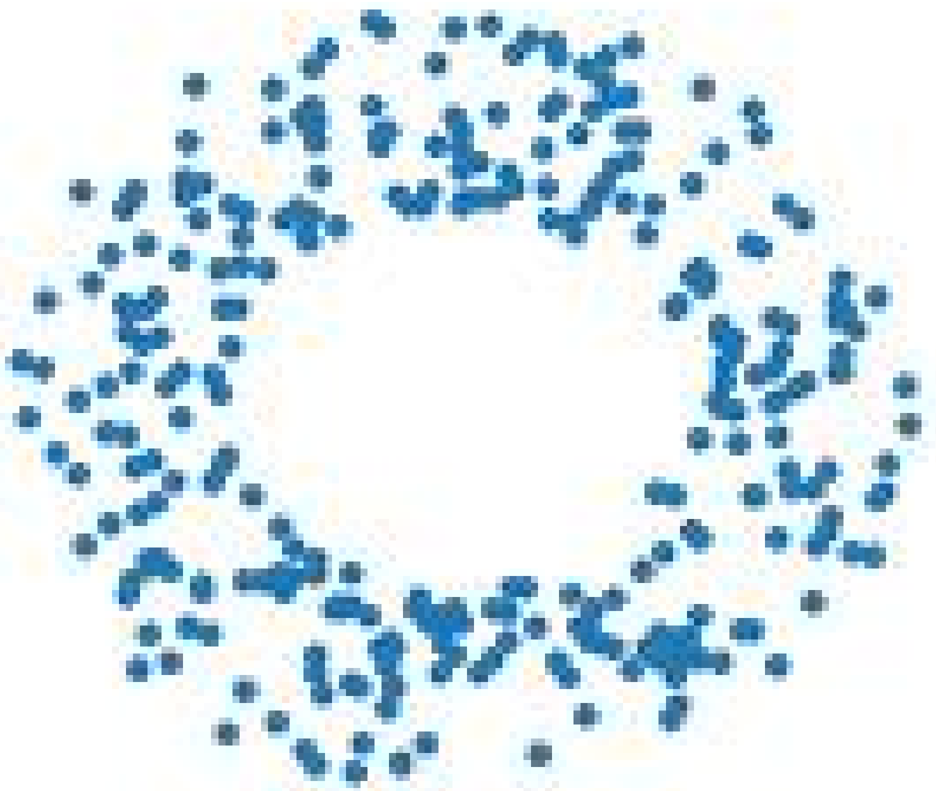
Introduction to Persistent Homology for Graph Analysis

The 5th International Conference on Econometrics and Statistics (EcoSta 2022)
EO385: Dependency in Network Data
6 June 2022
Shizuo Kaji (Kyushu Univ.)



How are these different?

Human is good at

- Rough estimation
- Panoramic view
- Discovering rules from a small number of examples
- Explaining why

AI is good at

- Precise observation
- Memorising/imitating examples
- Processing huge data
- Accurate operation

Deep Learning (DL)

Data-driven
local

Topological Data Analysis (TDA)

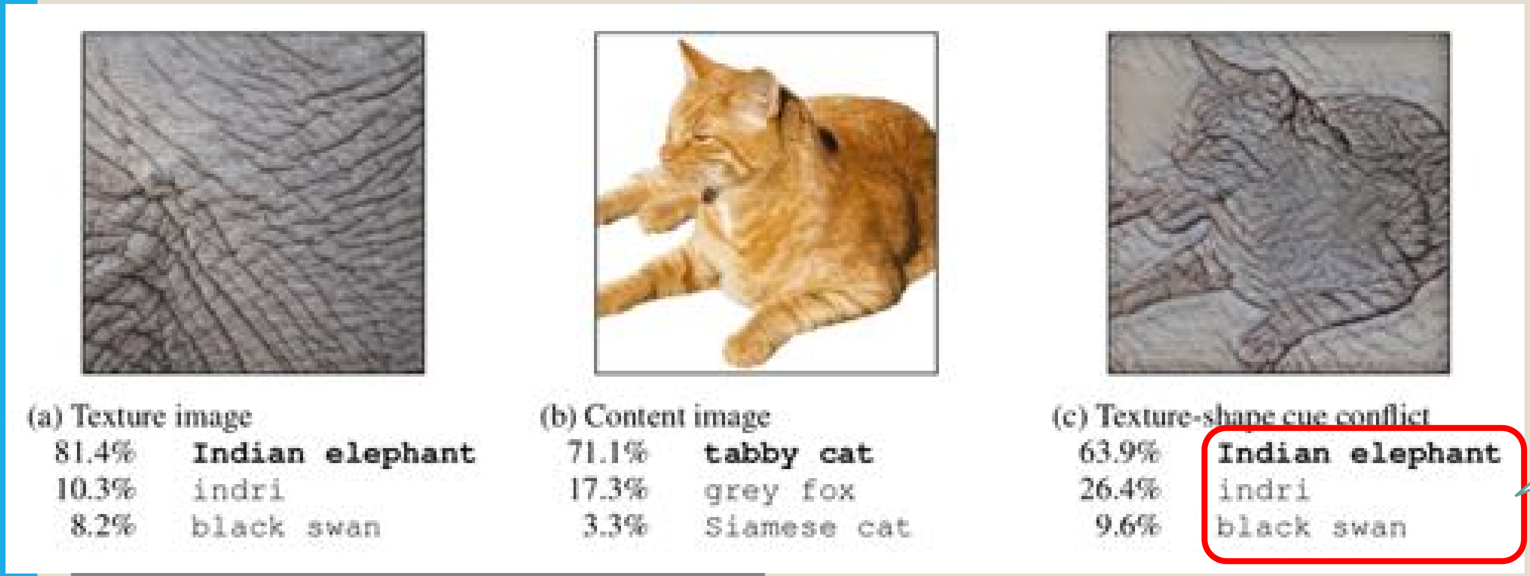
Maths-based
global

complementary

Background

- DL achieves high performance but has some weakness
- TDA has succeeded in capturing data features that conventional techniques have missed

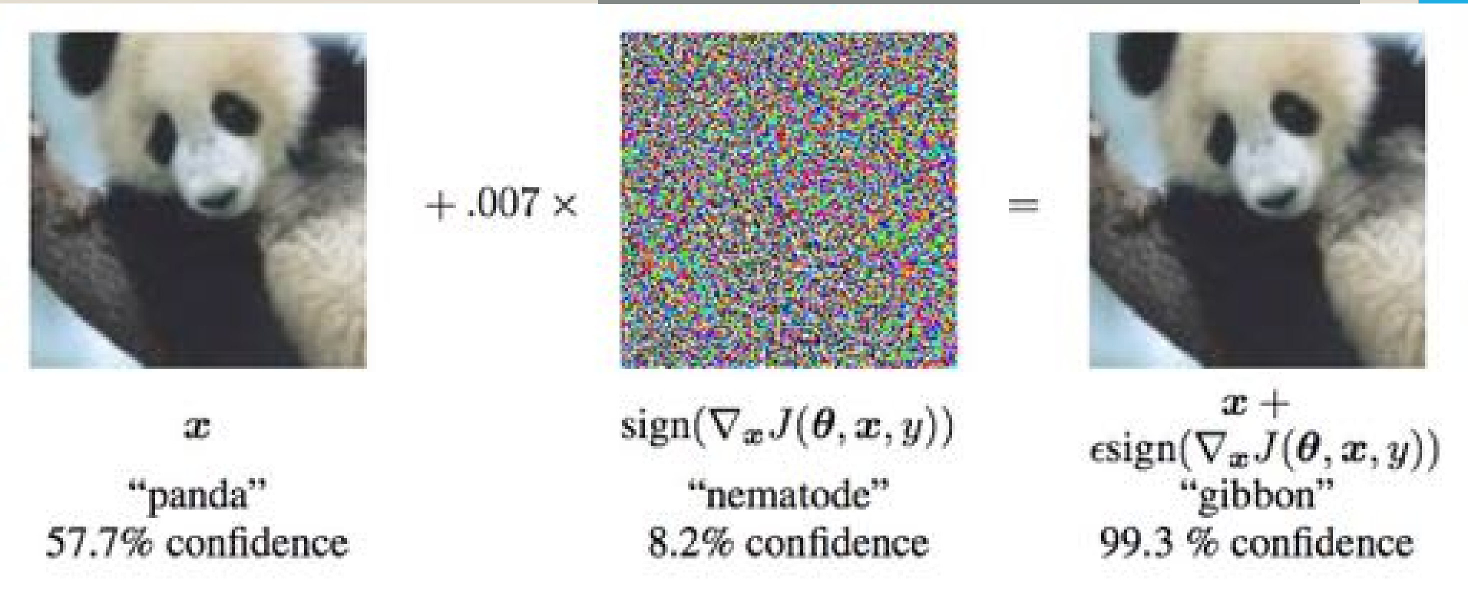
Convolutional Neural Nets are shortsighted



No attention is paid for the shape cue, but only the texture cue is considered by the model

