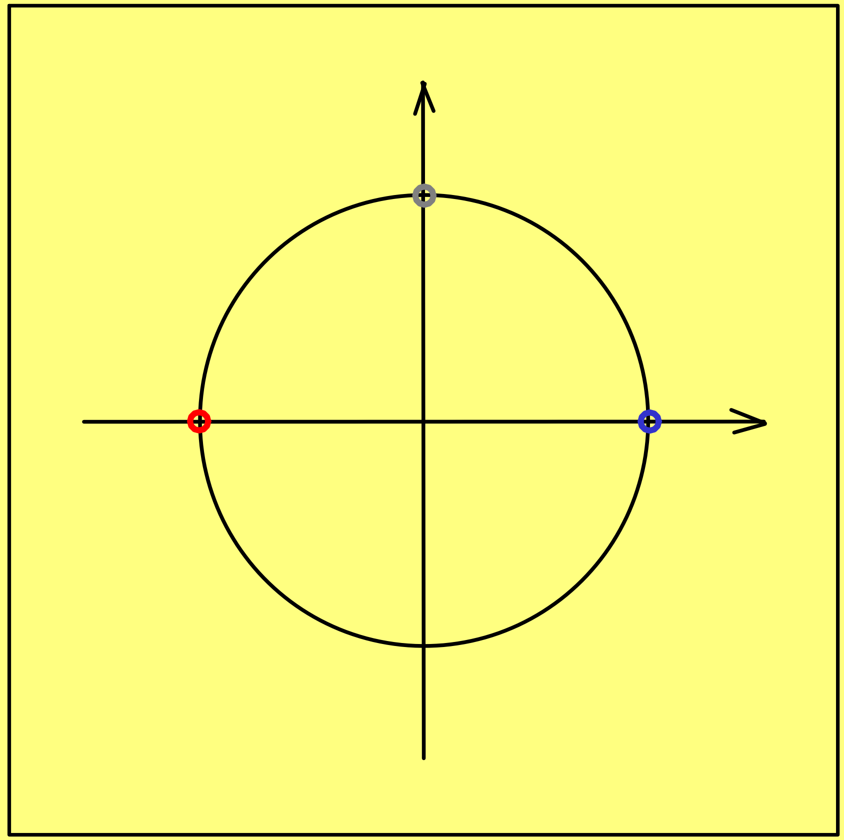


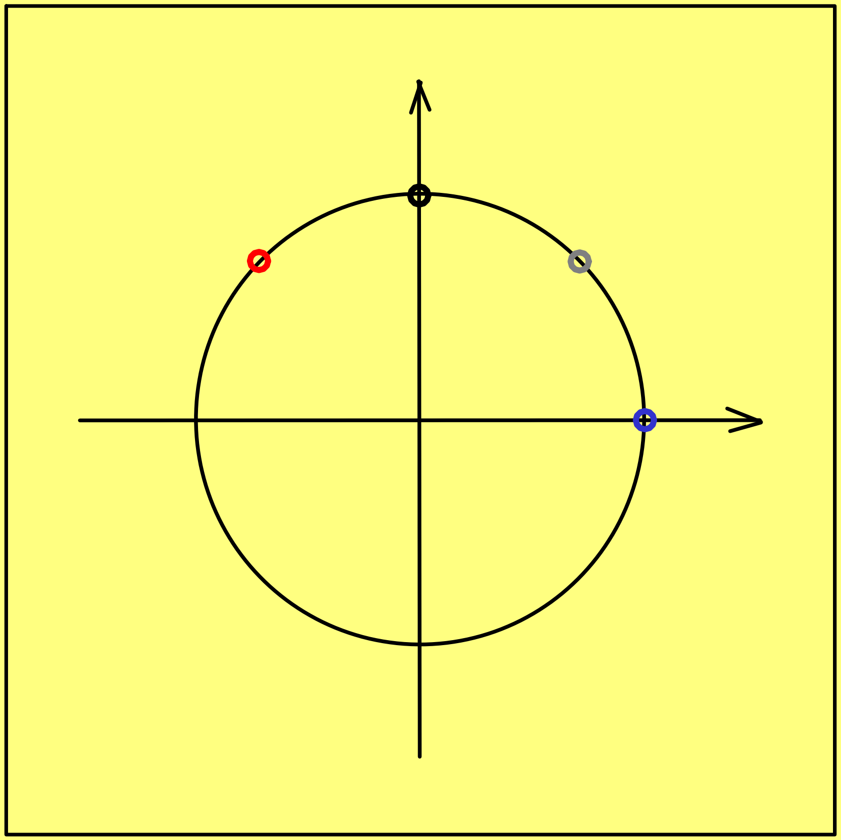


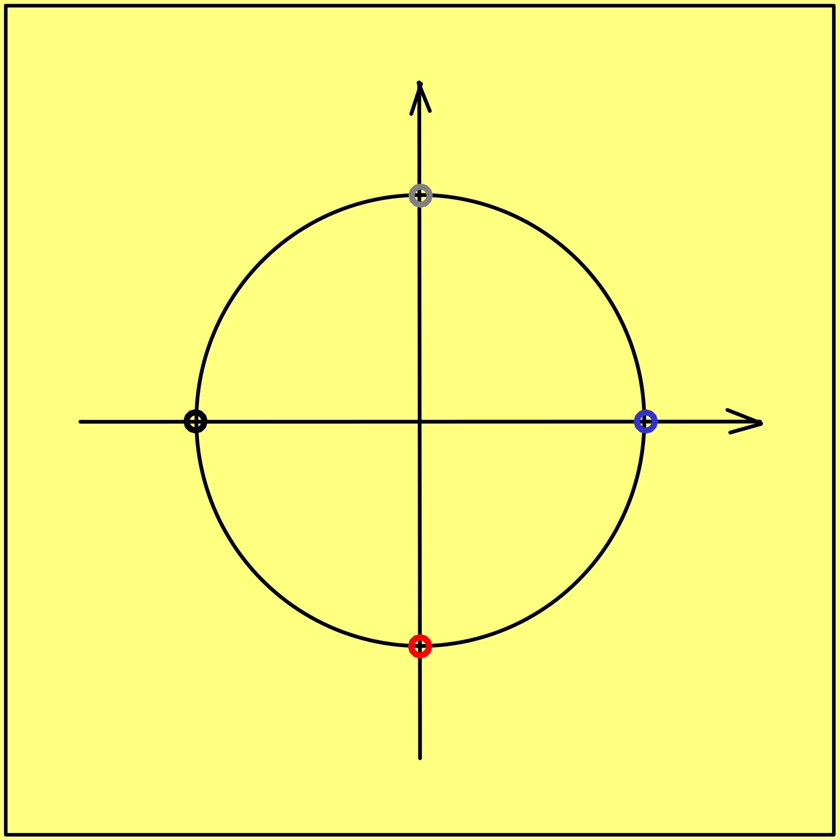


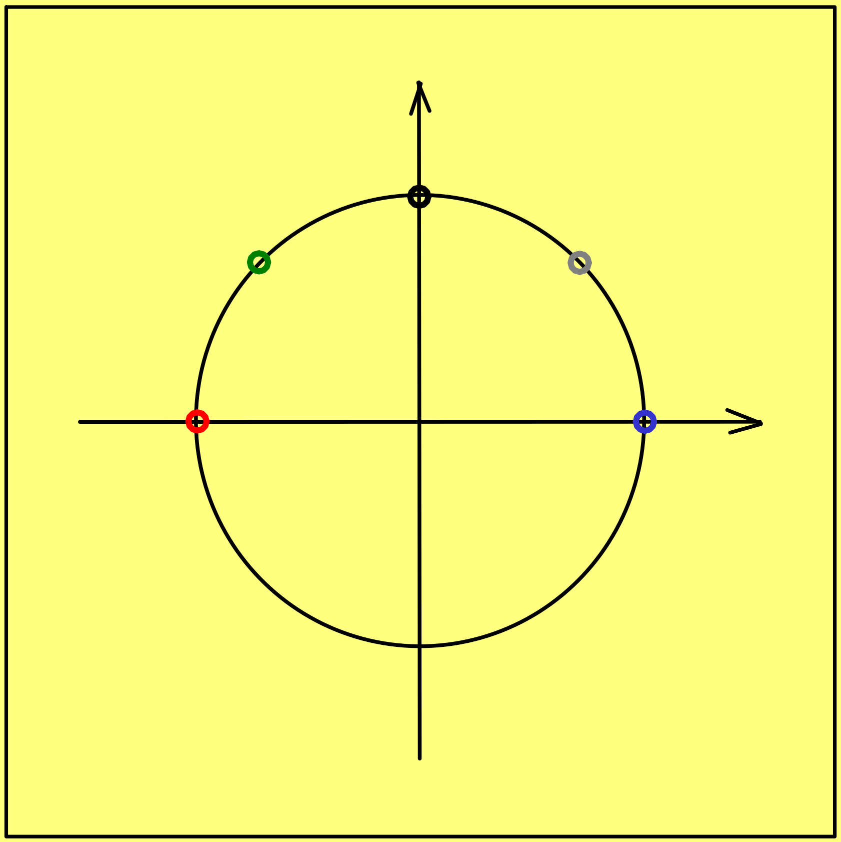


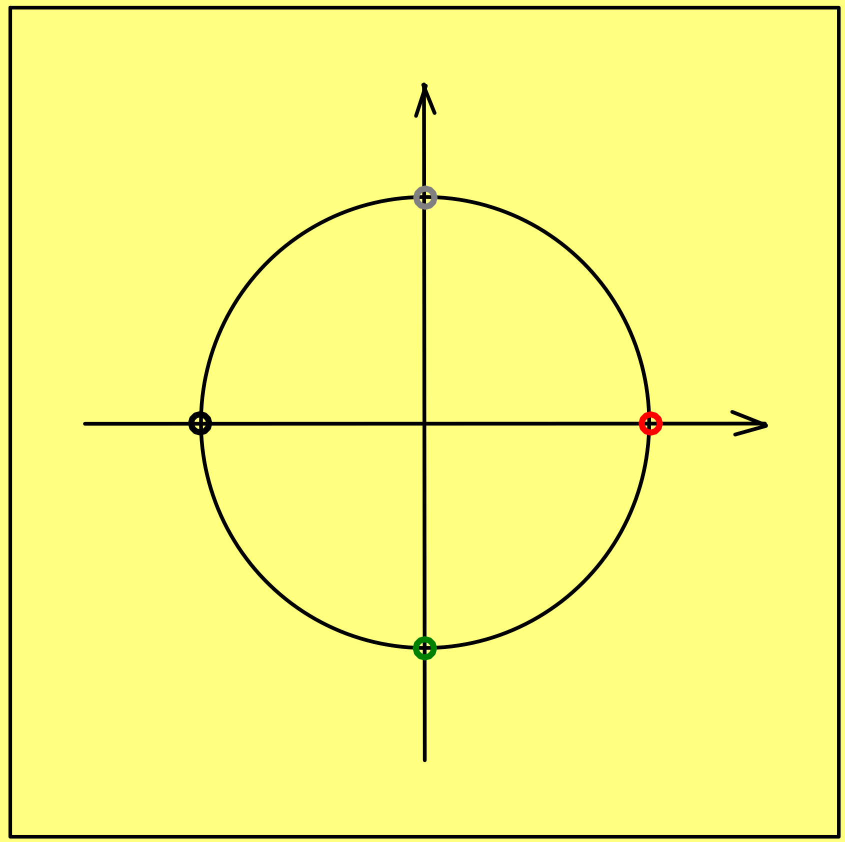


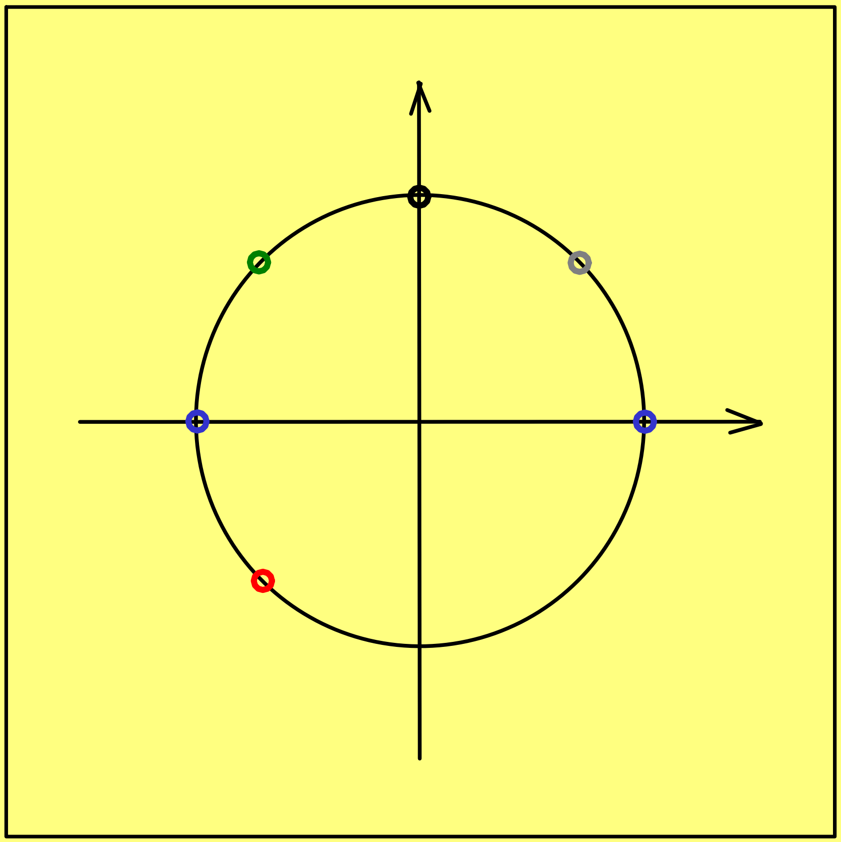



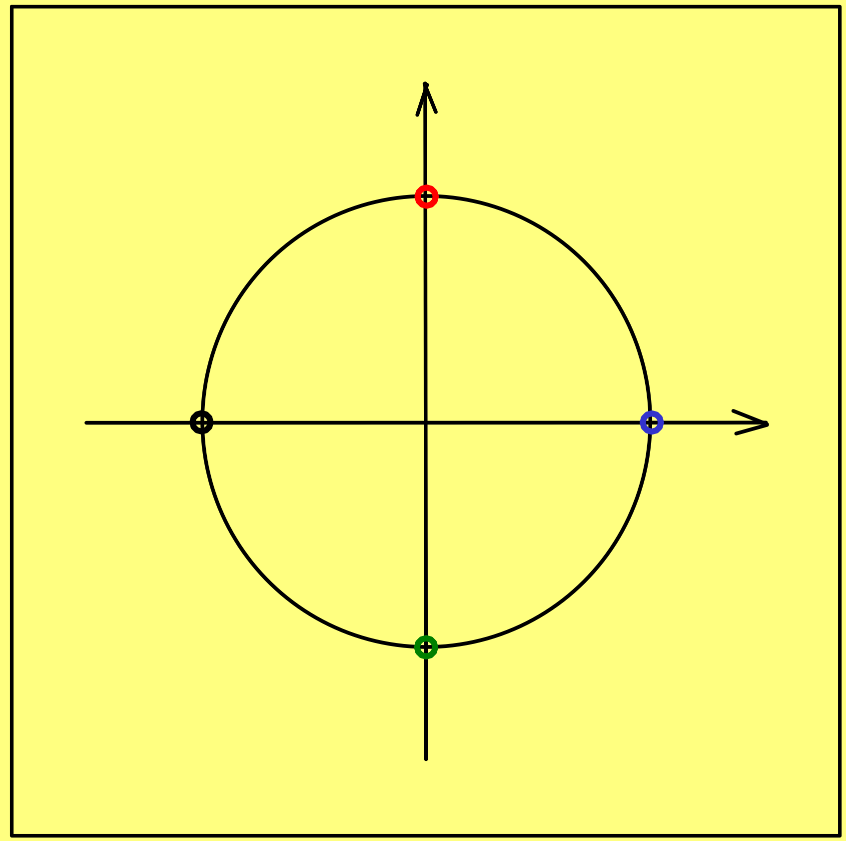


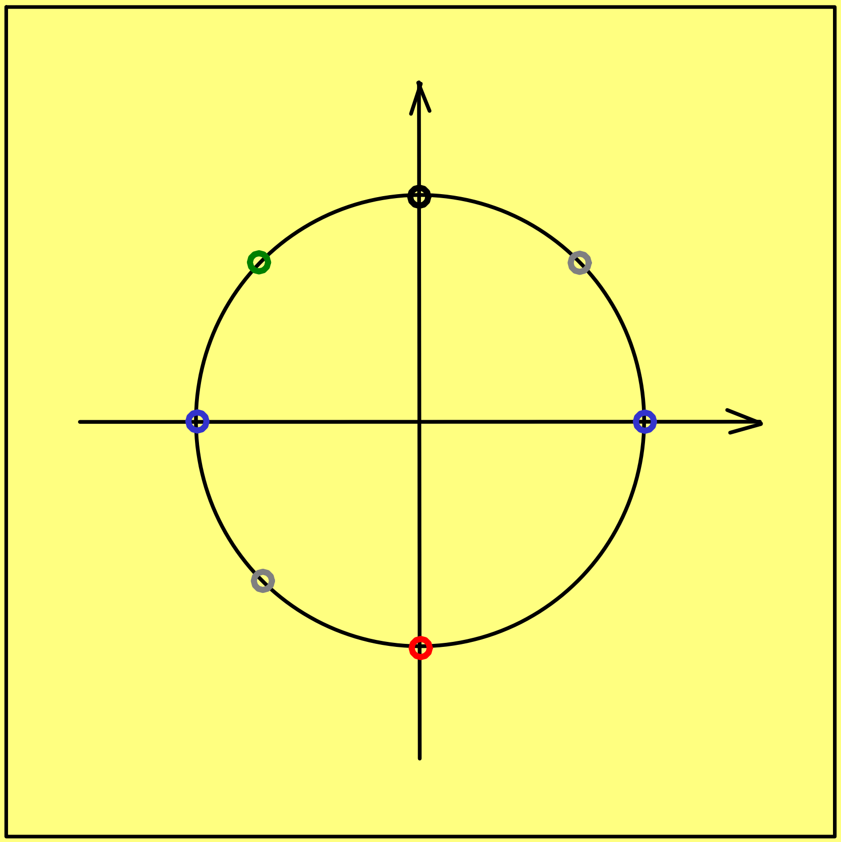


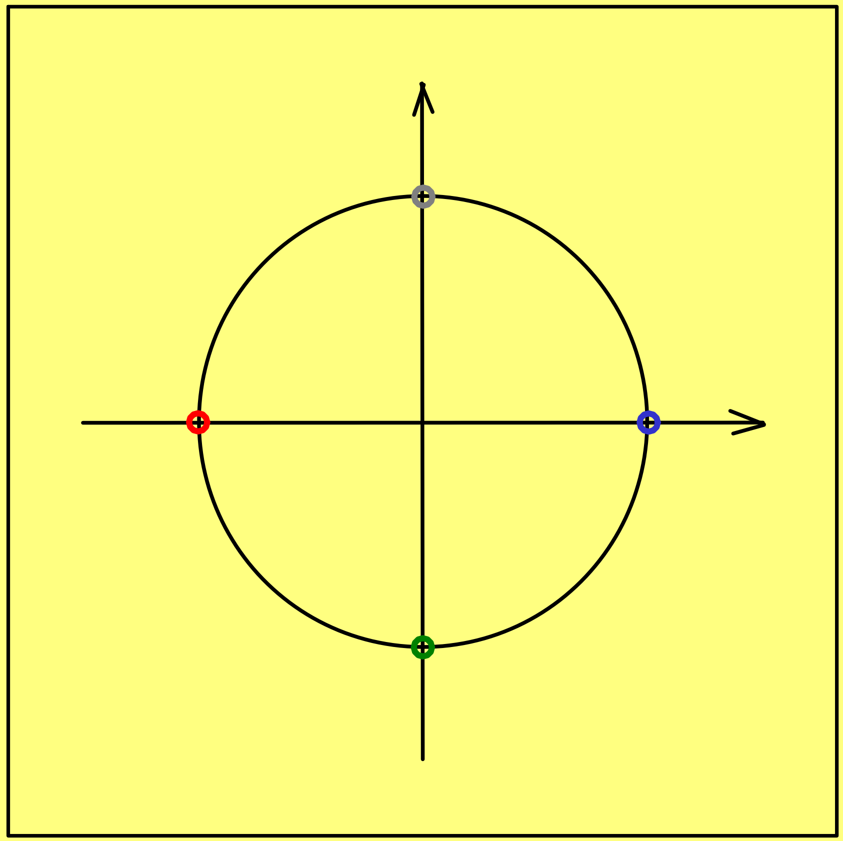


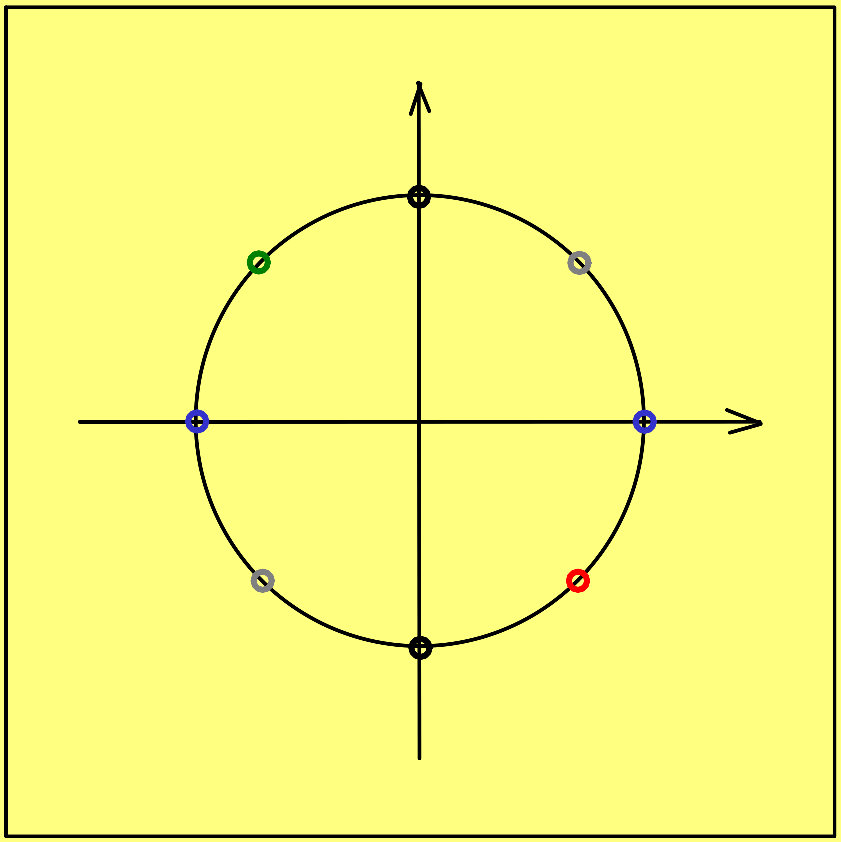


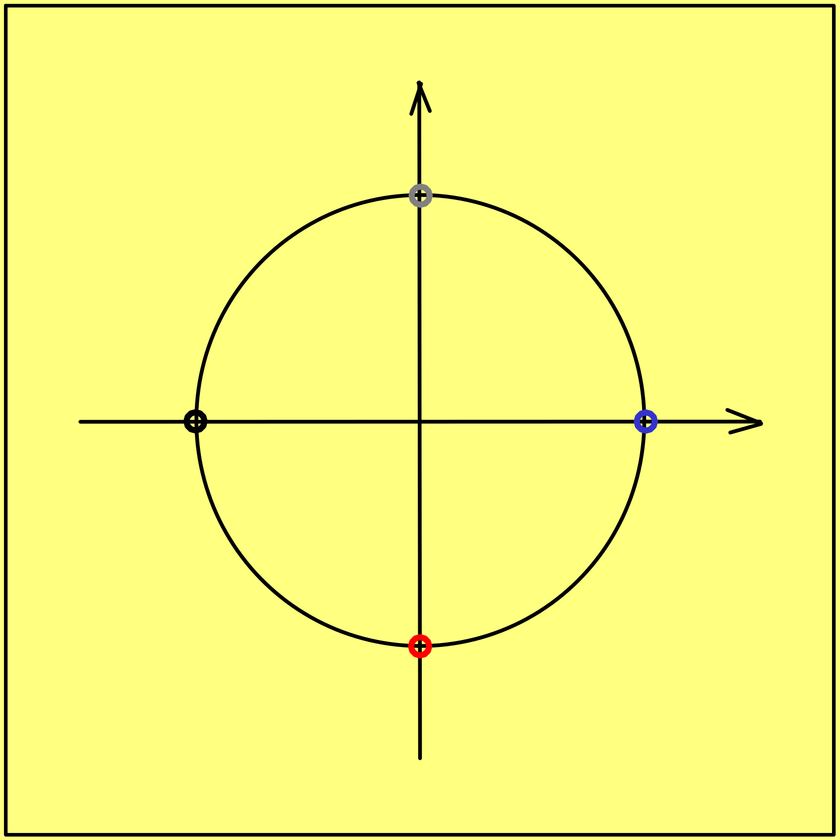


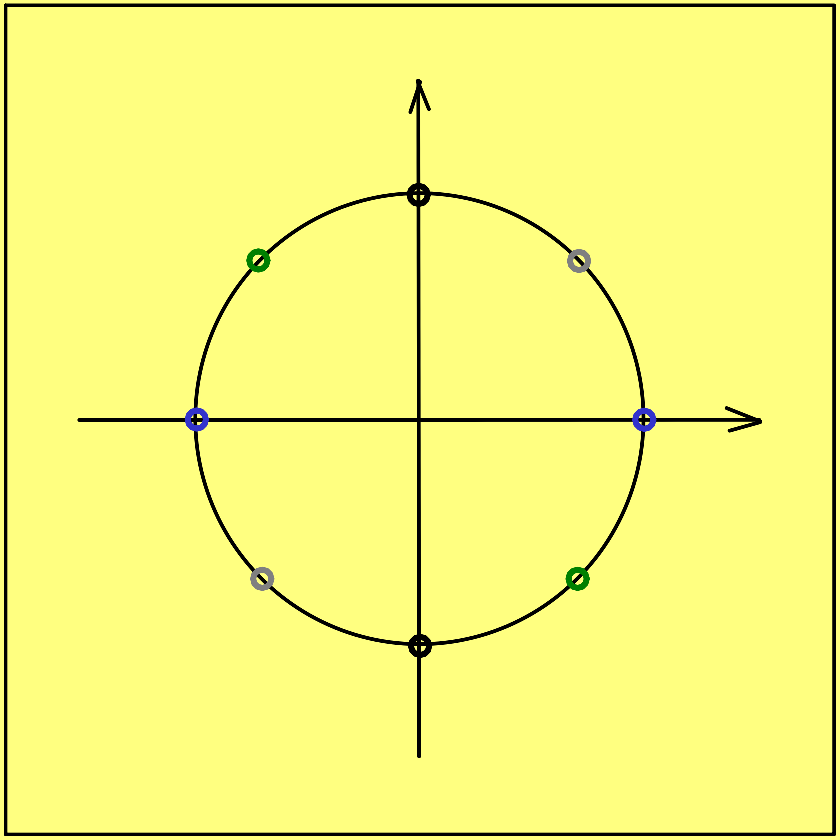


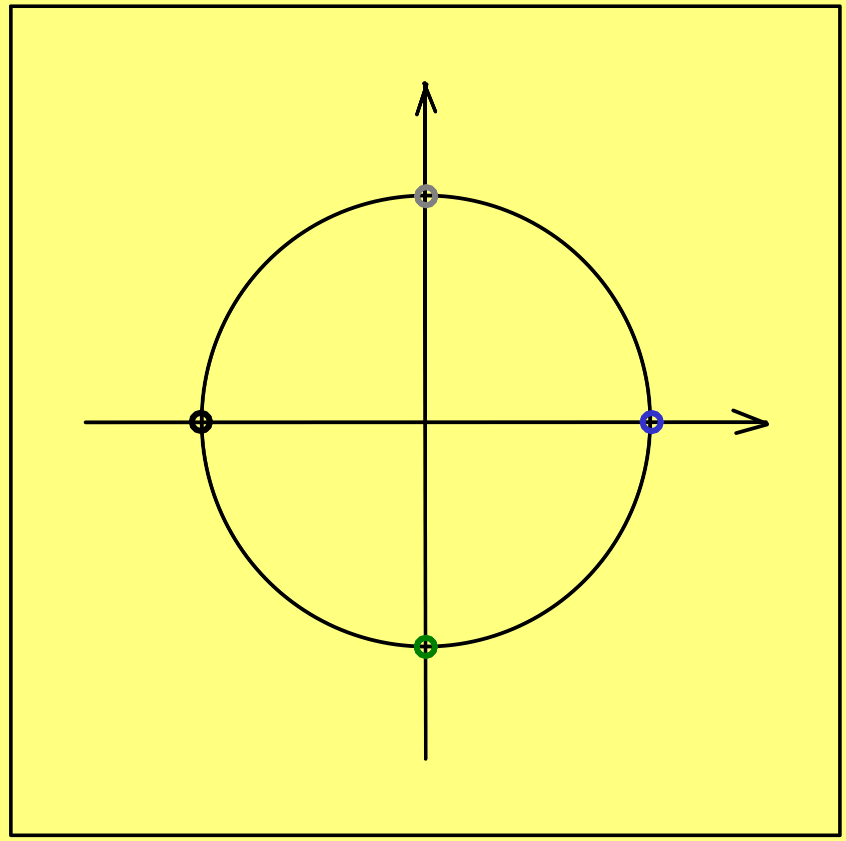












$f : S \to S$ defined by $f(x) = x^2$

The Problem

The Problem

$$\mathbb{X} \xrightarrow{f} \mathbb{X}$$