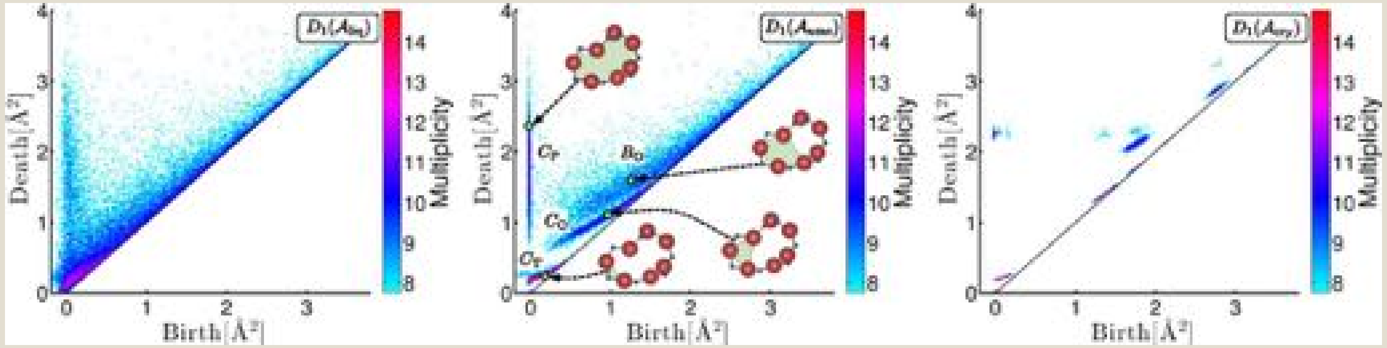
Geirhos et al. 2019

Goodfellow et al. 2014



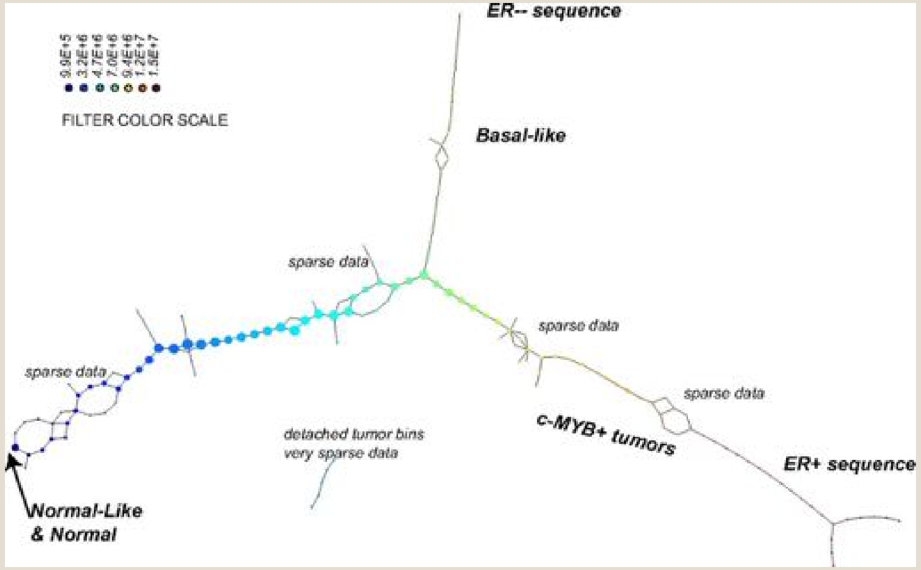
These results indicate CNNs are too sensitive to local features.

Notable applications of TDA



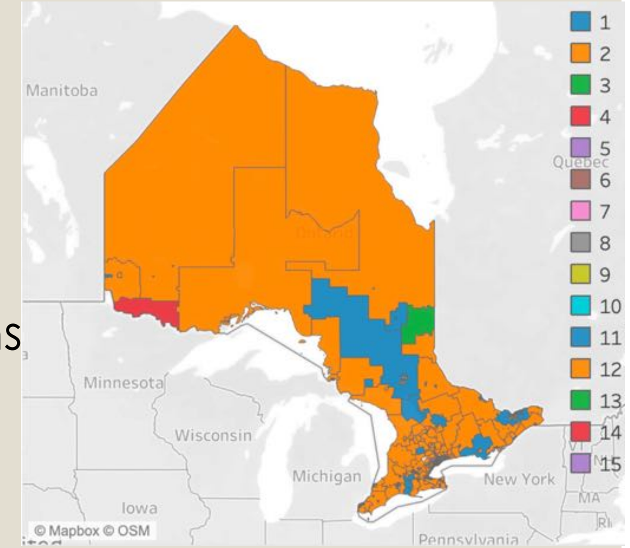
Liquid-Amorphous-Crystalline states of silica

Y. Hiraoka et al. PNAS 2016
Hierarchical structures of amorphous solids characterized by persistent homology



Gene expression data of cells

M. Nicholau et al. PNAS 2011
Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival



Clustering home insurance patterns

M. Yuvaraj et al. PNAS 2021
Topological clustering of multilayer networks

Q&A of TDA

TDA techniques

- Persistent homology
- Mapper
- Reeb graph
- TFDA
- Path signature
- Dimensionality reduction

Some common questions

- TDA = Persistent Homology?
- TDA = Point clouds?
- For what input data is TDA applicable?
- How to understand the output?
- Performance compared to conventional methods?
- Computational cost?
- How much data is necessary?
- How to incorporate TDA into an ML pipeline?

You can find my sample codes that provides a hands-on walkthrough on various aspects in TDA: Google

“Tutorial on Topological Data Analysis” kaji

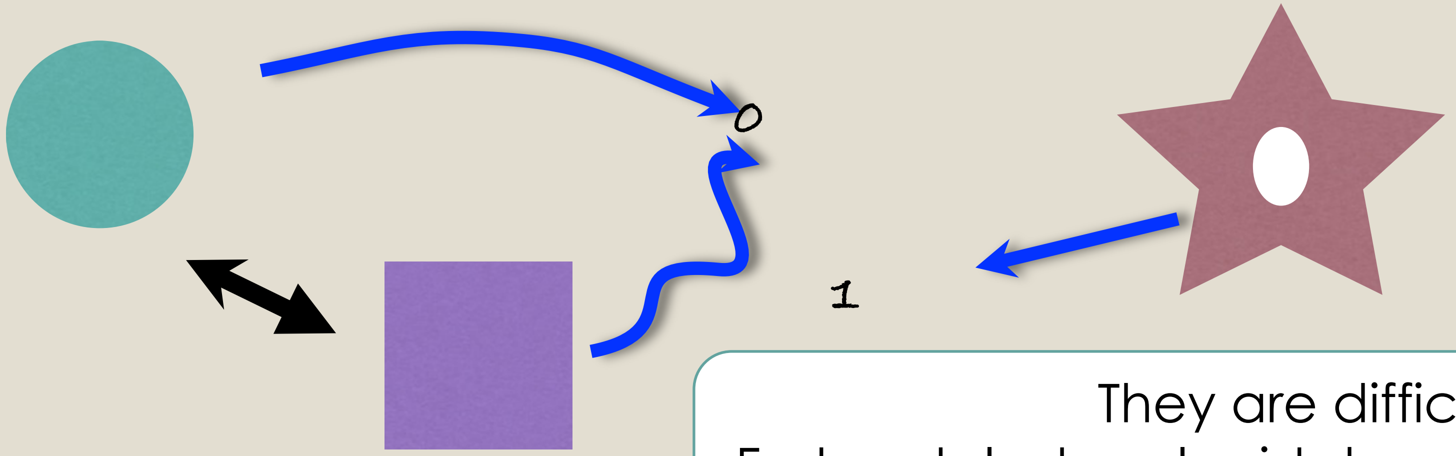
Topological Features

Data ➔ Topological space ➔ Topological invariants as features

Topological invariants

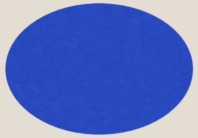
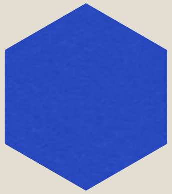
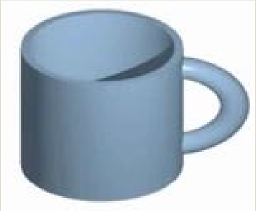
Easy to compare and manipulate

- Transform Spaces → "Numbers"
- Stay unchanged under continuous perturbation; capture **global characteristics**
- Are computable