The Problem

$$\mathbb{X} \xrightarrow{f} \mathbb{X}$$

$$H \Downarrow$$

$$X \xrightarrow{\varphi} X$$

The Problem

$$\begin{tikzcd}
\mathbb{X} \arrow[r, "f"] \arrow[d, Rightarrow, "H"'] & \mathbb{X} \\
X \arrow[r, "\varphi"] & X
\end{tikzcd}$$

eigenvalue $t \in \mathbb{Z}_k$, eigenvector $\varphi(x) = tx$

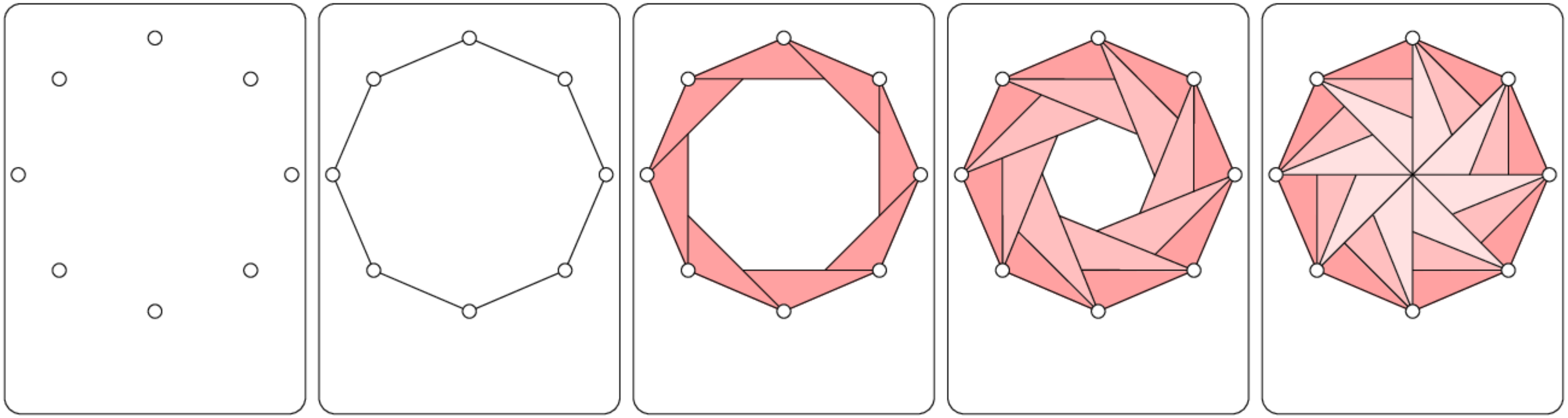
The Problem

$$\begin{tikzcd} \mathbb{X} \arrow[r, "f"] \arrow[d, Rightarrow, "H"'] & \mathbb{X} \\ X \arrow[r, "\varphi"] & X \end{tikzcd}$$

eigenvalue $t \in \mathbb{Z}_k$, eigenvector $\varphi(x) = tx$

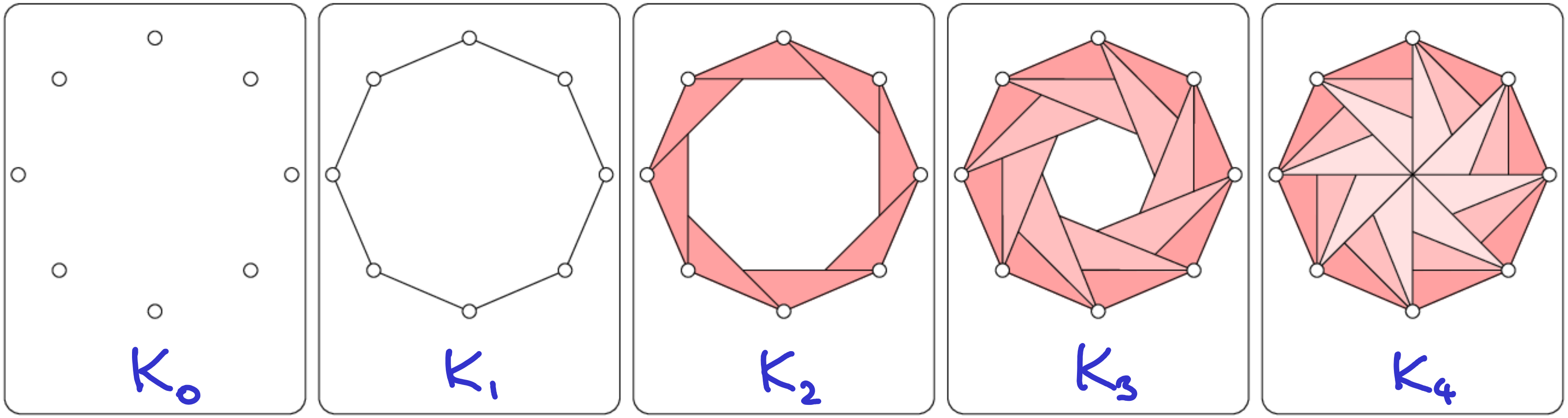
Question: Can we compute the eigen-values and -vectors from a finite sample of f ?

$f : S \to S$ defined by $f(x) = x^2$



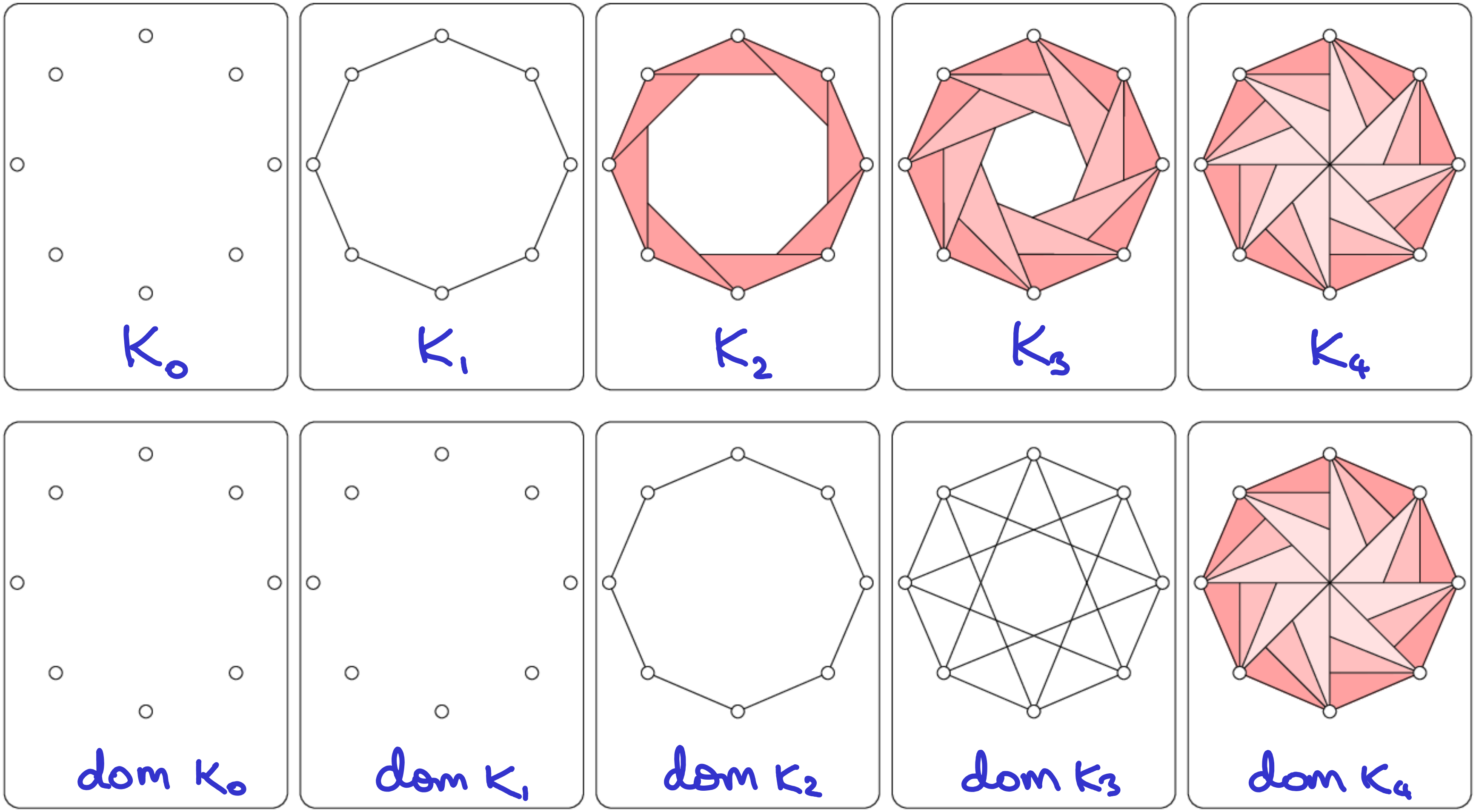
Vietoris-Rips complex K_i

$f: S \to S$ defined by $f(x) = x^2$



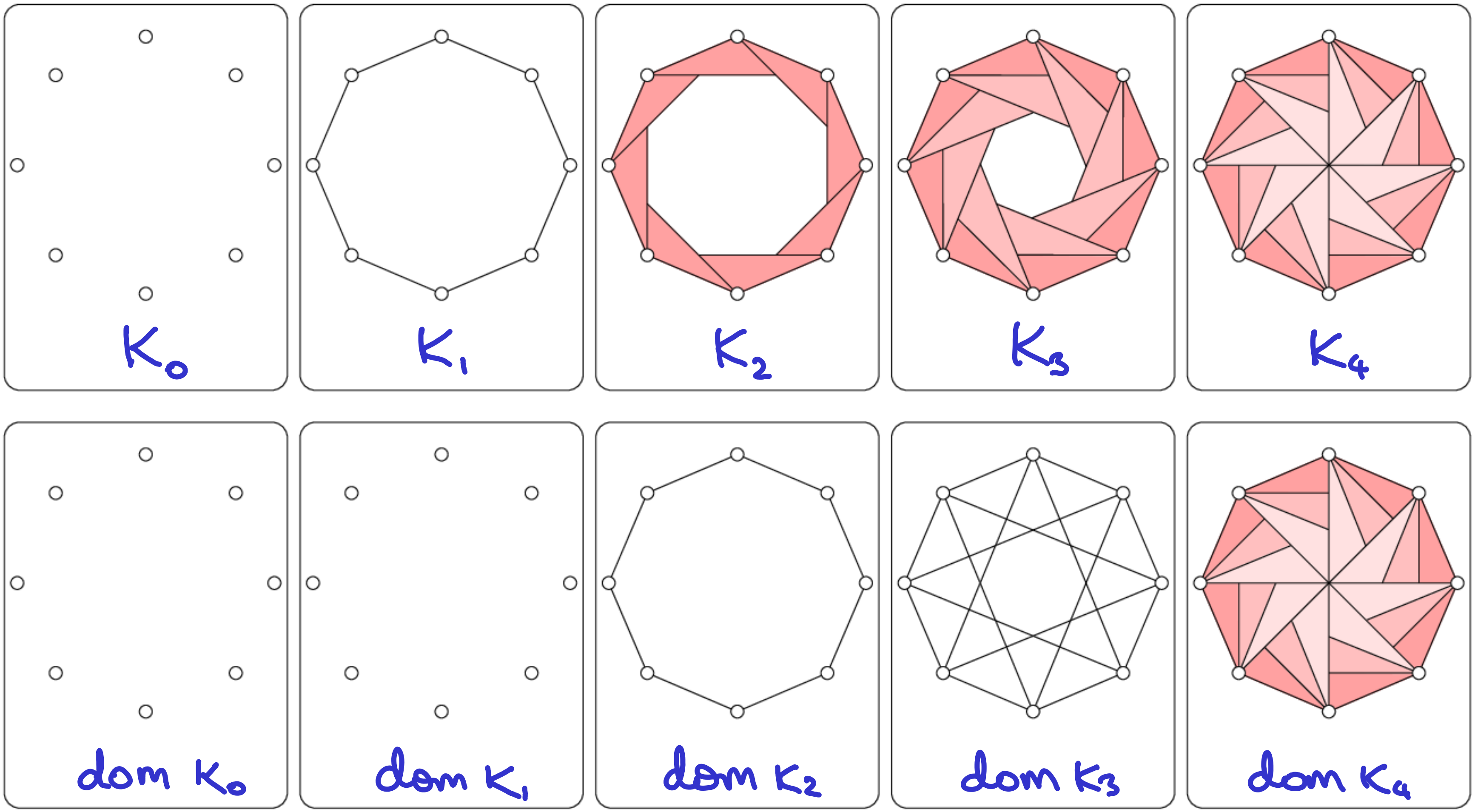
Vietoris-Rips complex K_i

$f : S \to S$ defined by $f(x) = x^2$



Vietoris-Rips complex K_i

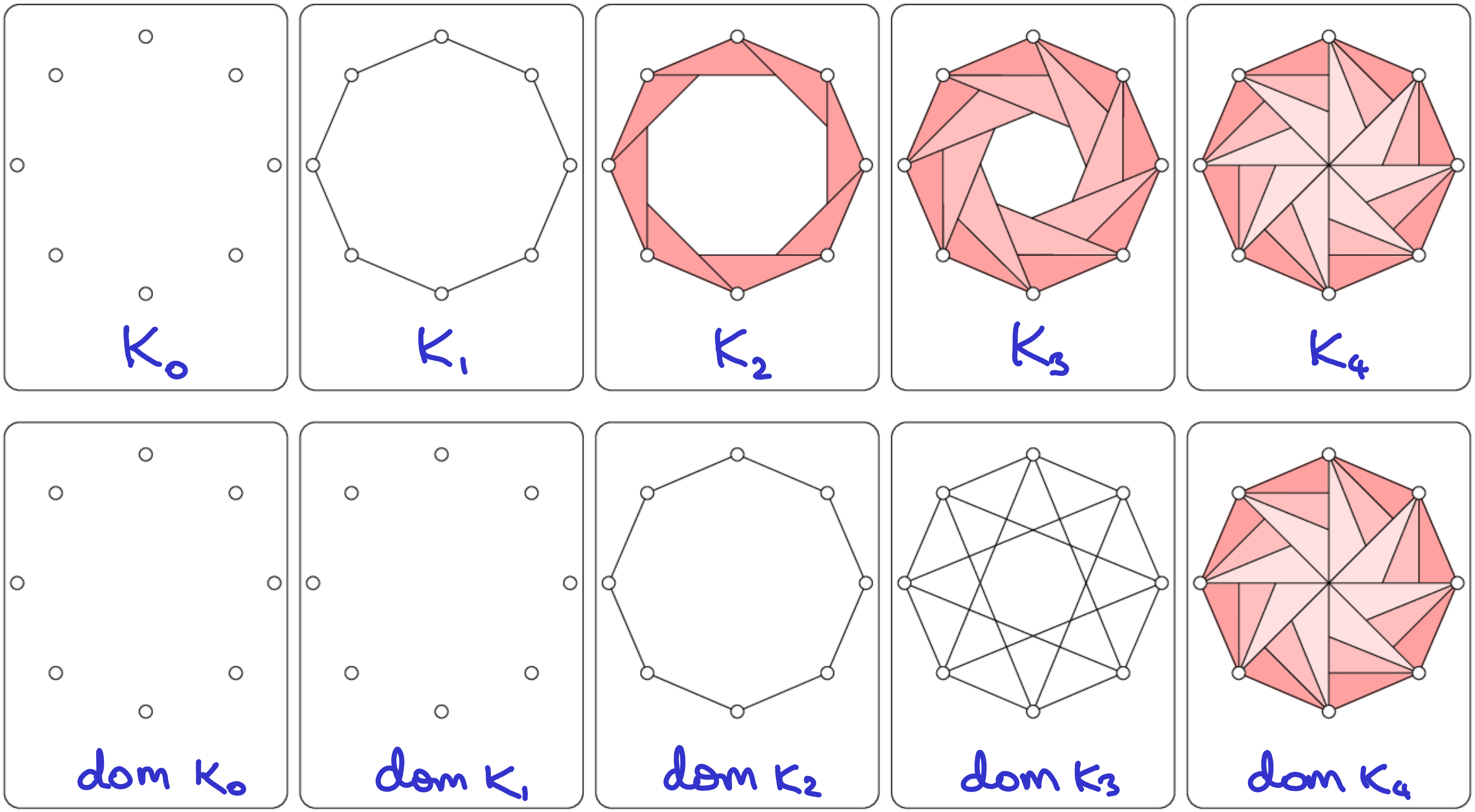
$\kappa_i : K_i \to K_i$



Vietoris-Rips complex K_i

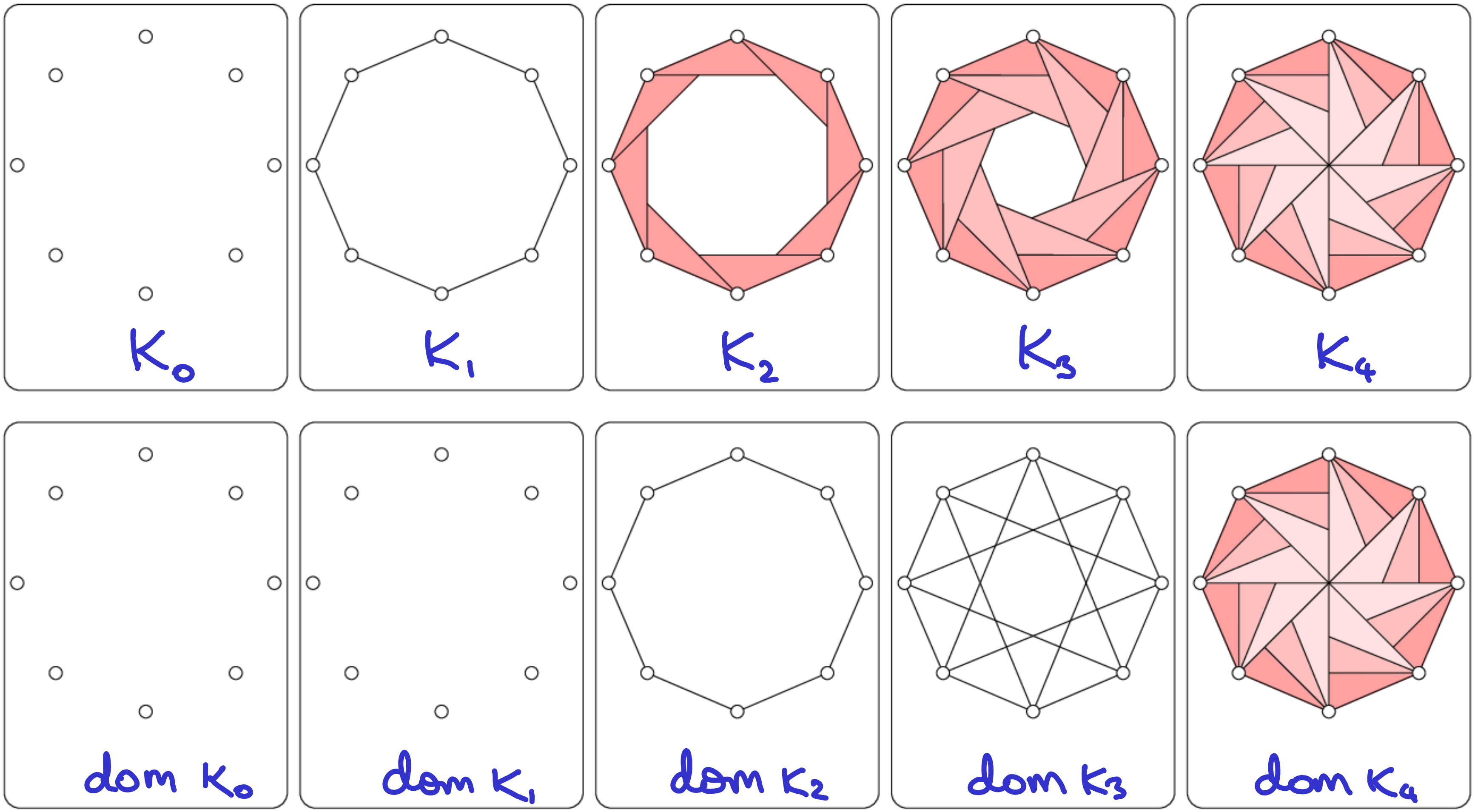
$\kappa_i : K_i \to K_i$

$\mathrm{dom}\,\kappa_i$ is preimage of κ_i



compare restriction $\operatorname{dom} K_j \to K_j$

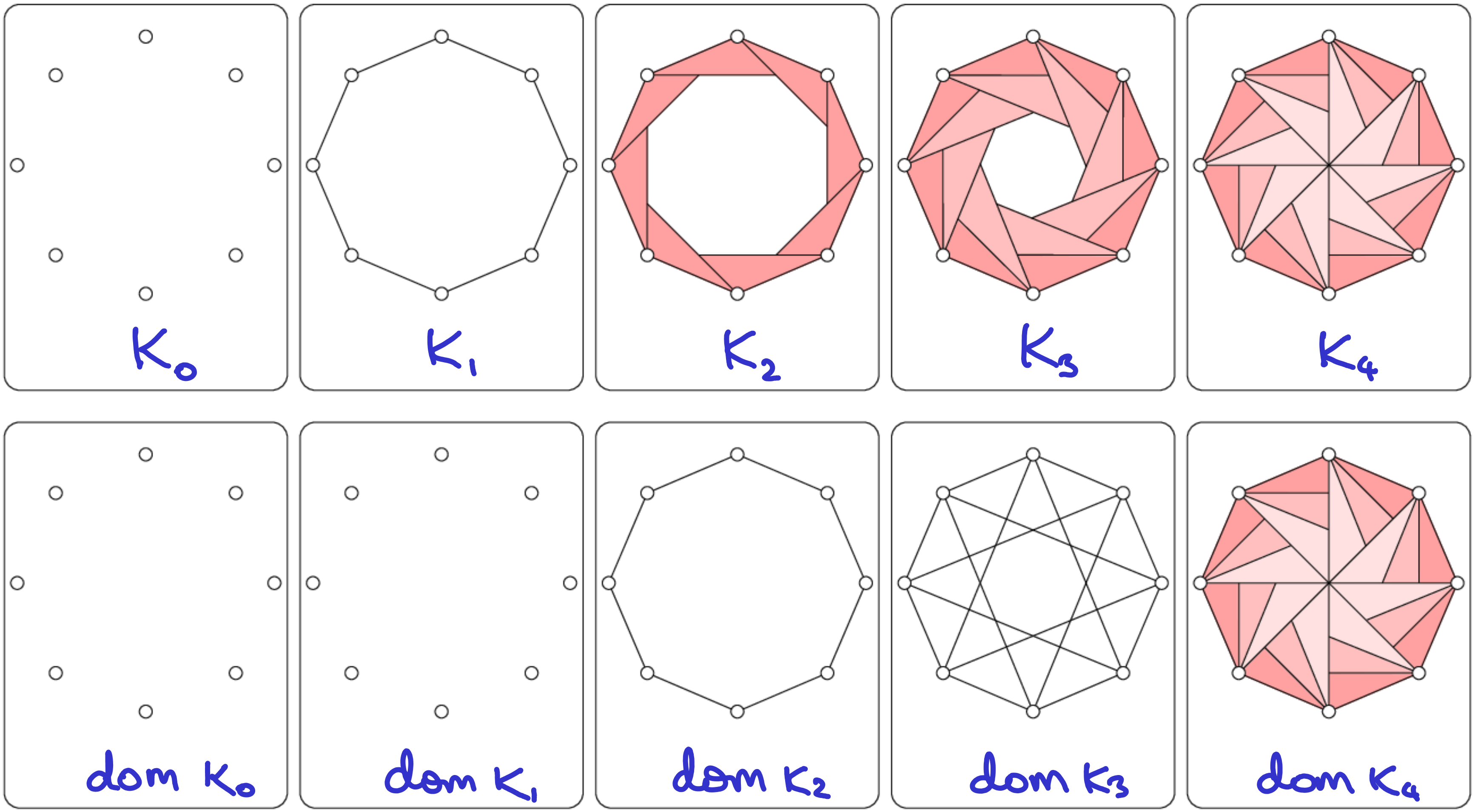
with inclusion $\operatorname{dom} K_j \to K_j$



compare restriction $\operatorname{dom} \kappa_j \to K_j$

with inclusion $\operatorname{dom} \kappa_j \hookrightarrow K_j$

For 1-dim. homology, we get eigenvalue $t = 2$ for $j = 2, 3$.



I. **Categories**

II. Linear Maps

III. Algorithm

IV. Analysis

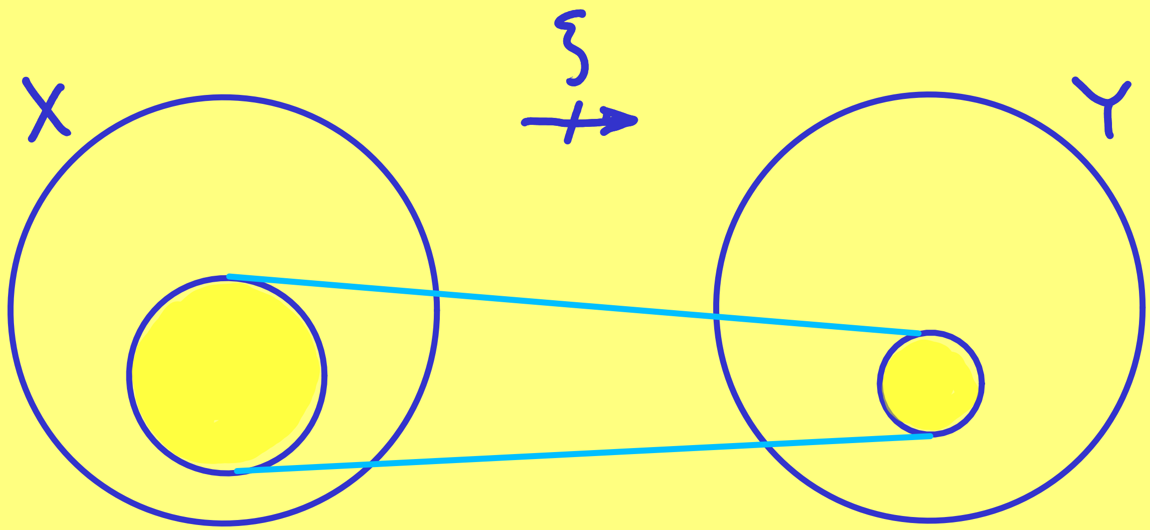
I.1 Partial Functions

I.1 Partial Functions

$f : X \rightharpoonup Y$ is **partial function** if every $x \in X$ maps to no or one element of Y.