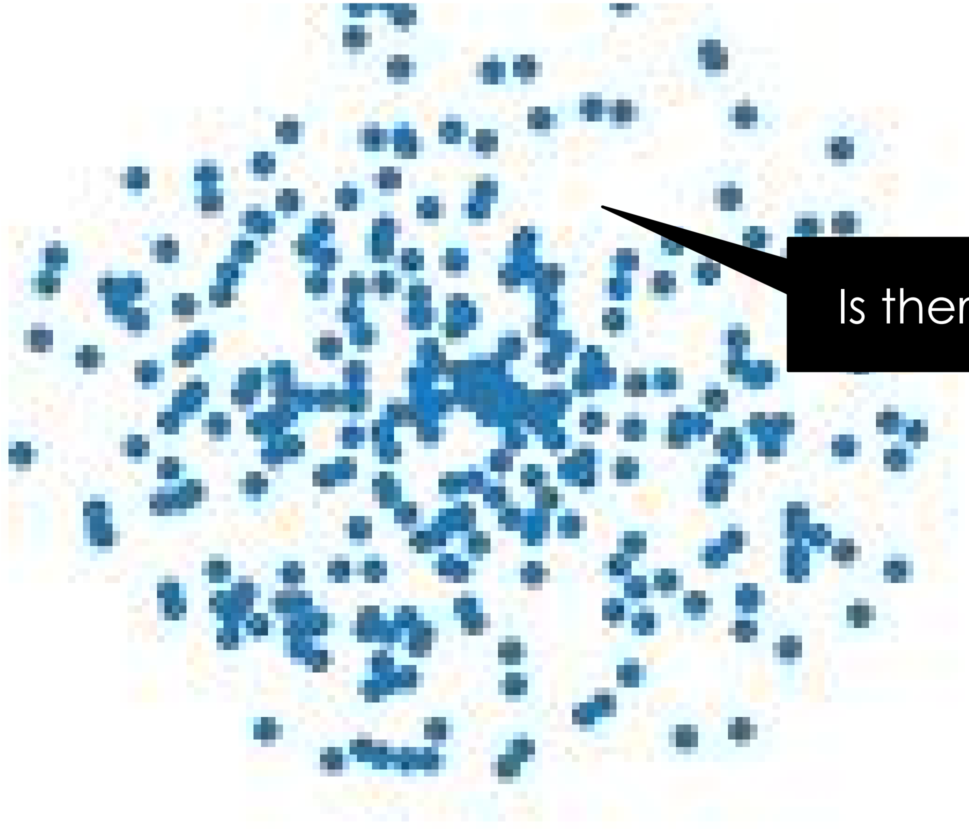
They are difficult to find.
Fortunately, topologists have worked for a long time to discover some nice topological invariants!

Example: Betti number (homology)

						
b_0 connected components	1	1	1	1	1	1
b_1 loop	0	0	1	2	2	0
b_2 cavity	0	0	0	1	1	1
Euler $b_0 - b_1 + b_2$	1	1	0	0	0	2

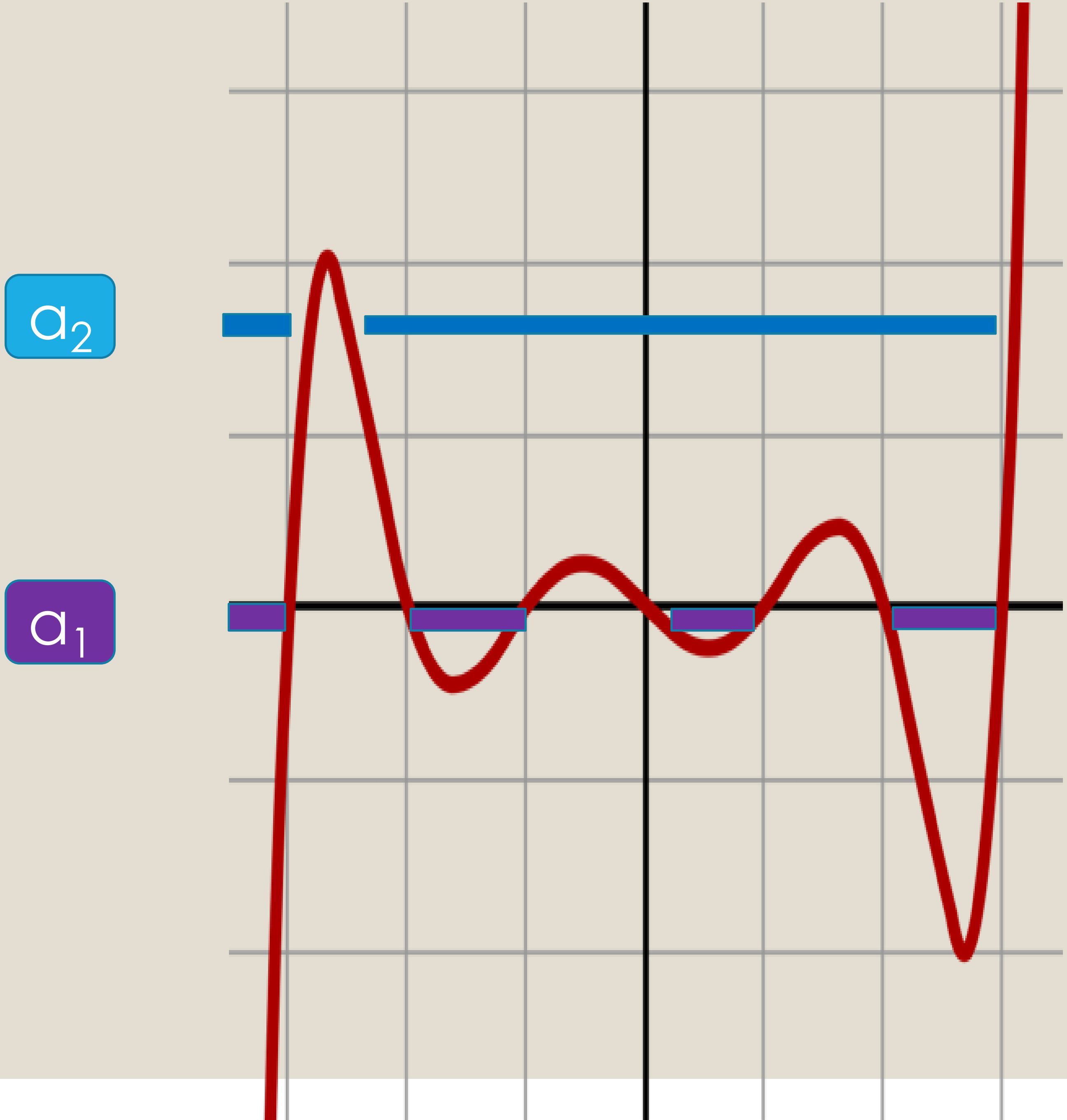
Space → Homology → Sequence of numbers

Wait, we do not have “spaces”



Is there a hole?

When we look at these samples, we unconsciously perform density estimation and thresholding to recover the “shape”

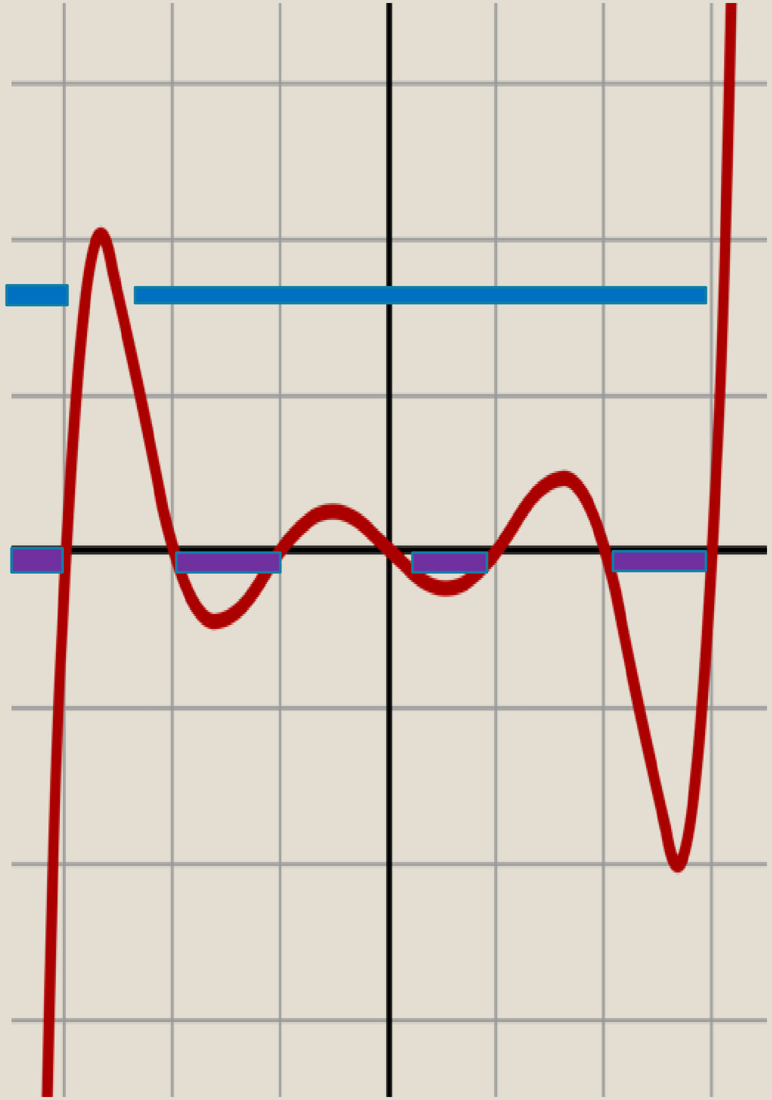


An easy example of PH_0

For each threshold a, we can count the number of segments in $\{x \mid f(x)<a\}$

Persistent homology (PH)

- Extension of homology defined for **functions** over topological spaces (e.g., images, weighted graphs)
- For each topological feature(cycle), the threshold values with which it was born and destroyed are recorded



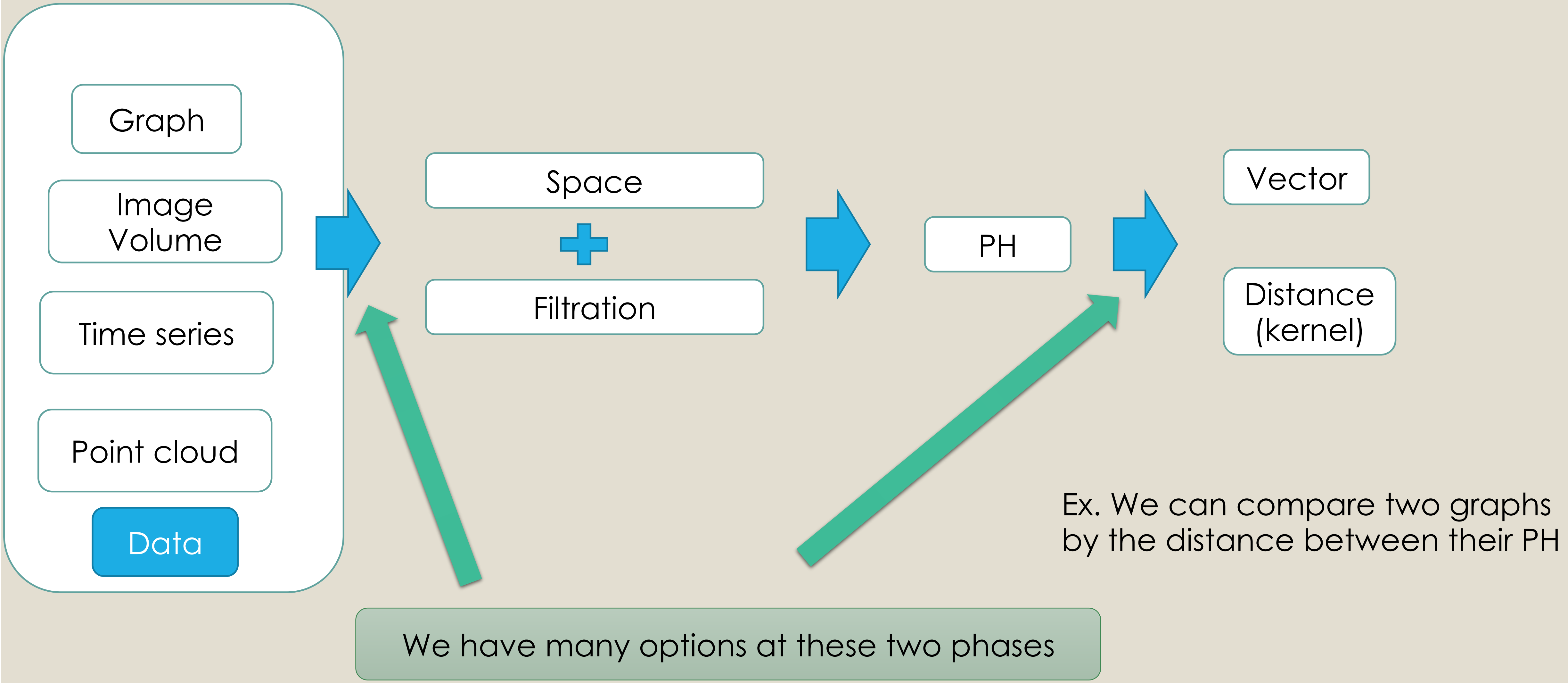
Space X
Function X→R

PH

Multi-set of points in R^2

Looks too exotic…
Luckily, we can turn it into a vector!

Data Flow

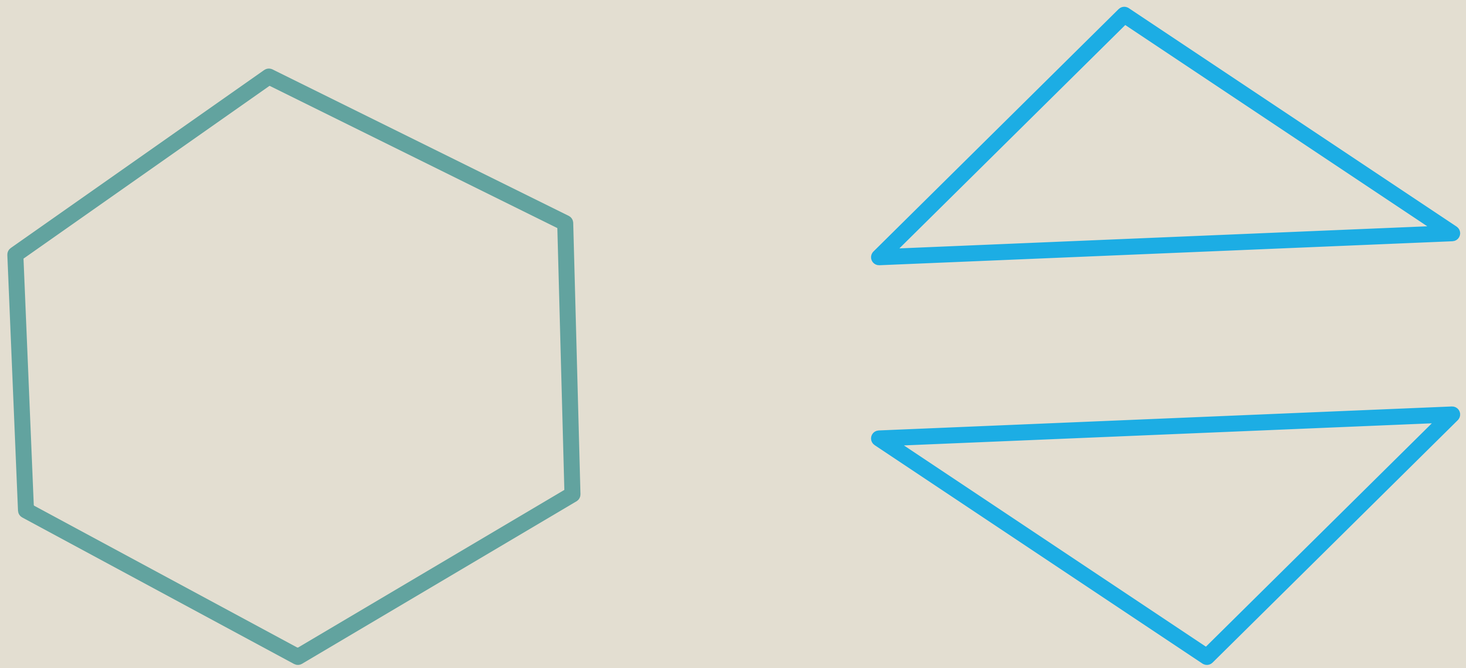

Graph
Image
Volume
Time series
Point cloud
Data
Space
Filtration
PH
Vector
Distance
(kernel)
Ex. We can compare two graphs
by the distance between their PH
We have many options at these two phases

PH Analysis of Graph

Graph Convolution vs Topology

If we look at 1-neighbour of each vertex, the discriminative power is limited to 1-Weisfeiler-Lehman test (1-WL test).
(K. Xu, W. Hu, J. Leskovec, S. Jegelka. How Powerful are Graph Neural Networks?, ICLR2019)

This results in a theoretical bound for message passing Graph Convolutional Networks.
For example, they cannot distinguish the following two graphs.

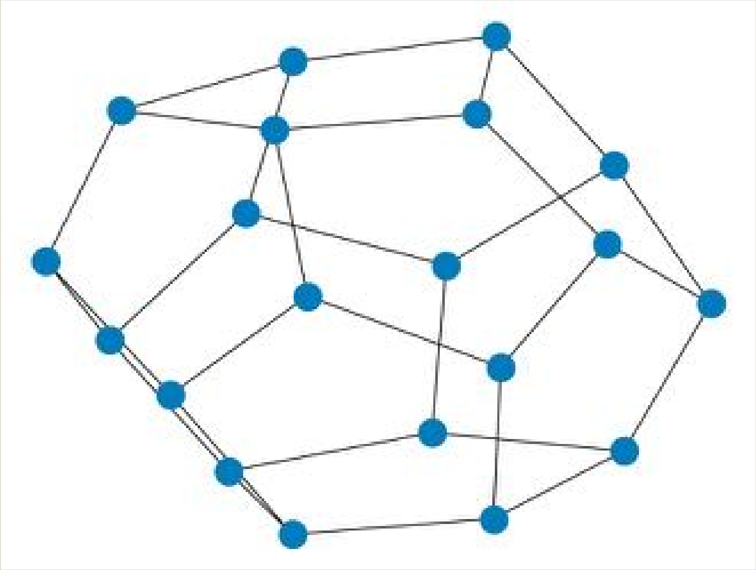


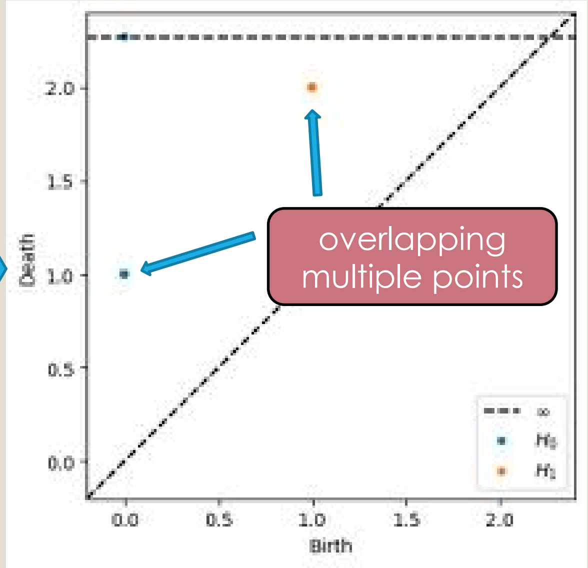
They have different topology and distinguishable by their homology.
Global features complements for local features!