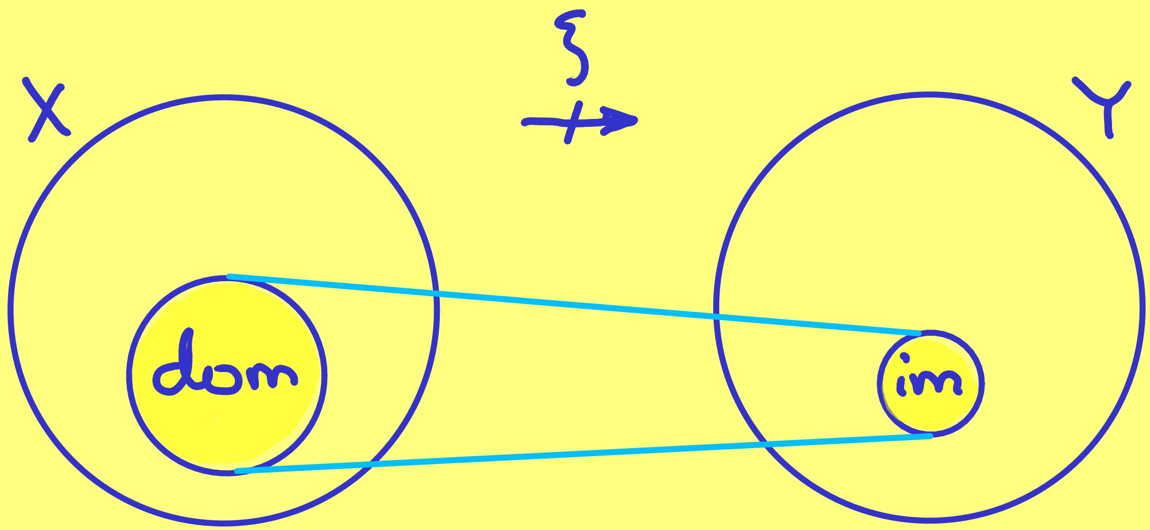
I.1 Partial Functions

$\xi : X \not\to Y$ is partial function if every $x \in X$ maps to no or one element of Y.



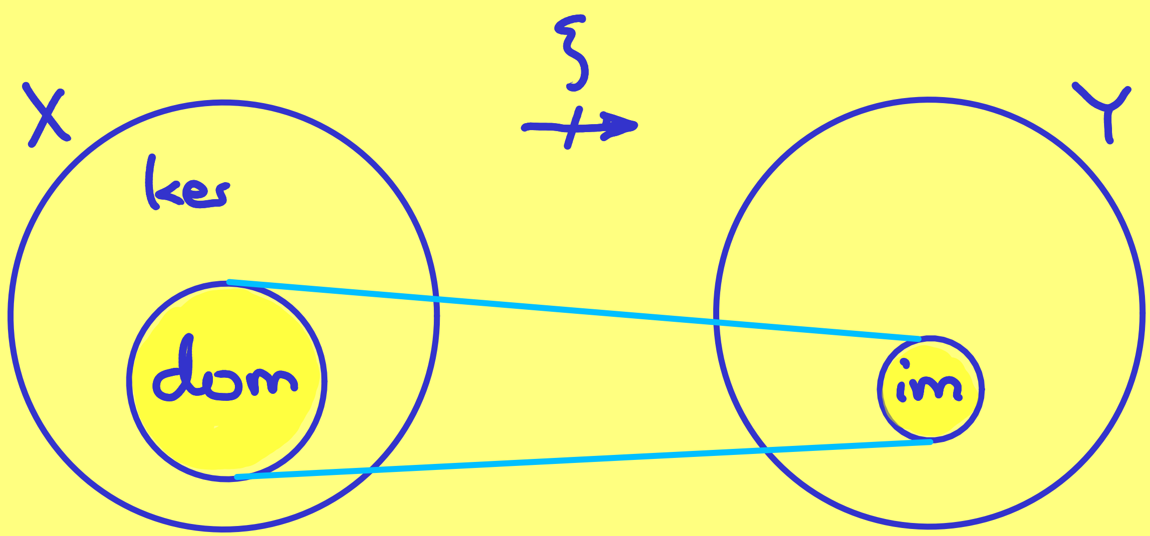
I.1 Partial Functions

$\xi : X \nrightarrow Y$ is partial function if every $x \in X$ maps to no or one element of Y.



I.1 Partial Functions

$\xi : X \nrightarrow Y$ is partial function if every $x \in X$ maps to no or one element of Y.



I.2 CATEGORY Part

I.2 CATEGORY Part

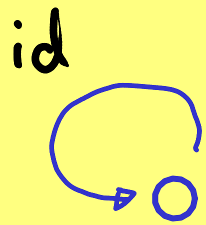
. . . objects are finite sets,

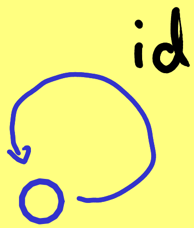
arrows are partial functions.

I.2 CATEGORY Part

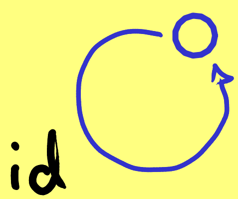
... objects are finite sets,

arrows are partial functions.





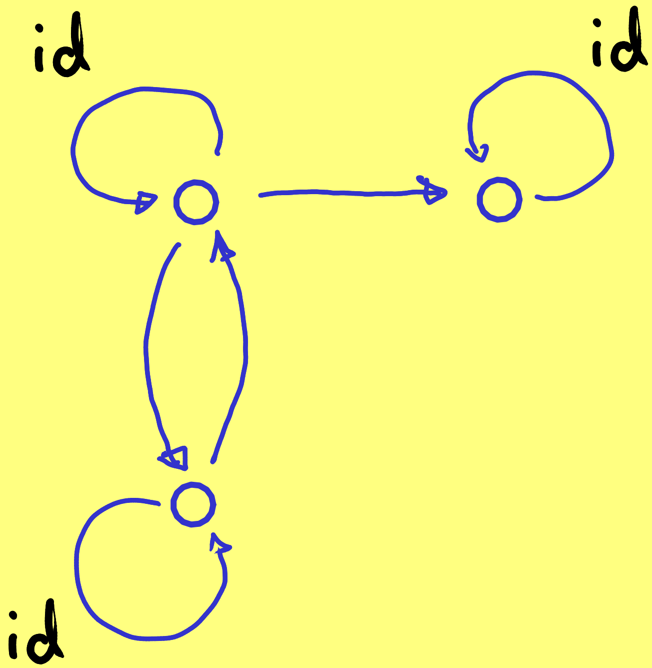
- identity arrows exist



I.2 CATEGORY Part

... objects are finite sets,

arrows are partial functions.

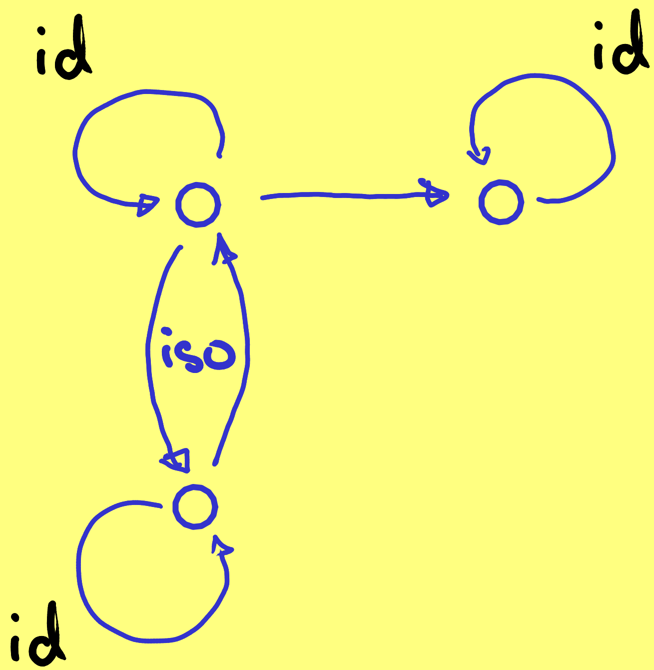


- identity arrows exist
- arrows compose associatively

I.2 CATEGORY Part

... objects are finite sets,

arrows are partial functions.

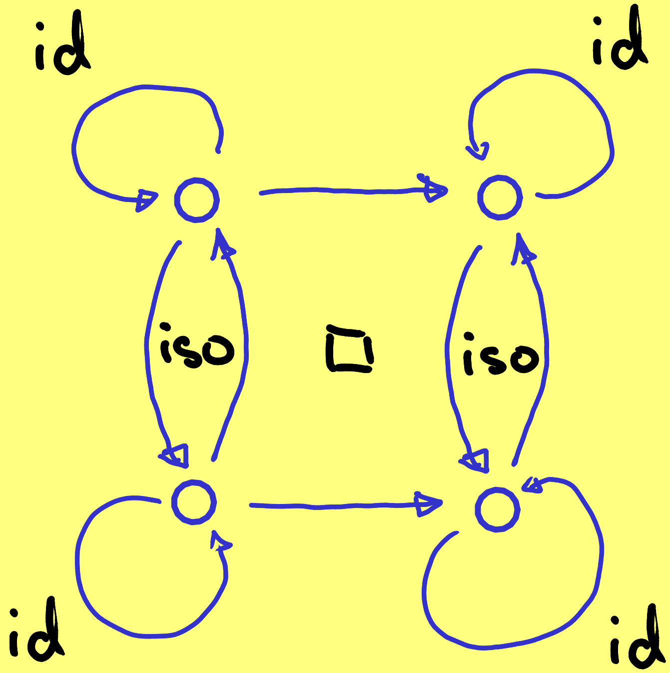


- identity arrows exist
- arrows compose associatively
- objects are isomorphic if connected by invertible arrows

I.2 Category Part

... objects are finite sets,

arrows are partial functions.



- identity arrows exist
- arrows compose associatively
- objects are isomorphic if connected by invertible arrows
- arrows are conjugate if connected by commuting isos.

I.3 CATEGORY Mch

I.3 CATEGORY $\underline{\mathbf{Mch}}$

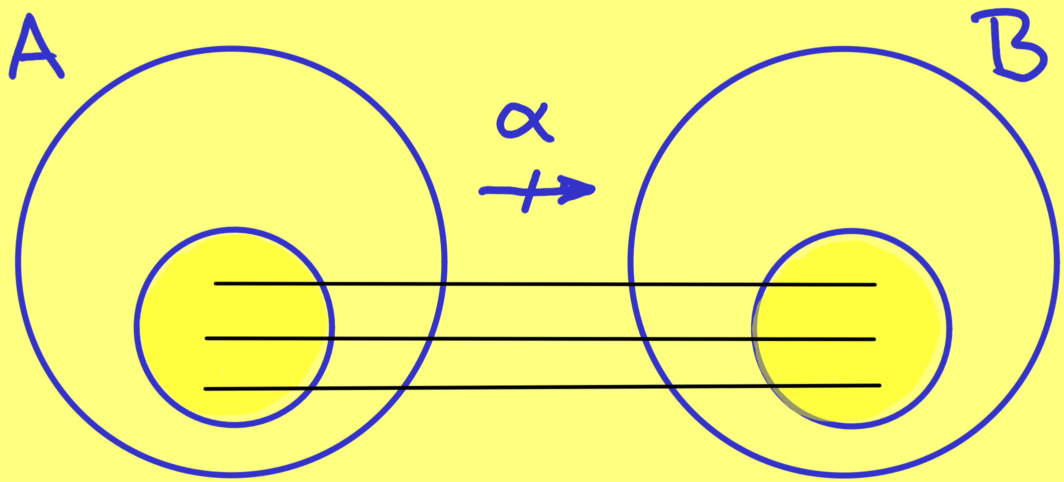
A matching is an injective partial function:

$$\alpha : A \nrightarrow B$$

I.3 CATEGORY Mch

A matching is an injective partial function:

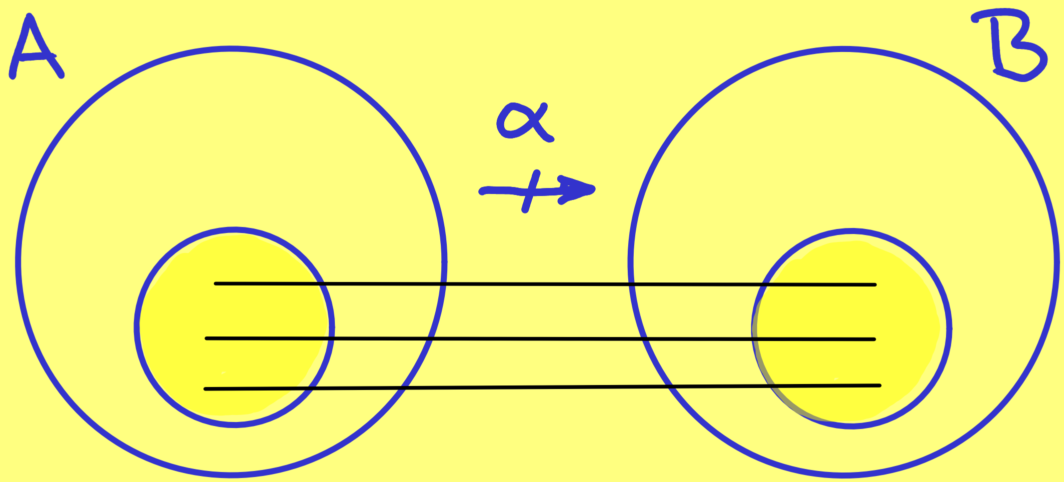
$$\alpha : A \nrightarrow B$$

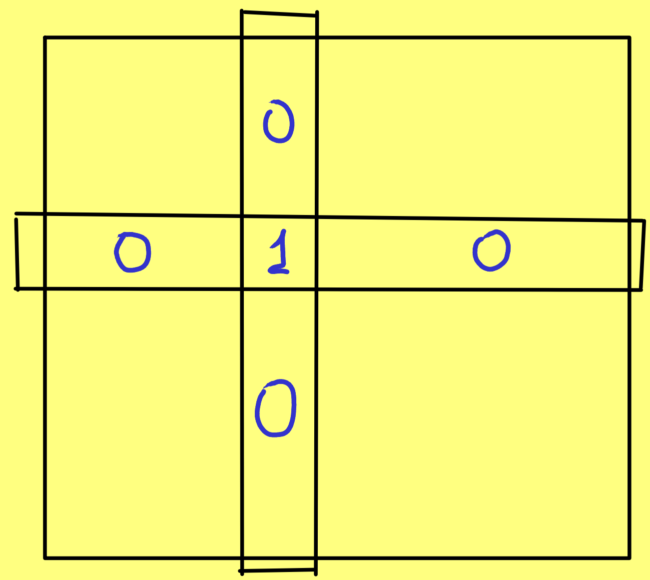


I.3 CATEGORY Mch

A matching is an injective partial function:

$$\alpha : A \nrightarrow B$$

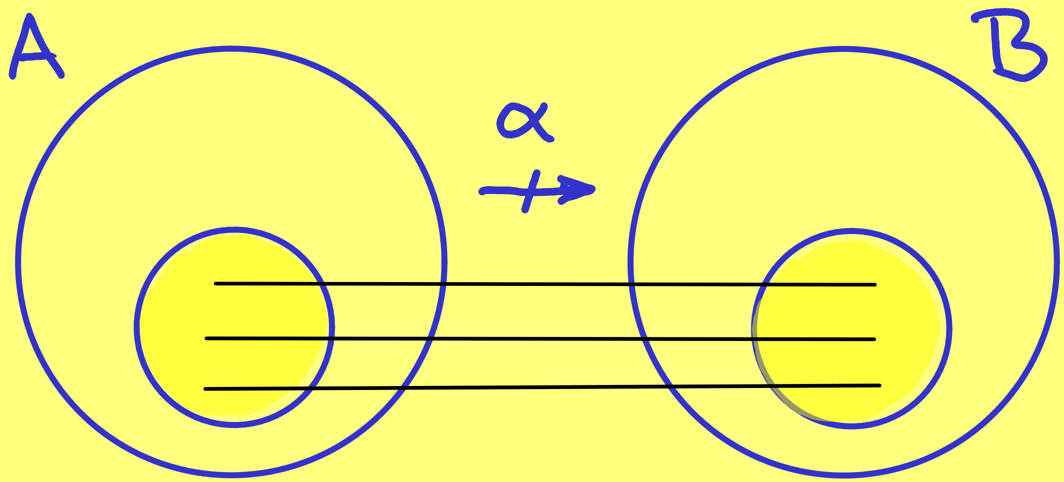


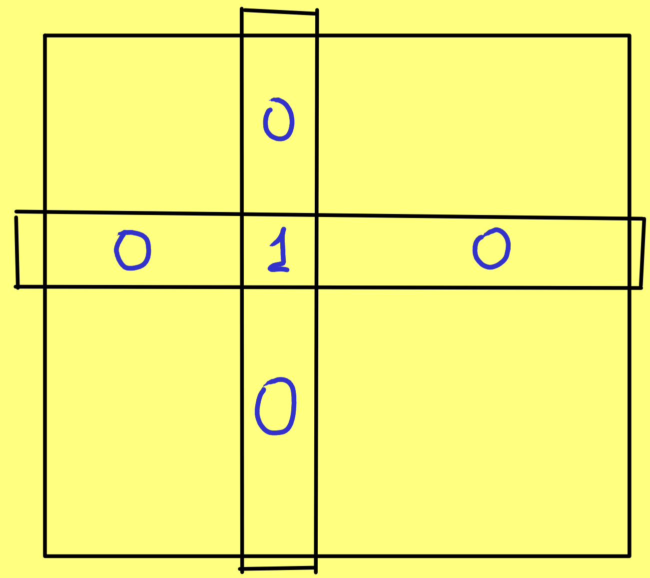


I.3 CATEGORY $\underline{\mathbf{Mch}}$

A **matching** is an injective partial function:

$$\alpha : A \nrightarrow B$$





rank $\alpha = \#\mathrm{dom}\,\alpha = \#\mathrm{im}\,\alpha$

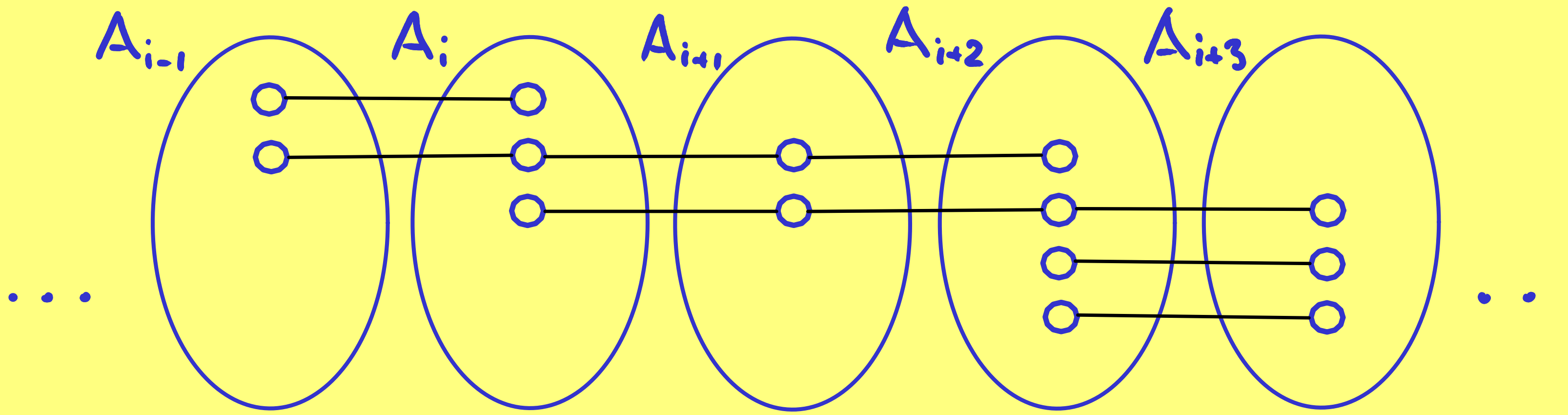
I.4 Tower of Matchings

I.4 Tower of Matchings

A **tower** is a path with finitely many non-zero objects in a category.