Persistent homology of graph

Use case 1: A single feature for the whole graph

Use case 2: A feature for each vertex





A fixed size vector

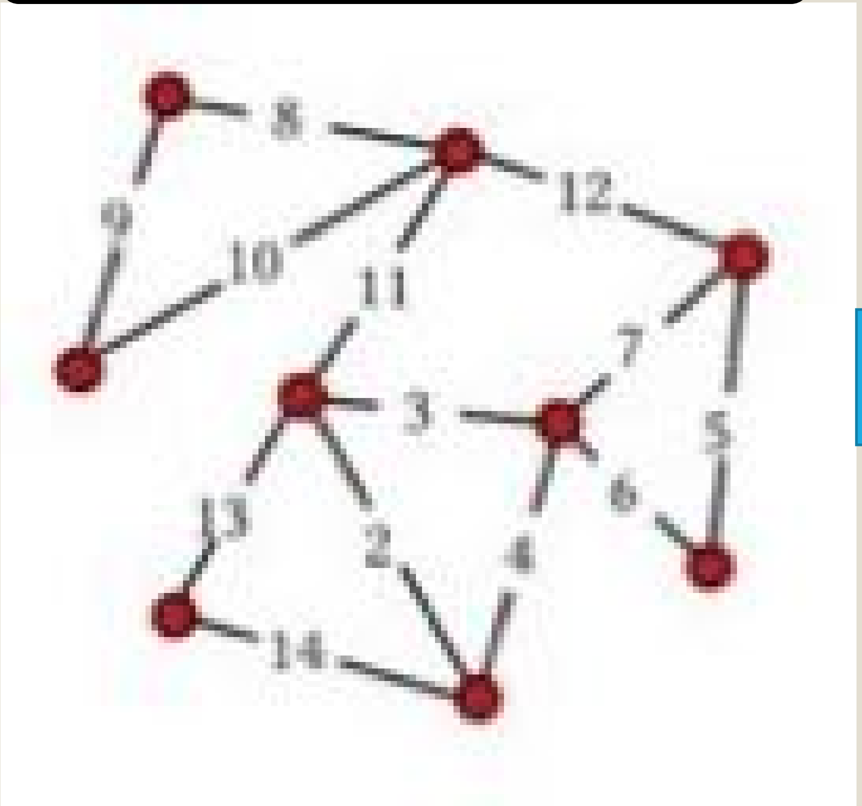
(directed, undirected, weighted)
Graph

Persistence diagram
(multi-set in R^2)

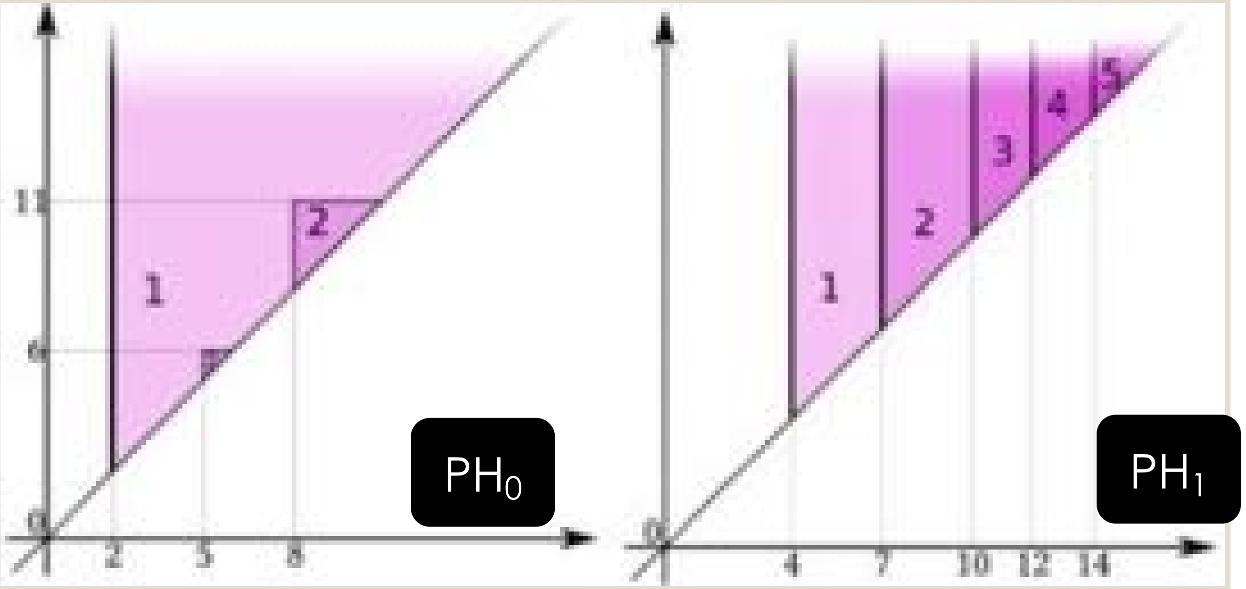
PH takes a graph and (vertex or edge) weights.
When weights are not present, we first have to define them in some way (discussed later)

Example

Input: Graph with edge weights

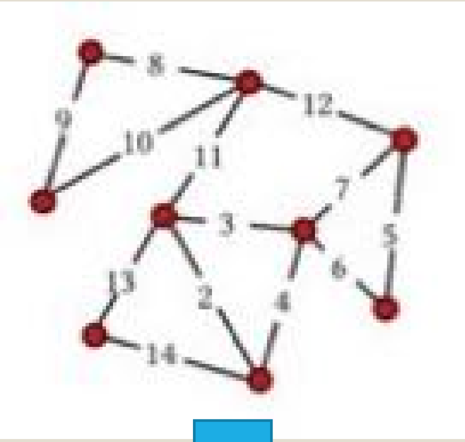


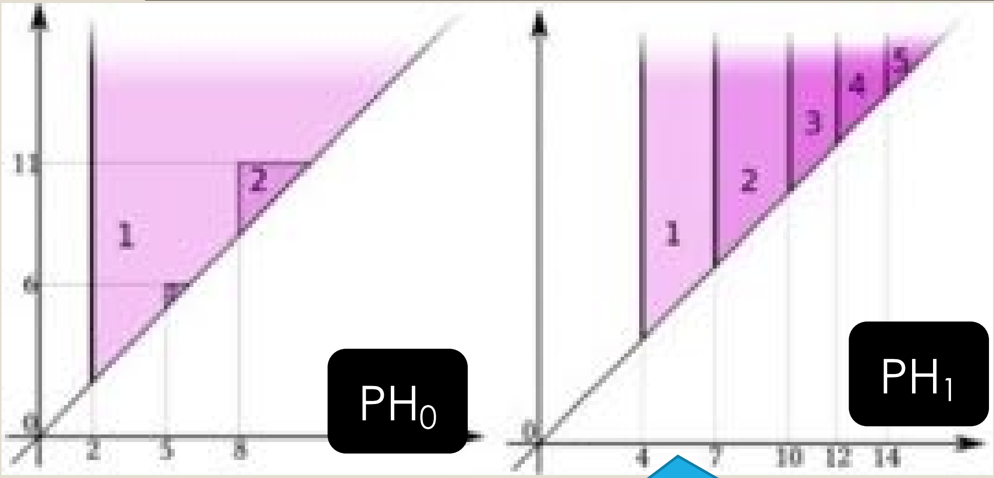
Output: Persistence diagrams

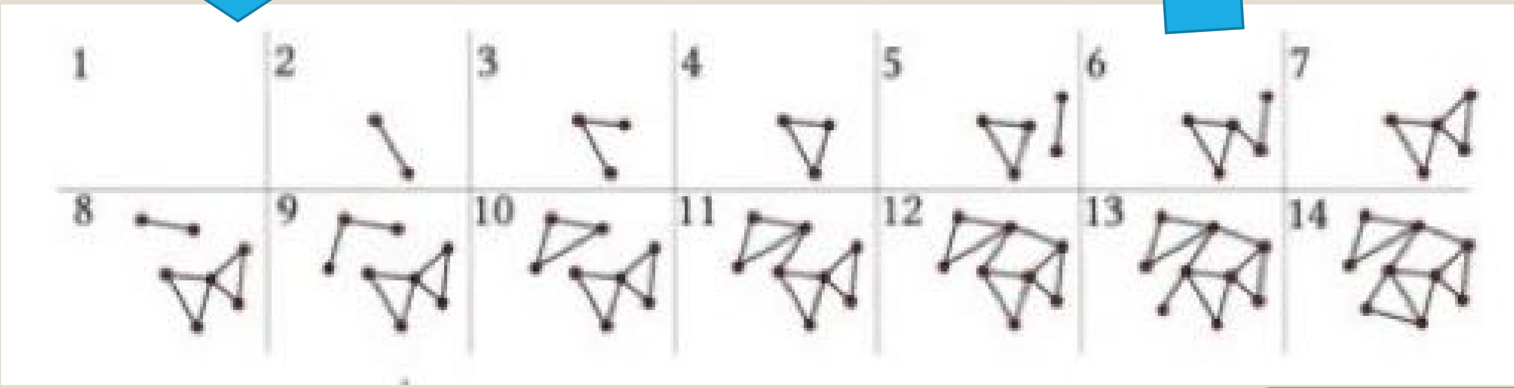


Bergomi MG, Ferri M, Vertechi P, Zuffi L. Beyond
Topological Persistence: Starting from Networks. 2021