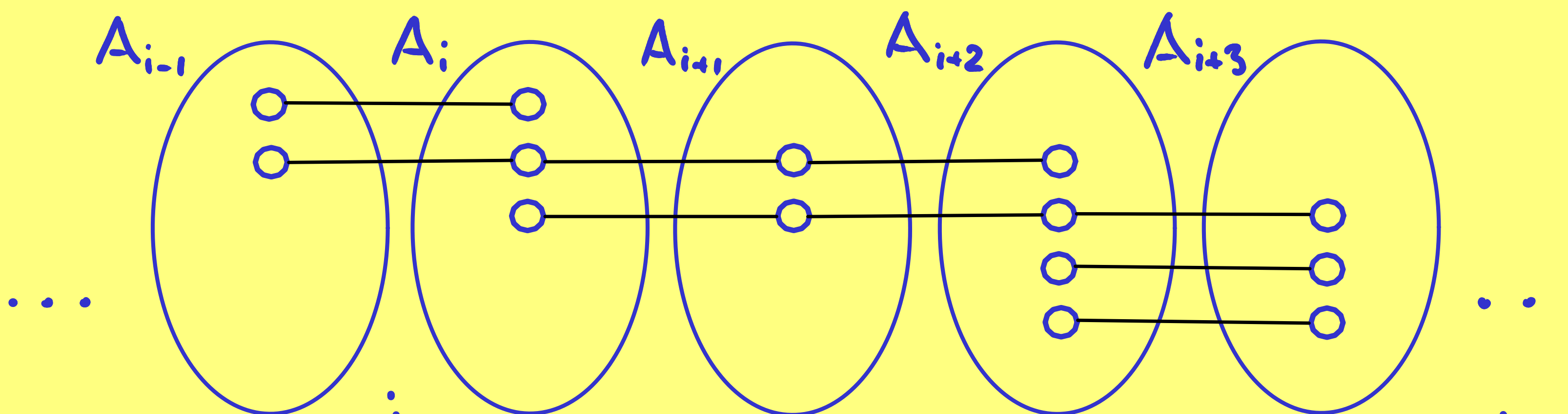
I.4 Tower of Matchings

A tower is a path with finitely many non-zero objects in a category.



I.4 Tower of Matchings

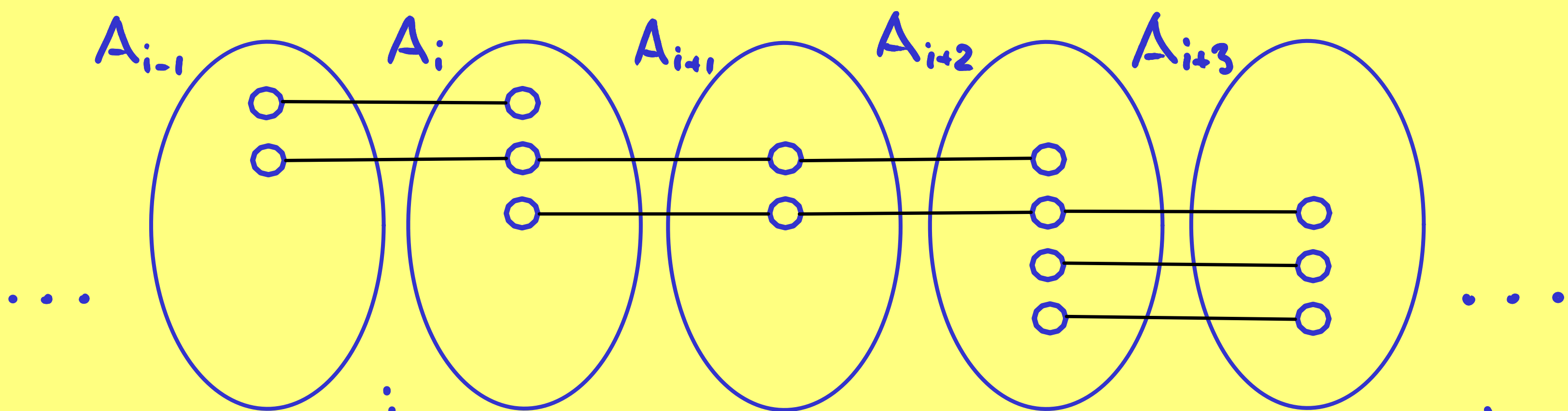
A tower is a path with finitely many non-zero objects in a category.



$\alpha_i^j = \alpha_{j-1} \circ \dots \circ \alpha_i : A_i \nrightarrow A_j \quad \operatorname{rank} \alpha_i^j$

I.4 TOWER OF MATCHINGS

A **tower** is a path with finitely many non-zero objects in a category.

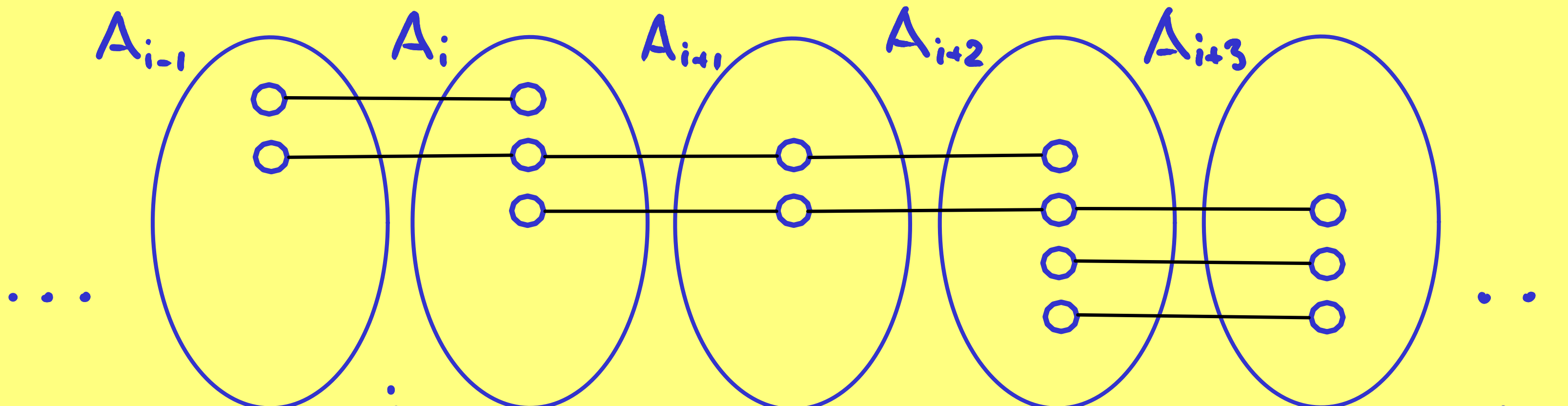


$\alpha_i^j = \alpha_{j-1} \circ \dots \circ \alpha_i : A_i \nrightarrow A_j$ rank α_i^j

THM Towers in $\underline{\mathrm{Mch}}$ are isomorphic

I.4 Tower of Matchings

A tower is a path with finitely many non-zero objects in a category.

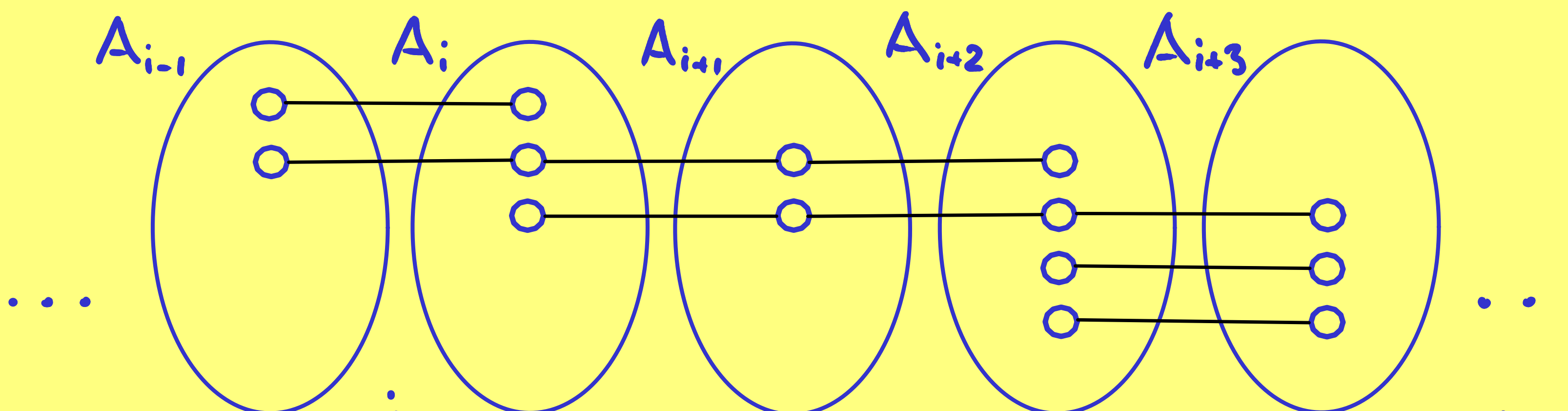


$\alpha_i^j = \alpha_{j-1} \circ \dots \circ \alpha_i : A_i \to A_j$ rank α_i^j

THM Towers in $\underline{\mathsf{Mch}}$ are isomorphic $\iff$ they have the same rank functions

I.4 Tower of Matchings

A **tower** is a path with finitely many non-zero objects in a category.



$\alpha_i^j = \alpha_{j-1} \circ \dots \circ \alpha_i : A_i \to A_j$ rank α_i^j

THM Towers in $\underline{\text{Mch}}$ are isomorphic

$\iff$ they have the same rank functions

$\iff$ their persistence diagrams are the same.