Basic Idea: Threshold & Count & Connect







1. Thresholding is applied to obtain subgraphs by sweeping threshold values
2. Homology is computed for each subgraph considered as a 1-dimensional simplicial complex
3. “correspondence” between cycles are considered to identify persistent cycles.

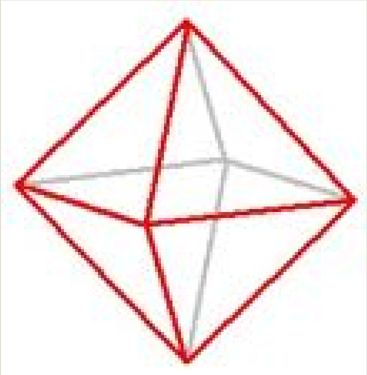
“Functoriality”

Bergomi MG, Ferri M, Vertechi P, Zuffi L. Beyond Topological Persistence: Starting from Networks. 2021

1&2 are just “threshold and compute features.”
What makes “persistence” special is 3.

Graph to space (complex)

- Option 1: Consider a graph as a 1-dimensional simplicial complex
  PH_0: connected components (clusters)
  PH_1: cycles
  no higher PH
- Option 2: (directed) flag complex: k-clique = (k-1)-simplex
  PH_0: connected components (clusters)
  PH_1: cycles of length > 3
  PH_2: cavity consisting of >4 vertices surrounded by triangles
- Option 3: Path complex (rarely used in applications)



Constructing Filtrations

Use vertex-dependent filtration to obtain vertex-wise features.

attributes → Filtration

1. No attributes => define vertex weights only from the combinatorial structure
   - Degree at vertices
   - Power filtration (= create a complete graph with edge weighted with path length)
   - WL-test
2. Graph with vertex weights (0-cochain)
   - edge weight is defined as the maximum of the vertex weights of its ends
3. Graph with edge weights (1-cochain)
   - vertex weight is defined as the minimum of the adjacent edges
4. Dynamic graph with only vertex/edge creation
   - Weight is defined by the creation time
   - e.g., SNS (user –(follow,like)-> user), Wikipedia (user –(edit)-> article), Twitter (user –(retweet)-> user),
5. Dynamic graph with vertex/edge creation and deletion
   - Zig-zag persistence

PH Vectorisation techniques

Persistence diagrams can be turned into fixed-size vectors.
The size of the vector as well as some hyper-parameters are specified by the user

- persistence image (JMLR2015)
- persistence landscape (JMLR2015)
- persistence curves (CVPRW2020)

There are also deep-learning based vectorisation techniques

- Deep learning with topological signatures (NIPS2017)

Vectorisation involves some hyper-parameter tuning.
Vectorised PH features can be used in a standard ML pipeline.

GNN + PH

PH is differentiable at almost all input

- Persistent Weisfeiler–Lehman Procedure (ICML2019)
  Use WL-test to define distance between nodes to compute PH
- Persistent Homology based Graph Convolution Network (ICCV2021)
  Use PH as feature extractor
  and the loss function (similarity measure between two node-weighted graphs)
- Topological Graph Neural Networks (ICLR2022)
  Use PH as a layer of GNN (note that PH is differentiable)

Hybrid of Graph Neural Network and Persistent homology would be a promising approach to local-global analysis of networks