I. Categories

II. Linear Maps

III. Algorithm

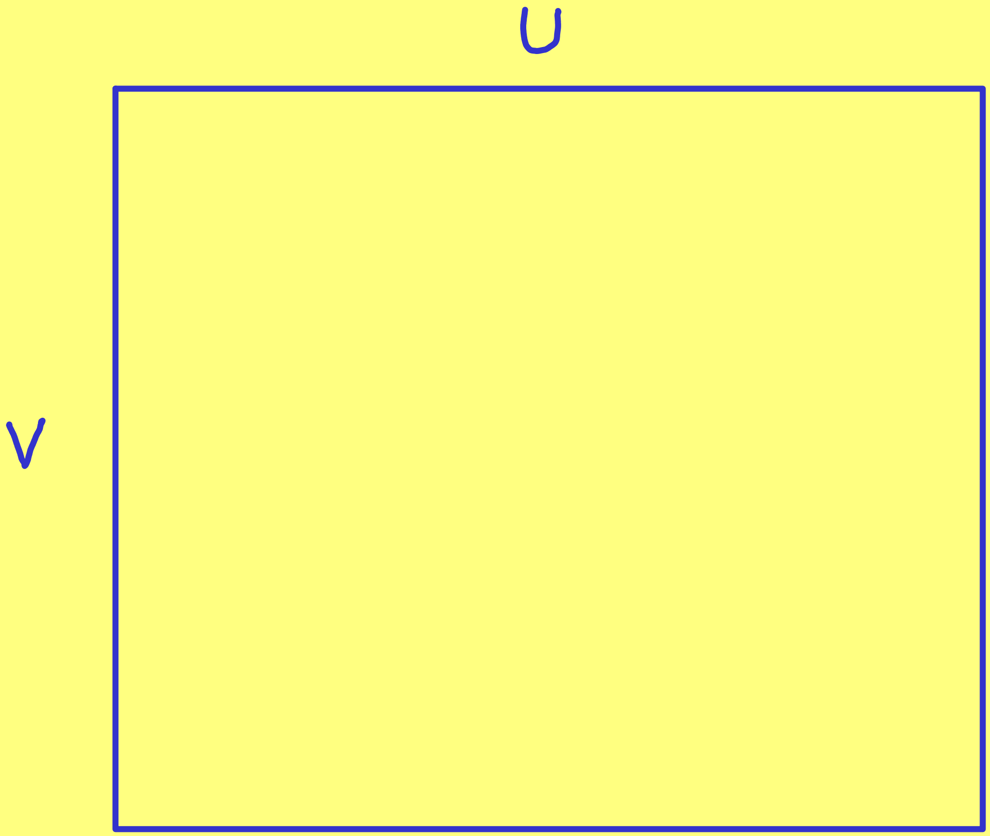
IV. Analysis

II.1 CATEGORY **Vect**

$v : U \to V$

II.1 CATEGORY <u>Vect</u>

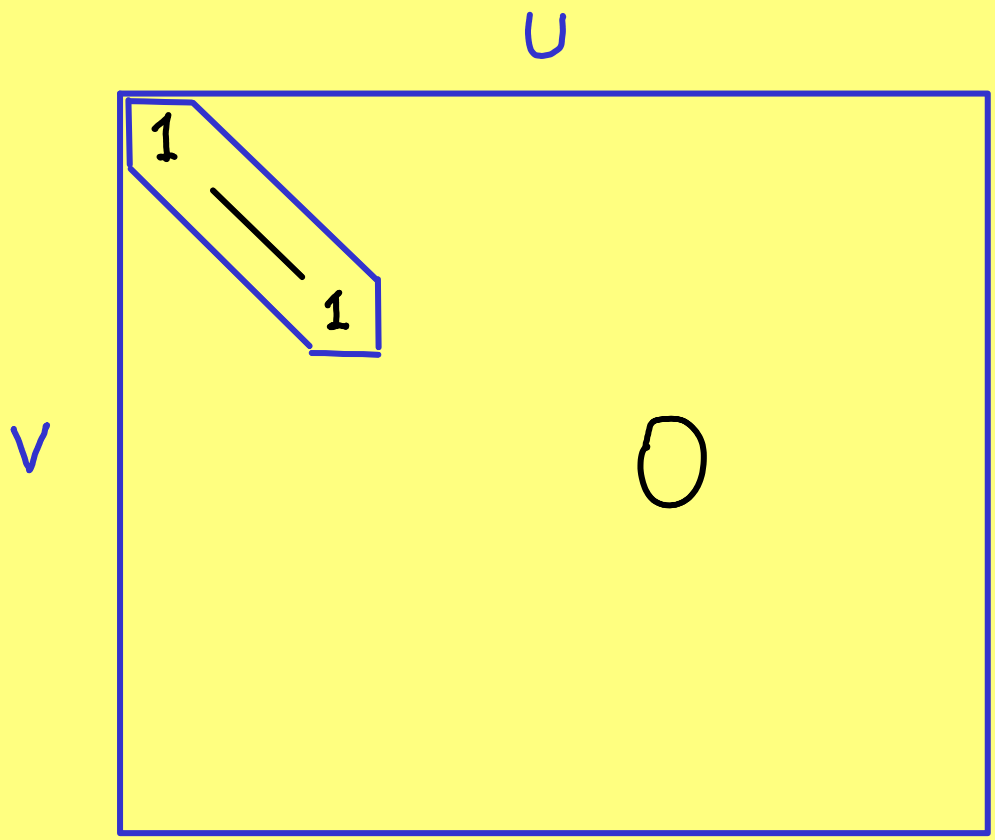
$v : U \to V$



rank v = $\dim \operatorname{im} v$

II.1 CATEGORY Vect

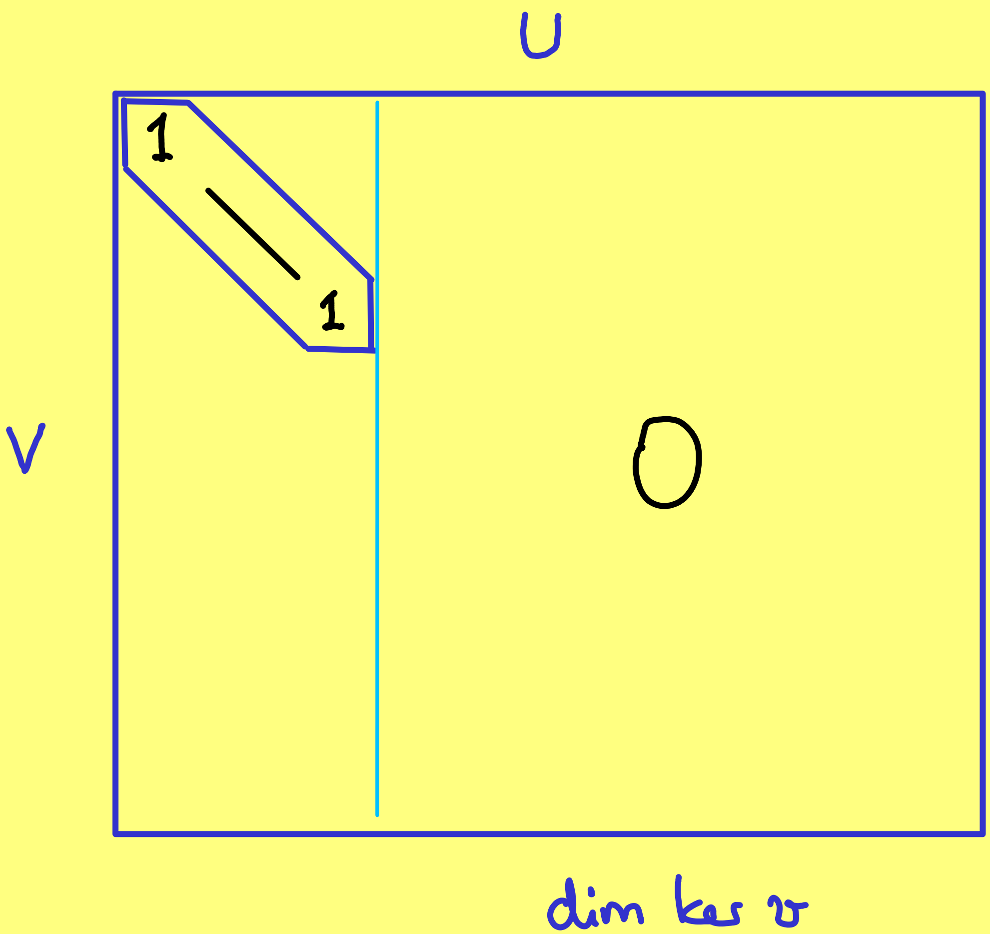
$v : U \to V$



rank v = dim im v

II.1 CATEGORY $\underline{\mathrm{Vect}}$

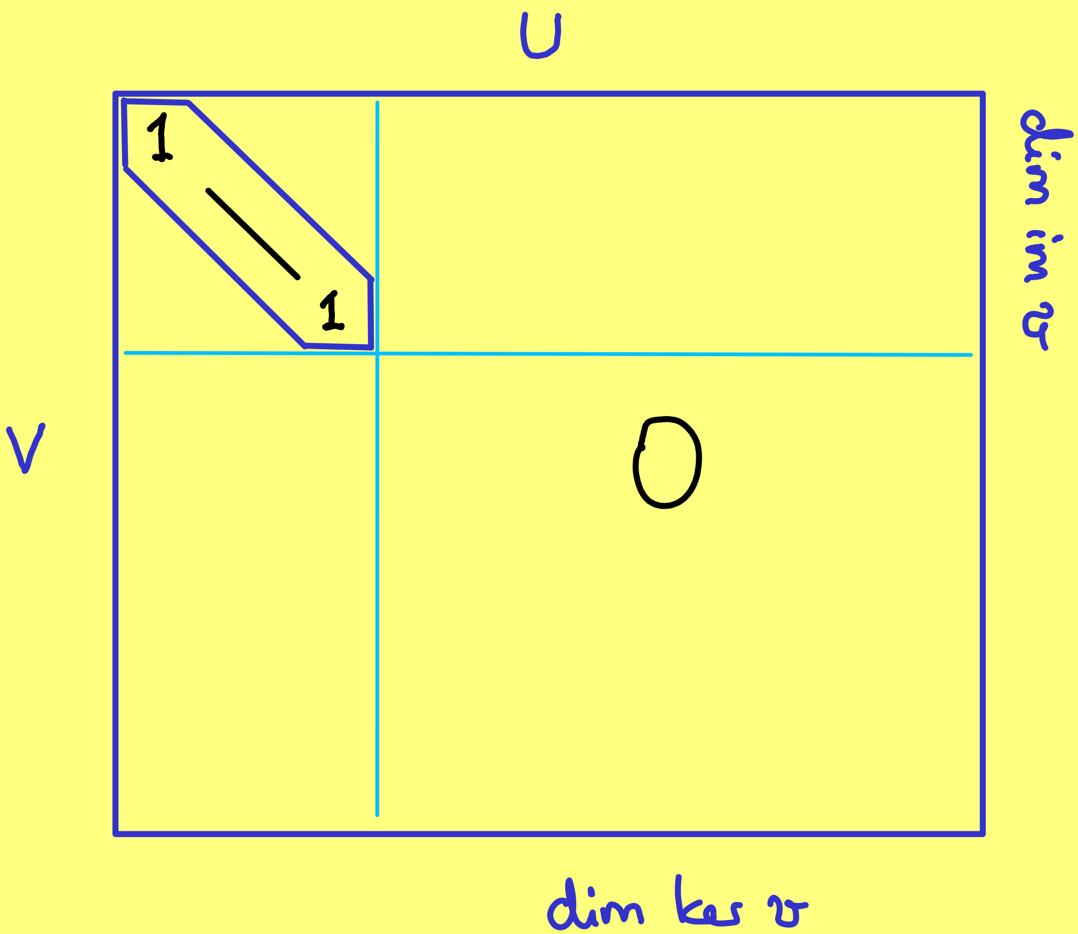
$v : U \to V$



$\mathrm{rank}\, v = \dim \mathrm{im}\, v$

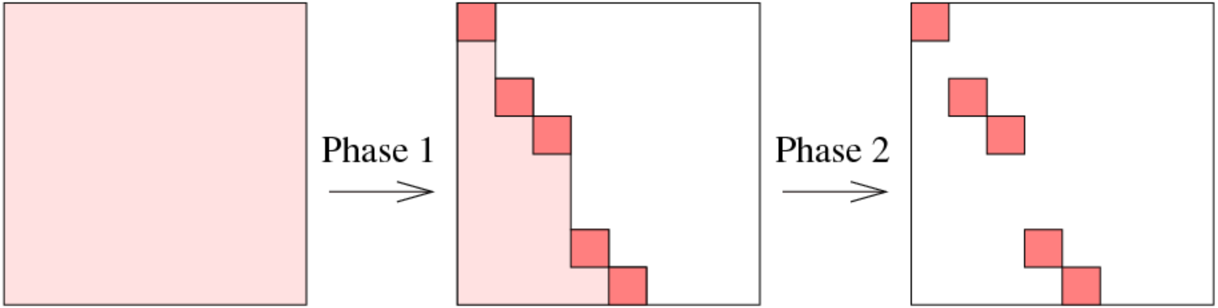
II.1 CATEGORY $\underline{\mathbf{Vect}}$

$v : U \to V$



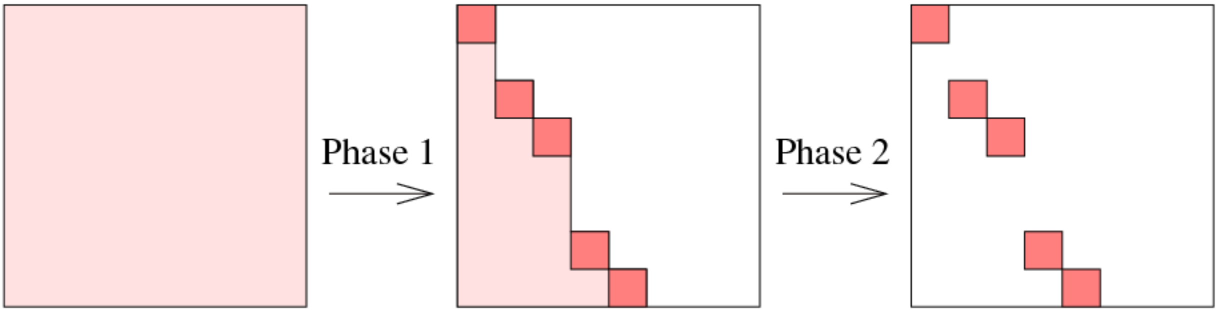
$\text{rank}\, v = \dim \operatorname{im} v$

II.2 Basis Algorithm



II.2 Basis Algorithm

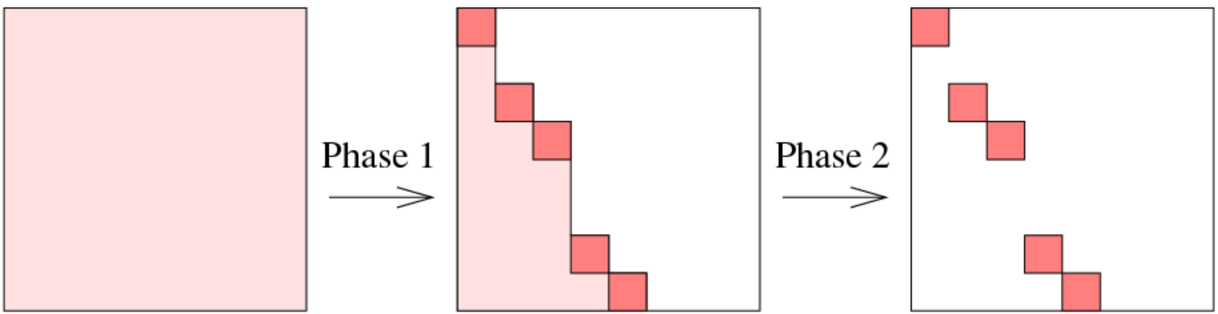
$$U_0 \xrightarrow{v_0} U_1 \xrightarrow{v_1} U_2 \xrightarrow{v_2} \cdots \xrightarrow{v_{n-1}} U_n .$$



II.2 Basis Algorithm

$$U_0 \xrightarrow{v_0} U_1 \xrightarrow{v_1} U_2 \xrightarrow{v_2} \cdots \xrightarrow{v_{n-1}} U_n .$$

DEF. $A_0 \overset{\alpha_0}{\nrightarrow} A_1 \overset{\alpha_1}{\nrightarrow} A_2 \overset{\alpha_2}{\nrightarrow} \cdots \overset{\alpha_{n-1}}{\nrightarrow} A_n$ is a basis if α_i is a matching and $\operatorname{rank} \alpha_i = \operatorname{rank} v_i$, for $0 \le i < n$.

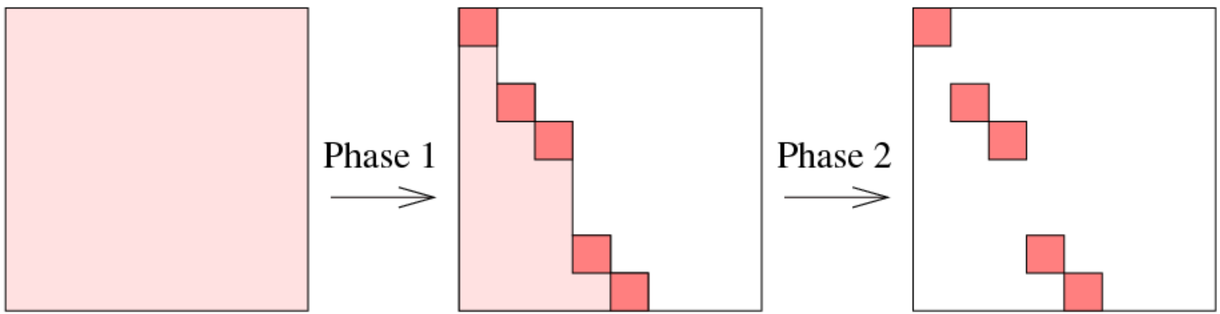


II.2 Basis Algorithm

$$U_0 \xrightarrow{v_0} U_1 \xrightarrow{v_1} U_2 \xrightarrow{v_2} \cdots \xrightarrow{v_{n-1}} U_n .$$

Def. $A_0 \overset{\alpha_0}{\nrightarrow} A_1 \overset{\alpha_1}{\nrightarrow} A_2 \overset{\alpha_2}{\nrightarrow} \cdots \overset{\alpha_{n-1}}{\nrightarrow} A_n$ is a basis if α_i is a matching and $\operatorname{rank} \alpha_i = \operatorname{rank} v_i$, for $0 \le i < n$.

Thm. A basis exists but is generally not unique.

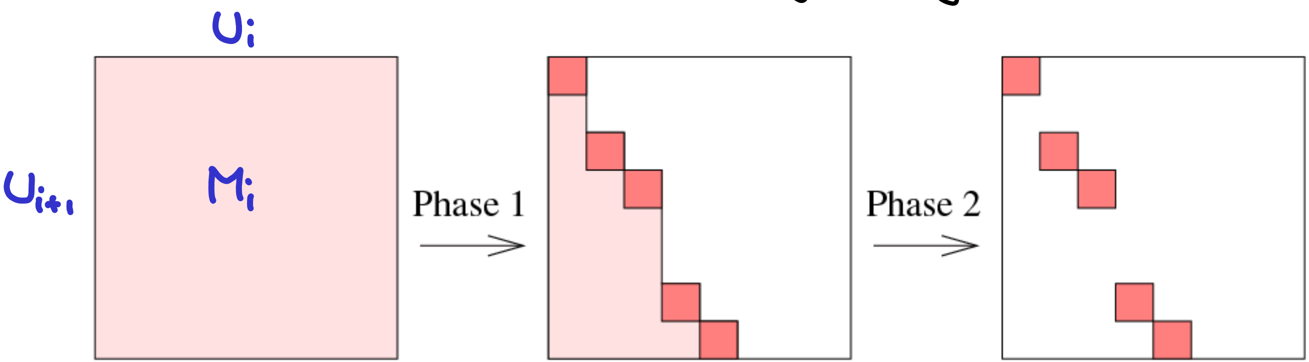


II.2 Basis Algorithm

$$U_0 \xrightarrow{v_0} U_1 \xrightarrow{v_1} U_2 \xrightarrow{v_2} \cdots \xrightarrow{v_{n-1}} U_n .$$

Def. $A_0 \overset{\alpha_0}{\nrightarrow} A_1 \overset{\alpha_1}{\nrightarrow} A_2 \overset{\alpha_2}{\nrightarrow} \cdots \overset{\alpha_{n-1}}{\nrightarrow} A_n$ is a basis if α_i is a matching and $\operatorname{rank} \alpha_i = \operatorname{rank} v_i$, for $0 \le i < n$.

Thm. A basis exists but is generally not unique.

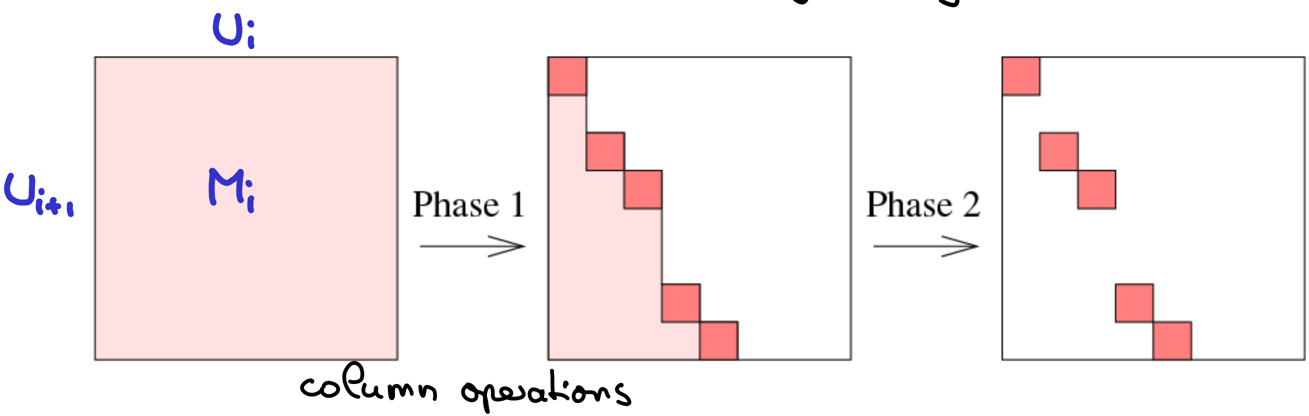


II.2 Basis Algorithm

$$U_0 \xrightarrow{v_0} U_1 \xrightarrow{v_1} U_2 \xrightarrow{v_2} \cdots \xrightarrow{v_{n-1}} U_n .$$

Def. $A_0 \overset{\alpha_0}{\nrightarrow} A_1 \overset{\alpha_1}{\nrightarrow} A_2 \overset{\alpha_2}{\nrightarrow} \cdots \overset{\alpha_{n-1}}{\nrightarrow} A_n$ is a basis if α_i is a matching and $\operatorname{rank} \alpha_i = \operatorname{rank} v_i$, for $0 \le i < n$.

Thm. A basis exists but is generally not unique.

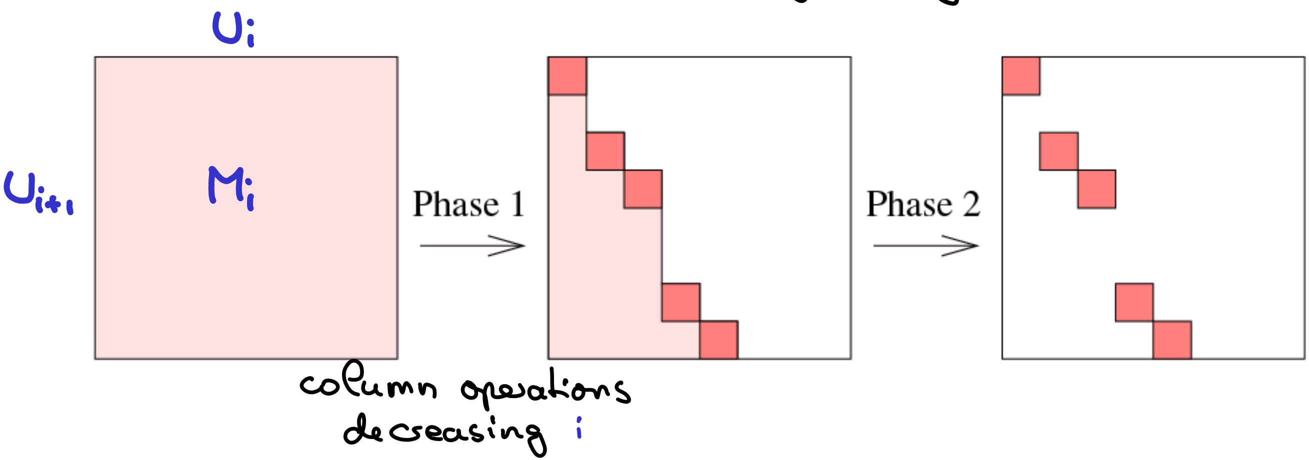


II.2 Basis Algorithm

$$U_0 \xrightarrow{v_0} U_1 \xrightarrow{v_1} U_2 \xrightarrow{v_2} \cdots \xrightarrow{v_{n-1}} U_n .$$

DEF. $A_0 \overset{\alpha_0}{\nrightarrow} A_1 \overset{\alpha_1}{\nrightarrow} A_2 \overset{\alpha_2}{\nrightarrow} \cdots \overset{\alpha_{n-1}}{\nrightarrow} A_n$ is a basis if α_i is a matching and $\operatorname{rank} \alpha_i = \operatorname{rank} v_i$, for $0 \le i < n$.

THM. A basis exists but is generally not unique.