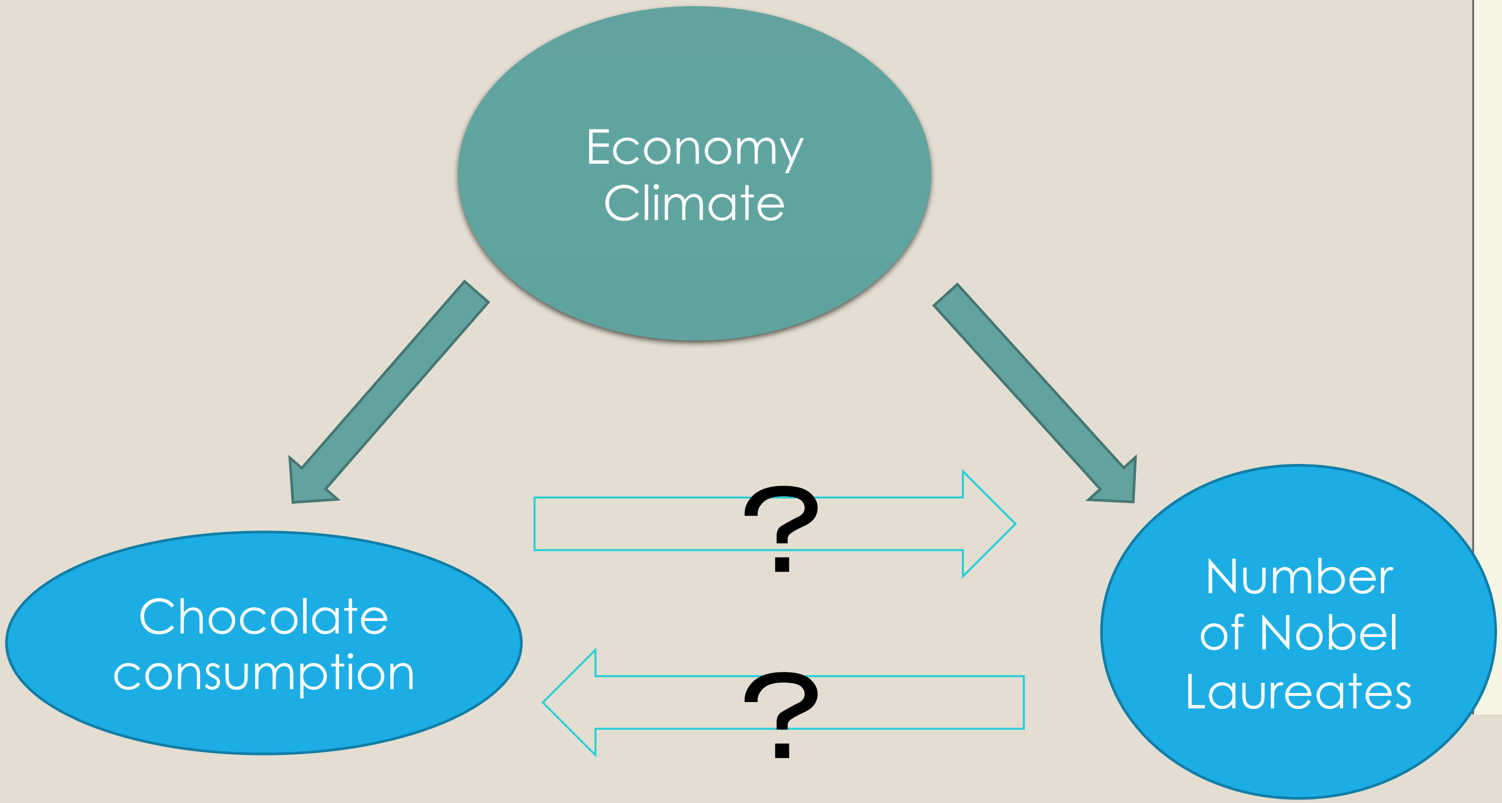
Software: PH for graphs

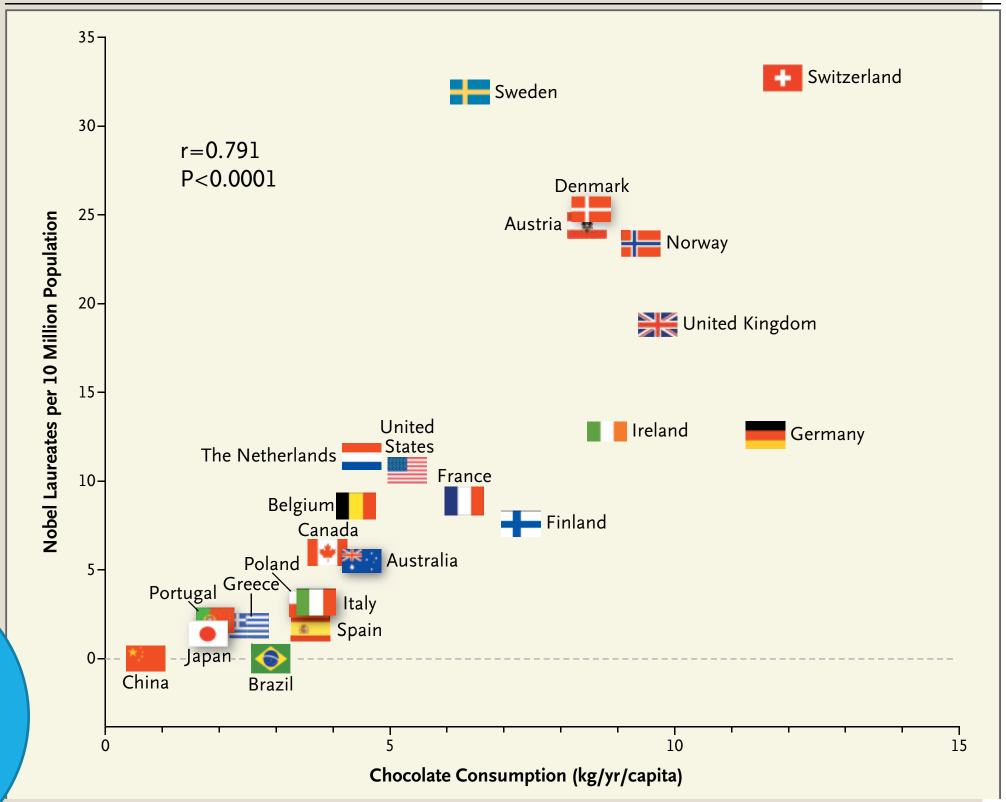
- **giotto-tda**
  a comprehensive Python package for persistent homology.
  It has a tutorial “Topological feature extraction from graphs”
- **A Persistent Weisfeiler–Lehman Procedure for Graph Classification**
  companion codes for the paper for graph classification
- **torchph**
  PyTorch extensions for persistent homology

PH for Identifying Graph Structure
-- Causality Detection --

Bando-K-Yaguchi (JSIAM Letters, 2022)
based on
Convergent Cross Mapping (CCM), Sugihara et al. Science, 2012

Causality in Network of variables





Messerli, F. H. (2012)

How to detect : Intervention (e.g., cut the supply of chocolate for 20 years!)

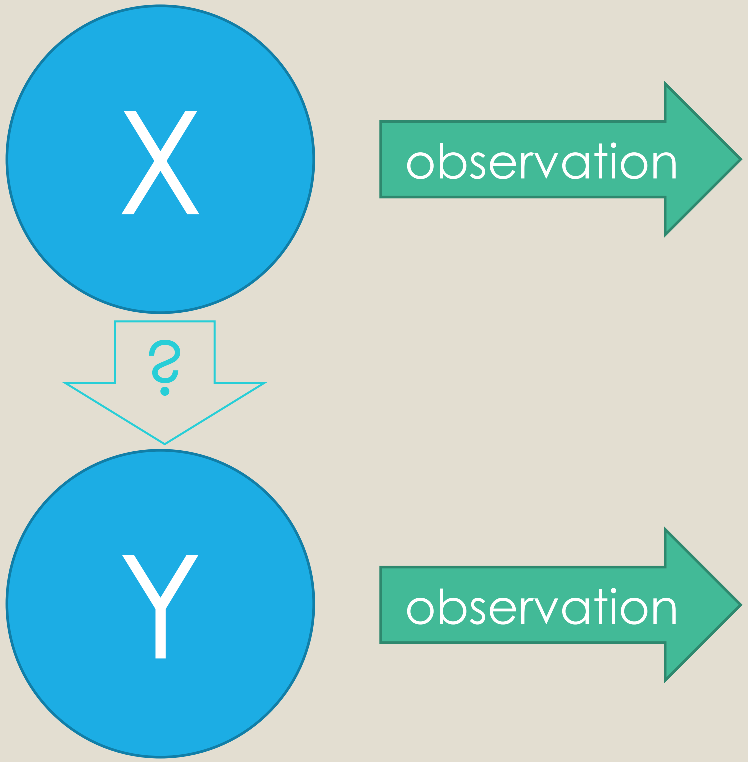
=> In many cases, it is not feasible

Goal today : Causal inference from observation (data)

⇒ Challenge: We cannot observe everything

Granger Causality

Nobel Laureate C. W. Granger (1969)



$x_1, x_2, \dots, x_t \dots$

$y_1, y_2, \dots, y_t \dots$

Compare two models

A $y_t = M(y_{<t})$

B $y_t = M(y_{<t}, x_{<t})$

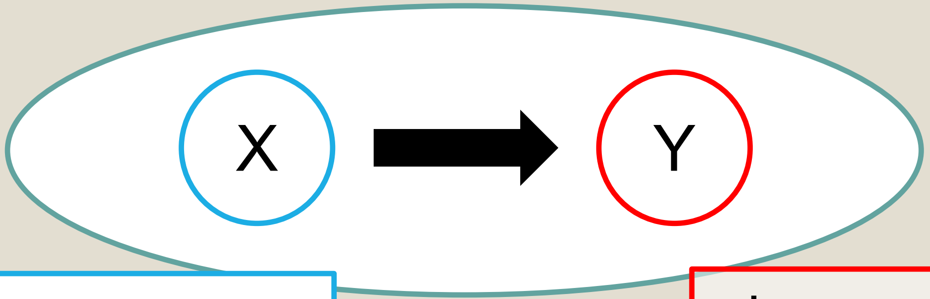
If Model B performs better than A
=>
<u>"X granger causes Y"</u>

Causality in dynamical systems

“X causes Y” = The evolution rule of Y is dependent on the state of X

total system $X \times Y$

$$\begin{cases} f: X \to X: & x_{t+1} = f(x_t) \\ g: X \times Y \to Y: & y_{t+1} = g(x_t, y_t) \end{cases}$$



upstream system X

$$x_{t+1} = f(x_t)$$

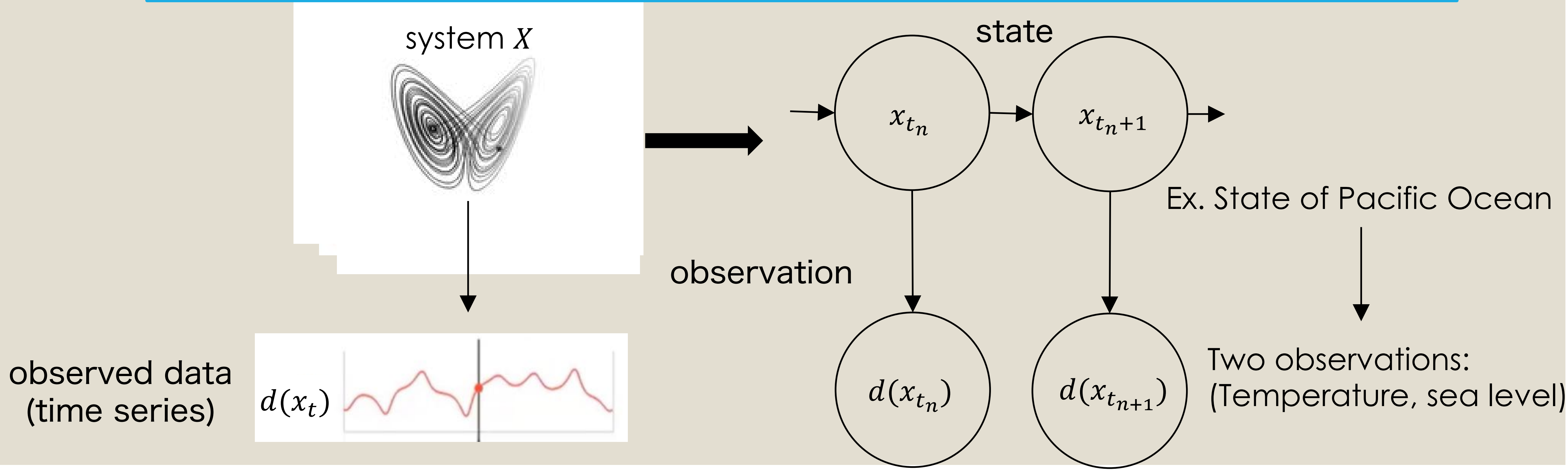
downstream system Y

$$y_{t+1} = g(x_t, y_t)$$

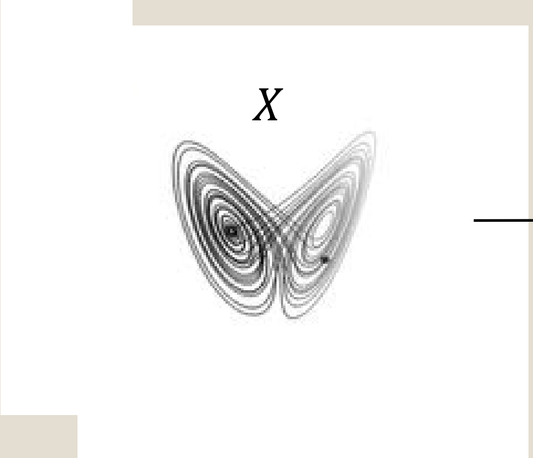
We have limited access to the system

- We have no way to gain complete knowledge of the state space
- Only partial information is available through observation

observation $d: X \to R^m$
(a map from the state space to a Euclidean space)