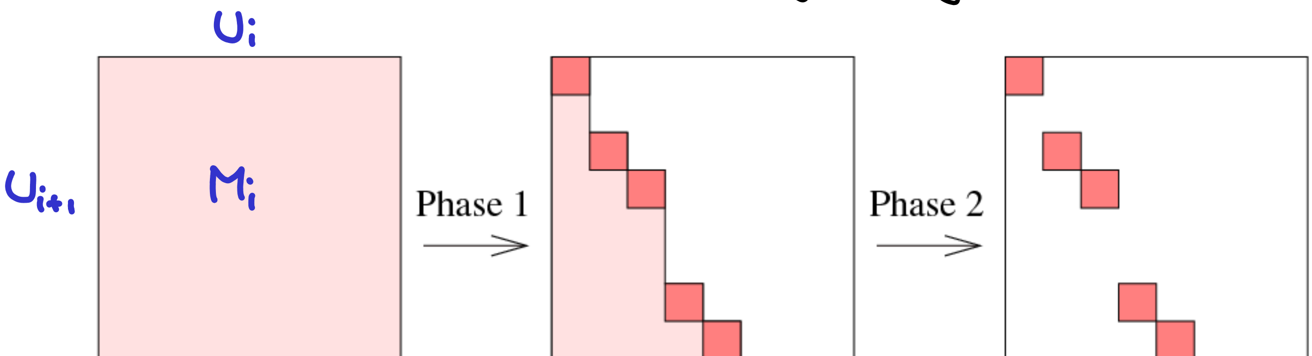


II.2 Basis Algorithm

$$U_0 \xrightarrow{v_0} U_1 \xrightarrow{v_1} U_2 \xrightarrow{v_2} \cdots \xrightarrow{v_{n-1}} U_n .$$

Def. $A_0 \overset{\alpha_0}{\nrightarrow} A_1 \overset{\alpha_1}{\nrightarrow} A_2 \overset{\alpha_2}{\nrightarrow} \cdots \overset{\alpha_{n-1}}{\nrightarrow} A_n$ is a basis if α_i is a matching and $\operatorname{rank} \alpha_i = \operatorname{rank} v_i$, for $0 \leq i < n$.

Thm. A basis exists but is generally not unique.



column operations decreasing i

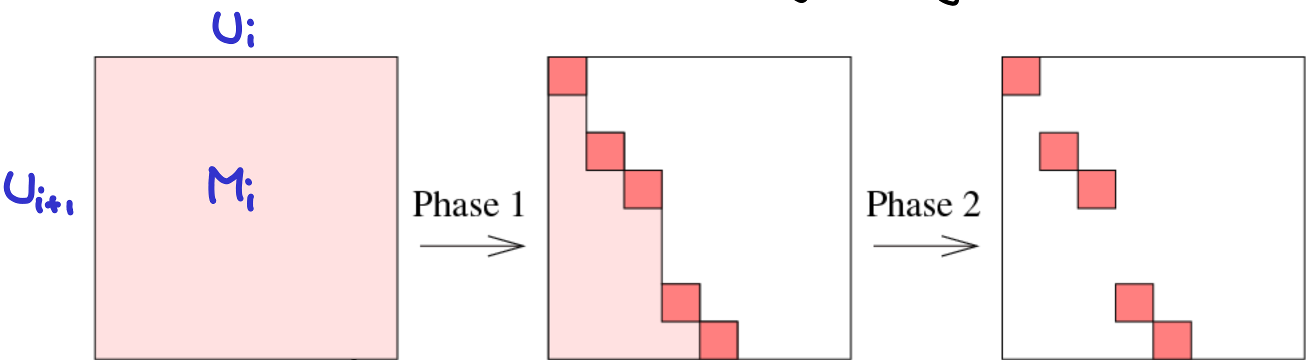
top down row operations

II.2 Basis Algorithm

$$U_0 \xrightarrow{v_0} U_1 \xrightarrow{v_1} U_2 \xrightarrow{v_2} \cdots \xrightarrow{v_{n-1}} U_n .$$

Def. $A_0 \overset{\alpha_0}{\nrightarrow} A_1 \overset{\alpha_1}{\nrightarrow} A_2 \overset{\alpha_2}{\nrightarrow} \cdots \overset{\alpha_{n-1}}{\nrightarrow} A_n$ is a basis if α_i is a matching and $\operatorname{rank} \alpha_i = \operatorname{rank} v_i$, for $0 \leq i < n$.

Thm. A basis exists but is generally not unique.



column operations decreasing i

top down row operations increasing i

II.3 EIGENSPACES

II.3 Eigenspaces

$$U \xrightarrow{\phi} U$$

II.3 EIGENSPACES

$U \xrightarrow{\phi} U$

$\Rightarrow$

$\{u \mid \phi(u) = tu\}$

$\parallel$

$E_t(\phi)$

II.3 EIGENSPACES

$$\begin{tikzcd} U \arrow[r, "\phi"] \arrow[d, "v"'] & U \arrow[d, "v"] \\ U' \arrow[r, "\phi'"] & U' \end{tikzcd} \quad \Rightarrow \quad \begin{tikzcd} \{u \mid \phi(u) = tu\} \arrow[d, phantom, "\shortparallel"] \\ E_t(\phi) \arrow[d] \\ E_t(\phi') \end{tikzcd}$$

II.3 Eigenspaces

$$\{u \mid \phi(u) = tu\}$$

$$\begin{tikzcd} U \arrow[r, "\phi"] \arrow[d, "v"'] & U \arrow[d, "v"] \\ U' \arrow[r, "\phi'"] & U' \end{tikzcd}$$

functor E_t from $\Rightarrow$ Endo(Vect) to Vect

$$\begin{tikzcd} E_t(\phi) \arrow[d] \\ E_t(\phi') \end{tikzcd}$$

where $\{u \mid \phi(u) = tu\} = E_t(\phi)$.

II.3 EIGENSPACES

$$\begin{array}{ccc} U & \xrightarrow{\phi} & U \\ {\scriptstyle v}\downarrow & & \downarrow{\scriptstyle v} \\ U' & \xrightarrow{\phi'} & U' \end{array}$$

functor E_t from

$\Rightarrow$

Endo(Vect) to Vect

$$\{u \mid \phi(u) = tu\}$$

$$\begin{array}{c} \| \\ E_t(\phi) \\ \downarrow \\ E_t(\phi') \end{array}$$

$$\begin{array}{ccccc} V & \xleftarrow{\varphi} & U & \xrightarrow{\psi} & V \\ \downarrow & & \downarrow & & \downarrow \\ V' & \xleftarrow{\varphi'} & U' & \xrightarrow{\psi'} & V' \end{array}$$

II.3 EIGENSPACES

$$\begin{tikzcd} U \arrow[r, "\phi"] \arrow[d, "v"'] & U \arrow[d, "v"] \\ U' \arrow[r, "\phi'"] & U' \end{tikzcd} \quad \text{functor } E_t \text{ from } \underline{\text{Endo}}(\underline{\text{Vect}}) \text{ to } \underline{\text{Vect}} \quad \Rightarrow \quad \begin{tikzcd} \{u \mid \phi(u) = tu\} \arrow[d, phantom, "\shortparallel"] \\ E_t(\phi) \arrow[d] \\ E_t(\phi') \end{tikzcd}$$

$$\begin{tikzcd} V \arrow[d] & U \arrow[l, "\varphi"'] \arrow[r, "\psi"] \arrow[d] & V \arrow[d] \\ V' & U' \arrow[l, "\varphi'"'] \arrow[r, "\psi'"] & V' \end{tikzcd} \quad \Rightarrow \quad \begin{tikzcd} E_t(\varphi, \psi) \arrow[d] \\ E_t(\varphi', \psi') \end{tikzcd}$$

II.3 Eigenspaces

$$\begin{tikzcd} U \arrow[r, "\phi"] \arrow[d, "v"'] & U \arrow[d, "v"] \\ U' \arrow[r, "\phi'"] & U' \end{tikzcd}$$

functor E_t from

$\Rightarrow$

Endo(Vect) to Vect

$$\begin{tikzcd} \{u \mid \phi(u) = tu\} \arrow[d, equal] \\ E_t(\phi) \arrow[d] \\ E_t(\phi') \end{tikzcd}$$

$$\begin{tikzcd} V \arrow[d] & U \arrow[l, "\varphi"'] \arrow[r, "\psi"] \arrow[d] & V \arrow[d] \\ V' & U' \arrow[l, "\varphi'"'] \arrow[r, "\psi'"] & V' \end{tikzcd}$$

$\Rightarrow$

$$\begin{tikzcd} \{u \mid \varphi(u) = t\psi(u)\} / \ker\varphi \cap \ker\psi \arrow[d, equal] \\ E_t(\varphi, \psi) \arrow[d] \\ E_t(\varphi', \psi') \end{tikzcd}$$

II.3 Eigenspaces

$$\begin{tikzcd} U \arrow[r, "\phi"] \arrow[d, "v"'] & U \arrow[d, "v"] \\ U' \arrow[r, "\phi'"] & U' \end{tikzcd}$$

functor E_t from

$\Rightarrow$

Endo(Vect) to Vect

$$\begin{tikzcd} \{u \mid \phi(u) = tu\} \arrow[d, phantom, "\shortparallel"] \\ E_t(\phi) \arrow[d] \\ E_t(\phi') \end{tikzcd}$$

$$\begin{tikzcd} V & U \arrow[l, "\varphi"'] \arrow[r, "\psi"] \arrow[d] & V \\ V' \arrow[from=u] & U' \arrow[l, "\varphi'"'] \arrow[r, "\psi'"] & V' \arrow[from=u] \end{tikzcd}$$

functor E_t from

$\Rightarrow$

Pairs (Vect) to Vect

$$\begin{tikzcd} \{u \mid \varphi(u) = t\psi(u)\} / \ker\varphi \cap \ker\psi \arrow[d, phantom, "\shortparallel"] \\ E_t(\varphi, \psi) \arrow[d] \\ E_t(\varphi', \psi') \end{tikzcd}$$

I. Categories

II. Linear Maps

III. Algorithm

IV. Analysis

III.1 SETTING

$f : \mathbb{X} \to \mathbb{X}$ is continuous self-map

III.1 SETTING

$f : \mathbb{X} \to \mathbb{X}$ is continuous self-map

invariant set is $N \subseteq \mathbb{X}$ with $f(N) = N$.

III.1 SETTING

$f : \mathbb{X} \to \mathbb{X}$ is continuous self-map

invariant set is $N \subseteq \mathbb{X}$ with $f(N) = N$.

General approach:

1.

2.

3.

4.

III.1 SETTING

$f : \mathbb{X} \to \mathbb{X}$ is continuous self-map

invariant set is $N \subseteq \mathbb{X}$ with $f(N) = N$.

General approach:

1. define hierarchy of representations;

2.

3.

4.

III.1 SETTING

$f : \mathbb{X} \to \mathbb{X}$ is continuous self-map

invariant set is $N \subseteq \mathbb{X}$ with $f(N) = N$.

General approach:

1. define hierarchy of representations;
2. apply homology functor;
3.
4.

III.1 SETTING

$f : \mathbb{X} \to \mathbb{X}$ is continuous self-map

invariant set is $N \subseteq \mathbb{X}$ with $f(N) = N$.

General approach:

1. define hierarchy of representations;
2. apply homology functor;
3. compute eigenspaces of induced linear maps;
4.

III.1 SETTING

$f: \mathbb{X} \to \mathbb{X}$ is continuous self-map

invariant set is $N \subseteq \mathbb{X}$ with $f(N) = N$.

General approach:

1. define hierarchy of representations;
2. apply homology functor;
3. compute eigenspaces of induced linear maps;
4. get persistence.

III.2 RESULTS for $f(x) = x^2$

III.2 RESULTS for $f(x) = x^2$

Data generation :

III.2 RESULTS for $f(x) = x^2$

Data generation :

pick m points z_j equally spaced on $\mathbb{S}^1$;

III.2 RESULTS for $f(x) = x^2$

Data generation:

pick m points z_j equally spaced on S^1;

choose the point x_j in nbhd. of z_j, with isotropic Gaussian noise of width σ;

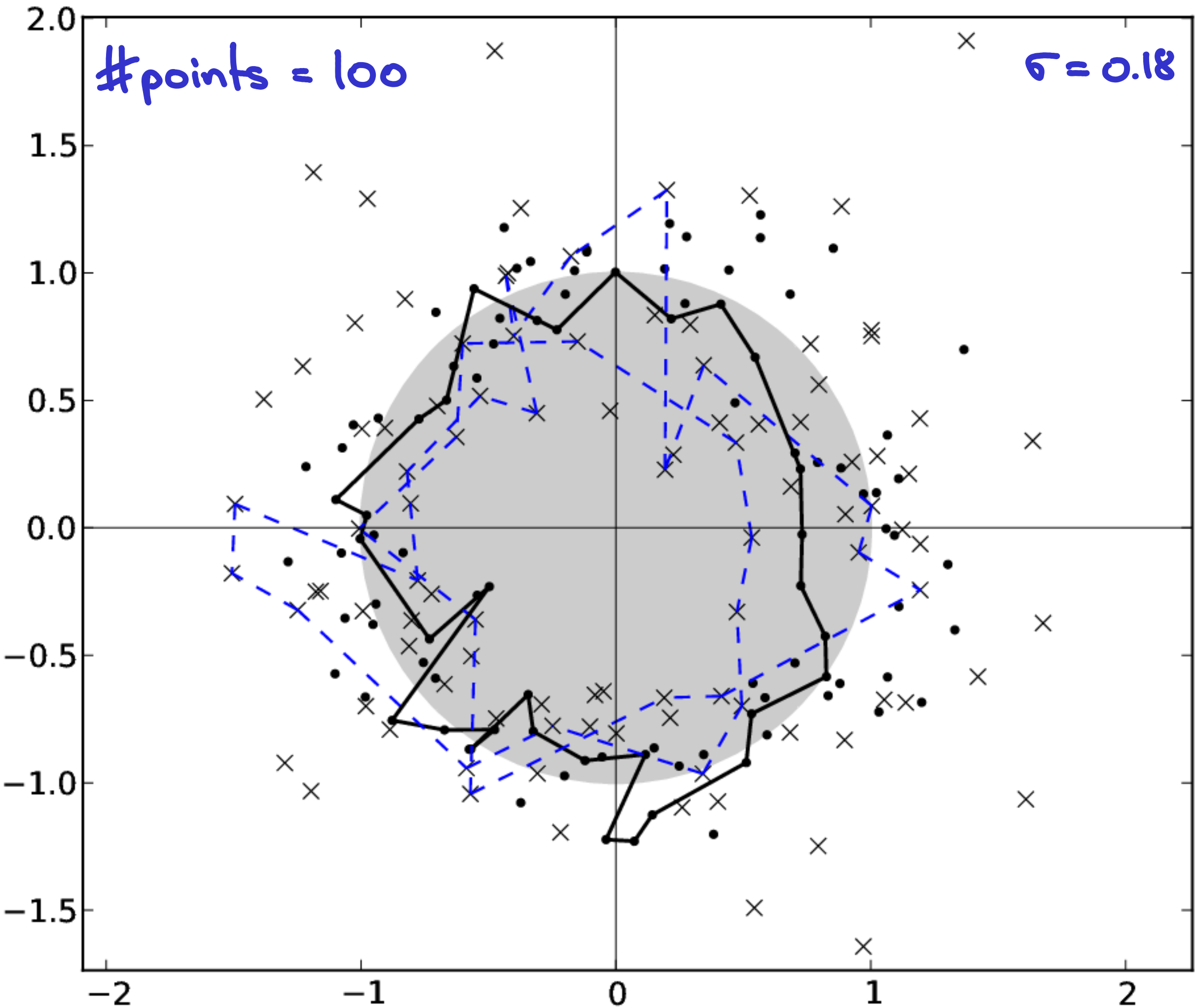
III.2 RESULTS for $f(x) = x^2$

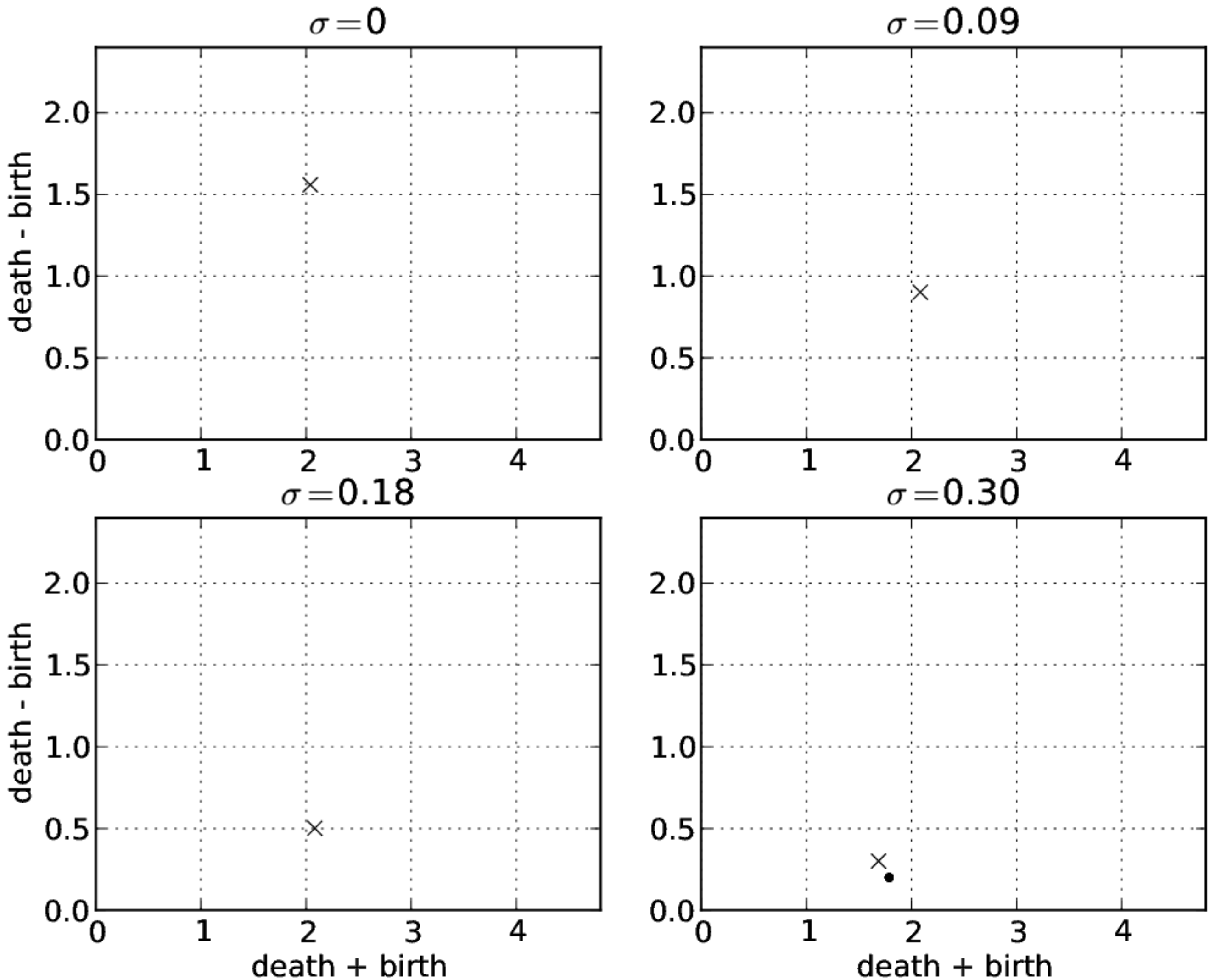
Data generation:

pick m points z_j equally spaced on S^1;

choose the point x_j in nbhd. of z_j, with isotropic Gaussian noise of width σ;

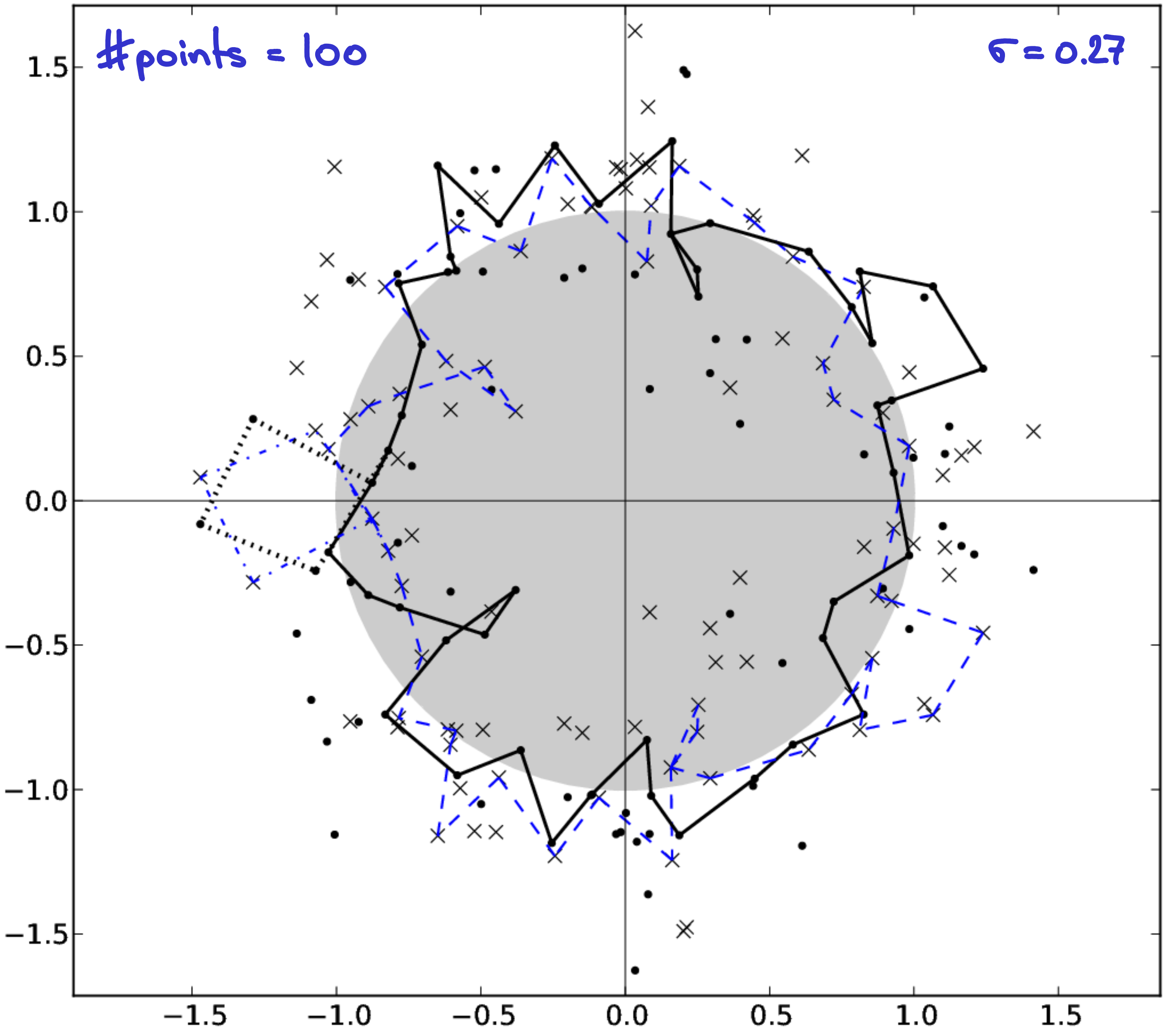
set $f(x_j)$ equal to x_i nearest to x_j^2.


#points = 100
σ = 0.18

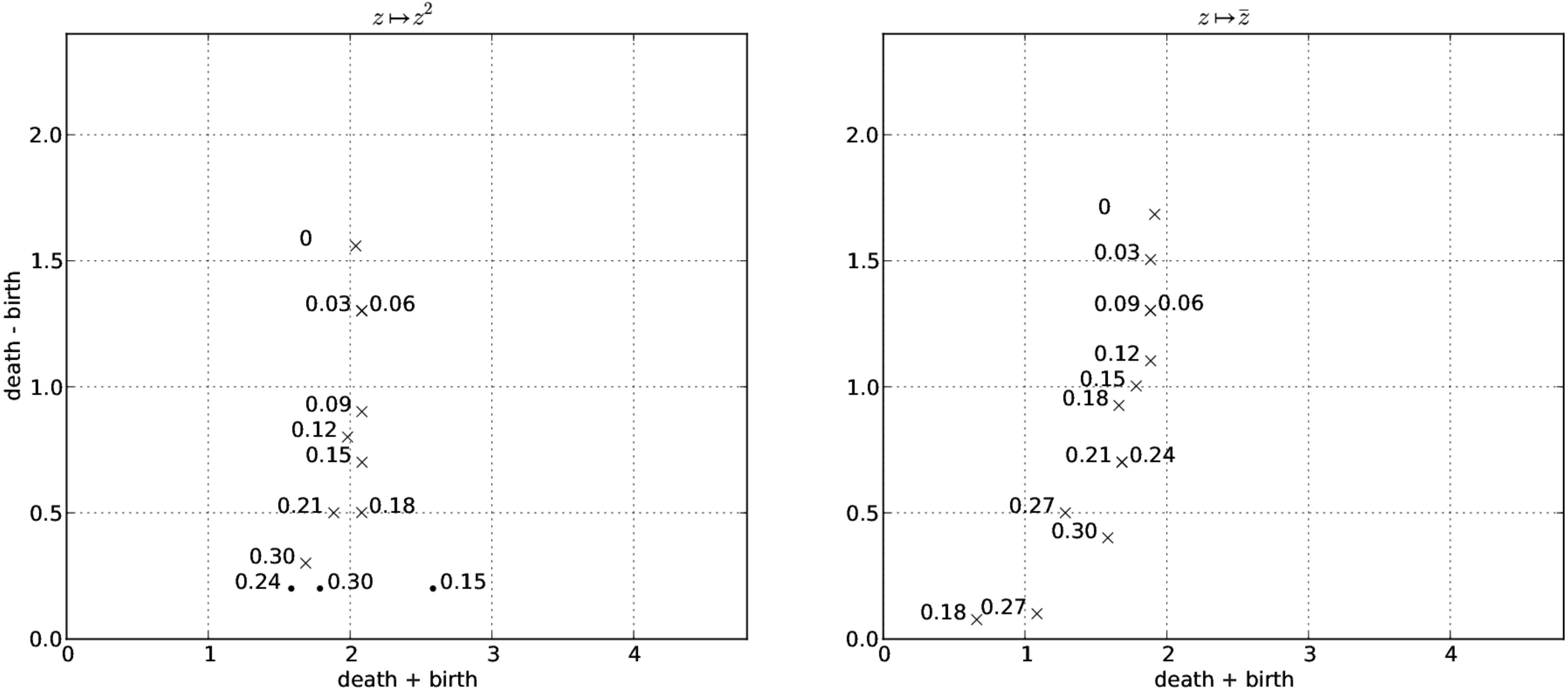


$\sigma = 0$
$\sigma = 0.09$
$\sigma = 0.18$
$\sigma = 0.30$
death - birth
death + birth
0.0
0.5
1.0
1.5
2.0
0
1
2
3
4

III.3 Results for $f(x) = \bar{x}$



#points = 100
σ = 0.27
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5



$z \mapsto z^2$
$z \mapsto \bar{z}$
death - birth
death + birth
0
0.03
0.06
0.09
0.12
0.15
0.18
0.21
0.24
0.27
0.30

I. Categories

II. Linear Maps

III. Algorithm

IV. Analysis

IV.1 Graphs and Distances

IV.1 Graphs and Distances

maps $\quad f : X \to X, \qquad g : S \to S$

IV.1 Graphs and Distances

maps $\quad f : X \to X, \qquad g : S \to S$

graphs $\quad Gf = \{(x, f(x))\}, \qquad Gg = \{(s, g(s))\}$

IV.1 Graphs and Distances

maps $f : \mathbb{X} \to \mathbb{X}$, $g : S \to S$

graphs $G_f = \{(x, f(x))\}$, $G_g = \{(s, g(s))\}$

Hausdorff distance satisfies $\mathrm{Hdf}(\mathbb{X}, S) \leq \mathrm{Hdf}(G_f, G_g)$.

IV.1 Graphs and Distances

maps $\quad f : \mathbb{X} \to \mathbb{X}, \qquad g : S \to S$

graphs $\quad Gf = \{(x, f(x))\}, \quad Gg = \{(s, g(s))\}$

Hausdorff distance satisfies $\quad \mathrm{Hdf}(\mathbb{X}, S) \leq \mathrm{Hdf}(Gf, Gg)$.

distance functions $\quad d_{\mathbb{X}}, d_S : \mathbb{R}^\ell \to \mathbb{R}$

$$d_{Gf}, d_{Gg} : \mathbb{R}^\ell \times \mathbb{R}^\ell \to \mathbb{R}$$

IV.1 Graphs and Distances

maps $f : \mathbb{X} \to X$, $g : S \to S$

graphs $G_f = \{(x, f(x))\}$, $G_g = \{(s, g(s))\}$

Hausdorff distance satisfies $\mathrm{Hdf}(\mathbb{X}, S) \leq \mathrm{Hdf}(G_f, G_g)$.

distance functions $d_{\mathbb{X}}, d_S : \mathbb{R}^\ell \to \mathbb{R}$

$d_{G_f}, d_{G_g} : \mathbb{R}^\ell \times \mathbb{R}^\ell \to \mathbb{R}$

if $\varepsilon \geq \mathrm{Hdf}(G_f, G_g)$ then sublevel sets satisfy

$\mathbb{X}_r \subseteq S_{r+\varepsilon} \subseteq \mathbb{X}_{r+2\varepsilon}$ and $G_{f_r} \subseteq G_{g_{r+\varepsilon}} \subseteq G_{f_{r+2\varepsilon}}$.

IV.2 INTERLEAVING

IV.2 INTERLEAVING

$$
\begin{tikzcd}
\mathbb{U}_r \arrow[d] & G h_r \arrow[l] \arrow[r] \arrow[d] & \mathbb{U}_r \arrow[d] \\
\mathbb{V}_{r'} & G k_{r'} \arrow[l] \arrow[r] & \mathbb{V}_{r'}
\end{tikzcd}
$$

IV.2 Interleaving

$$
\begin{tikzcd}
\mathbb{U}_r & Gh_r \arrow[l] \arrow[r] \arrow[d] & U_r \arrow[d] \\
\mathbb{V}_{r'} \arrow[from=u] & Gk_{r'} \arrow[l] \arrow[r] & V_{r'}
\end{tikzcd}
\quad \xRightarrow{\;H\;} \quad \xRightarrow{\;E_t\;} \quad
\begin{array}{c}
\epsilon^{r'}_{t,r} : E_t(\varphi_r, \psi_r) \\
\downarrow \\
E_t(\upsilon_{r'}, \nu_{r'})
\end{array}
$$

IV.2 INTERLEAVING

$$\begin{tikzcd} \mathbb{U}_r \arrow[d] & Gh_r \arrow[l] \arrow[r] \arrow[d] & \mathbb{U}_r \arrow[d] \\ \mathbb{V}_{r'} & Gk_{r'} \arrow[l] \arrow[r] & \mathbb{V}_{r'} \end{tikzcd} \quad \overset{H}{\Longrightarrow} \quad \overset{E_t}{\Longrightarrow} \quad \begin{tikzcd} \epsilon_{t,r}^{r'} : E_t(\varphi_r, \psi_r) \arrow[d] \\ E_t(\upsilon_{r'}, \nu_{r'}) \end{tikzcd}$$

INTERLEAVING LEMMA A. Let $\mathbb{U}, \mathbb{V} \subseteq \mathbb{R}^\ell$ and $h : \mathbb{U} \to \mathbb{U}$, $k : \mathbb{V} \to \mathbb{V}$ with tame distance functions.

IV.2 INTERLEAVING

$$
\begin{tikzcd}
\mathbb{U}_r \arrow[d] & Gh_r \arrow[l] \arrow[r] \arrow[d] & \mathbb{U}_r \arrow[d] \\
\mathbb{V}_{r'} & Gk_{r'} \arrow[l] \arrow[r] & \mathbb{V}_{r'}
\end{tikzcd}
\quad \overset{H}{\Longrightarrow} \quad \overset{E_t}{\Longrightarrow} \quad
\varepsilon_{t,r}^{r'} : \begin{array}{c} E_t(\varphi_r, \psi_r) \\ \downarrow \\ E_t(\upsilon_{r'}, \nu_{r'}) \end{array}
$$

INTERLEAVING LEMMA A. Let $\mathbb{U}, \mathbb{V} \subseteq \mathbb{R}^\ell$ and $h : \mathbb{U} \to \mathbb{U}$, $k : \mathbb{V} \to \mathbb{V}$ with tame distance functions. Set $\varepsilon = \mathrm{Hdf}(Gh, Gk)$.

IV.2 Interleaving

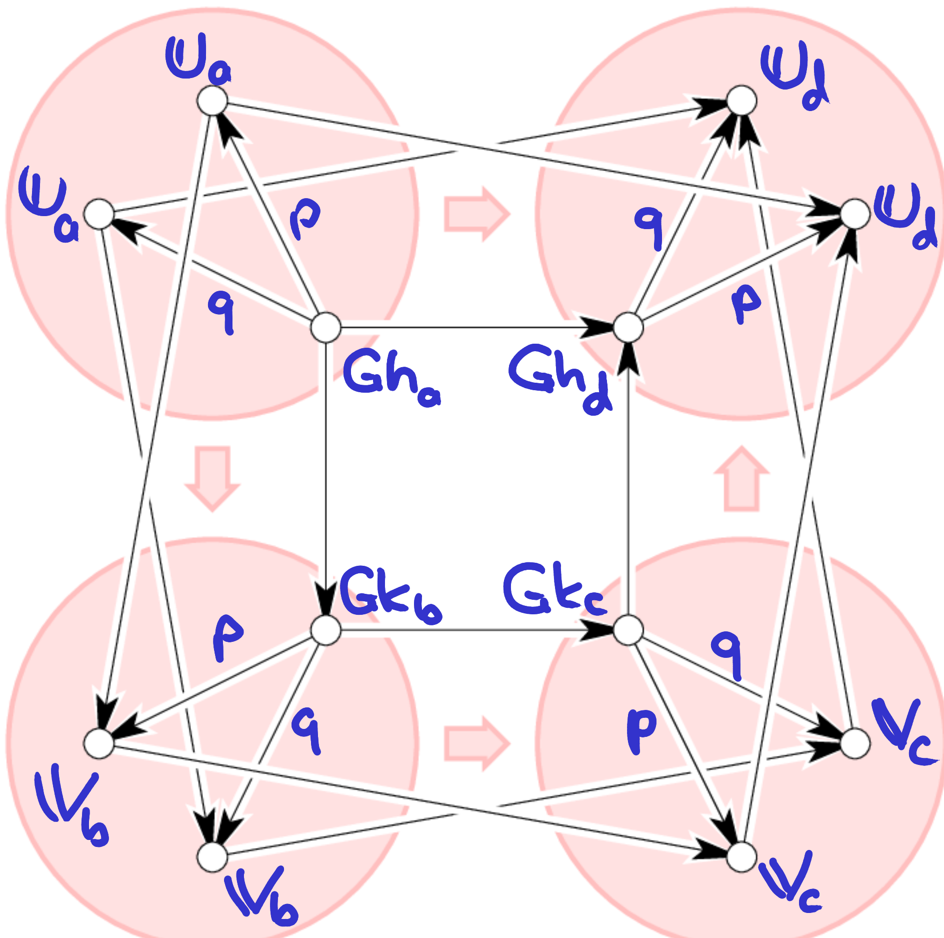
$$\begin{tikzcd} \mathbb{U}_r \arrow[d] & Gh_r \arrow[l] \arrow[r] \arrow[d] & \mathbb{U}_r \arrow[d] \\ \mathbb{V}_{r'} & Gk_{r'} \arrow[l] \arrow[r] & \mathbb{V}_{r'} \end{tikzcd} \quad \overset{H}{\Longrightarrow} \quad \overset{E_t}{\Longrightarrow} \quad \varepsilon_{t,r}^{r'} : \begin{tikzcd} E_t(\varphi_r, \psi_r) \arrow[d] \\ E_t(\upsilon_{r'}, \nu_{r'}) \end{tikzcd}$$

Interleaving Lemma A. Let $\mathbb{U}, \mathbb{V} \subseteq \mathbb{R}^\ell$ and $h : \mathbb{U} \to \mathbb{U}$, $k : \mathbb{V} \to \mathbb{V}$ with tame distance functions. Set $\varepsilon = \mathrm{Hdf}(Gh, Gk)$. If $a + \varepsilon \leq b \leq c \leq d - \varepsilon$, then

$$\begin{tikzcd} E_t(\varphi_a, \psi_a) \arrow[r] \arrow[d] & E_t(\varphi_d, \psi_d) \\ E_t(\upsilon_b, \nu_b) \arrow[r] & E_t(\upsilon_c, \nu_c) \arrow[u] \end{tikzcd}$$

commutes.

Proof



IV.3 CONVERGENCE

INFERENCE THM. Let $\mathbb{X} \subseteq \mathbb{R}^\ell$ be a compact absolute neighborhood retract, $S \subseteq \mathbb{X}$ finite, and $f : \mathbb{X} \to \mathbb{X}$ a map such that the distance functions for $\mathbb{X}$ and Gf are tame. Then any map $g : S \to S$ satisfies

$$\dim E_t(\phi) = \operatorname{rank} \epsilon_{t,\varepsilon}^{3\varepsilon},$$

for all $Hdf(Gf, Gg) < \varepsilon < \frac{1}{4} \min\{hfs(\mathbb{X}), hfs(Gf)\}$.

IV.4 Stability

Stability Thm. Let $U, V \subseteq \mathbb{R}^\ell$, and $h: U \to U$, $k: V \to V$ such that the associated distance functions are tame. Then

$$\mathrm{Bot}(\mathrm{Dgm}(\mathcal{E}_t), \mathrm{Dgm}(\mathcal{F}_t)) \leq \mathrm{Hdf}(Gh, Gk),$$

where $\mathcal{E}_t$ and $\mathcal{F}_t$ are eigenspace towers defined by h and by k.

THANK YOU