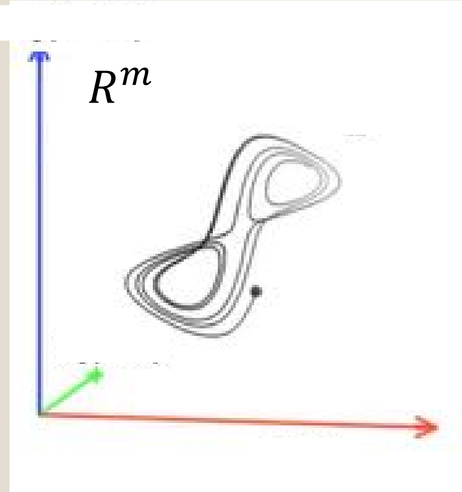


Reconstruction of the state space



random projection

$X \to R^m$



- For a large m, a generic observation $d: X \to R^m$ is known to be an embedding (more precisely, such d are dense in the space of smooth maps)

observation faithfully captures the state

But

- We cannot always make enough large observation.
  (intuitively, m is the number of simultaneous observations; e.g., temperature, humidity,⋯)

Delay-coordinate embedding (Takens' embedding)

Instead of large simultaneous observation,
we can (surprisingly!) reconstruct the state space
from a series of <u>**single observation**</u>

Define $\boldsymbol{d}: \mathrm{X} \to R^E$ by

$$\boldsymbol{d}(x) = (d(x_t), d(x_{t-1}), d(x_{t-2}), \dots, d(x_{t-E+1}))$$

E: embedding dimension

in practice, E is a hyper-parameter

<u>Theorem (Takens 1981)</u>

Under a mild assumption on the periodic orbits of the system,

for a large E and a generic d,

the image of the delay-coordinate embedding approximates the attractor of the system

Note: Delay-coordinate embedding provides a powerful tool for time-series analysis (not only for causality inference but also e.g., prediction).

Delay-coordinate embedding for a coupled system

Observation from the caused system can recover the total system

Total system $X \times Y$

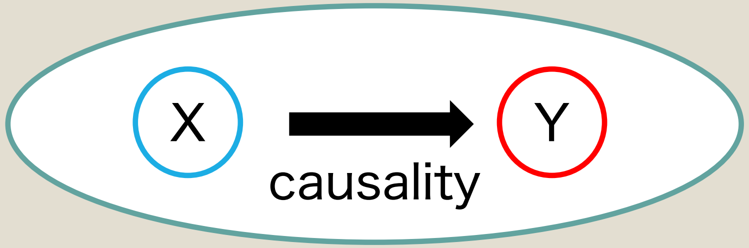
$$\begin{cases} x_{t+1} = f(x_t) \\ y_{t+1} = g(x_t, y_t) \end{cases}$$

upstream X

$$x_{t+1} = f(x_t)$$

observation

$$d_X\,(x_t)$$



downstream Y

$$y_{t+1} = g(x_t, y_t)$$

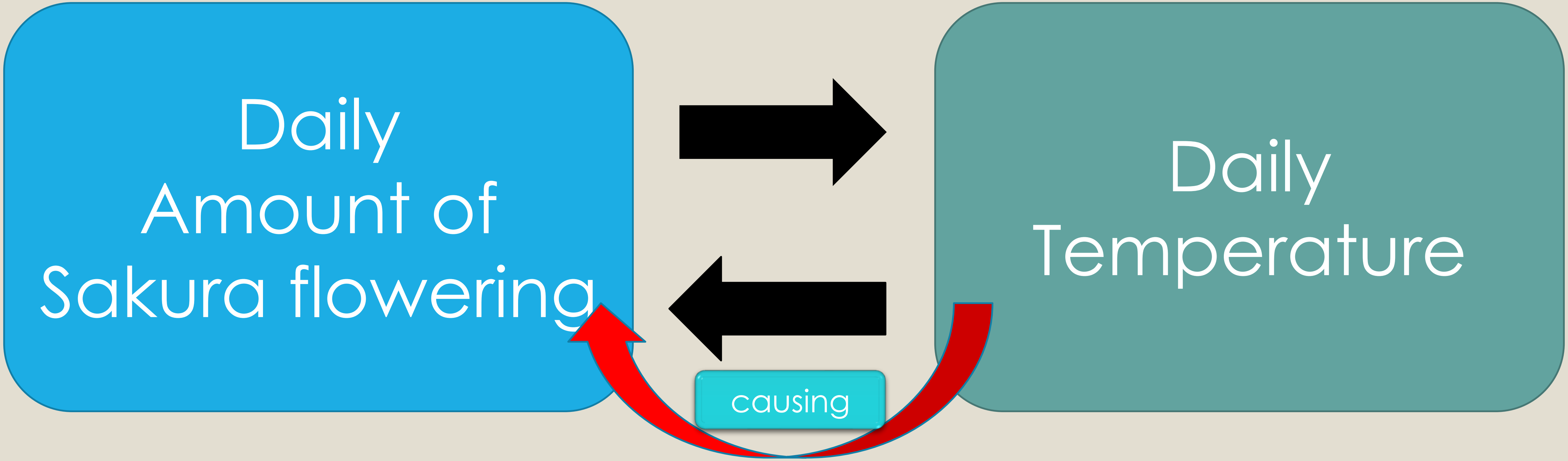
observation

$$d_Y\,(x_t)$$

Theorem (Stark 1999)

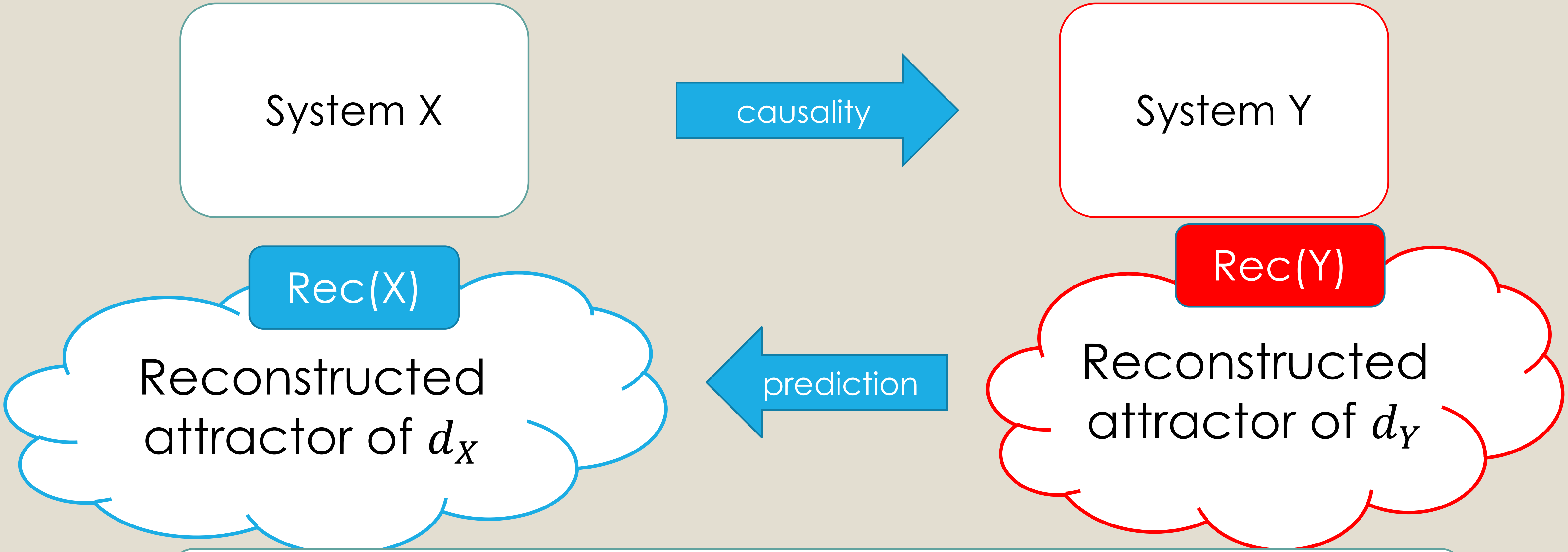
Under a mild assumption on the periodic orbits of f ,
the delay-coordinates of a generic observation $d_Y\,(x_t)$
approximates the attractor of the total system $X \times Y$

Down-stream system knows all



Which direction is easier to predict?

Topological consequence



There exists a continuous surjection (≒ prediction) from Rec(Y) to Rec(X).

Note: This means, Granger causality is not appropriate for a deterministic system

Causality detection by PH (Proof-of-Concept)

Applicability to real-world data is yet to be investigated.

Idea:
Time-series for each vertex
=> Spaces for each vertex and edge
=> comparing PH for the spaces