Similarities and Representations of Graphs

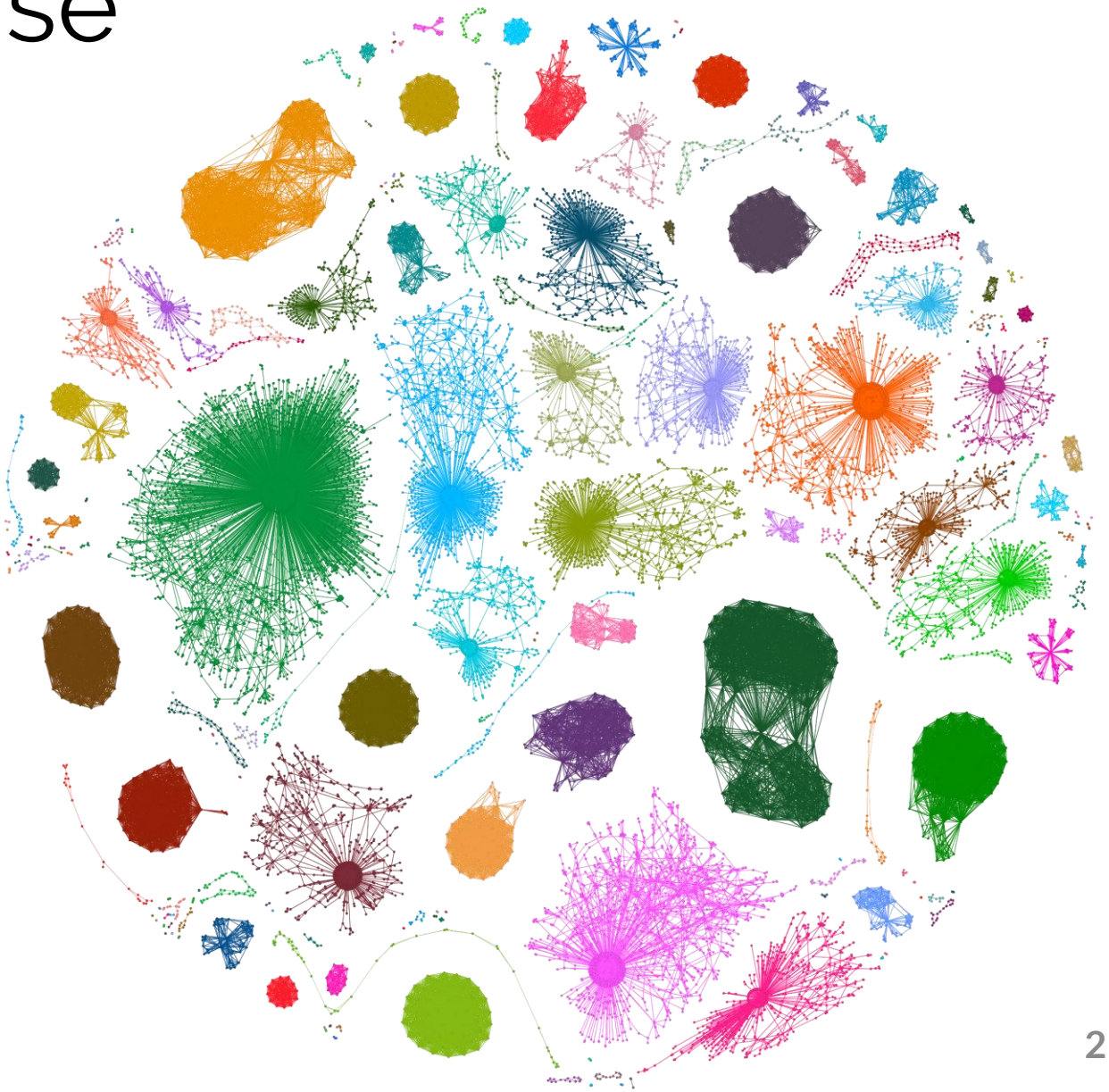
Anton Tsitsulin

Hasso Plattner Institute → University of Bonn

Graph world is diverse

Different **domains**:

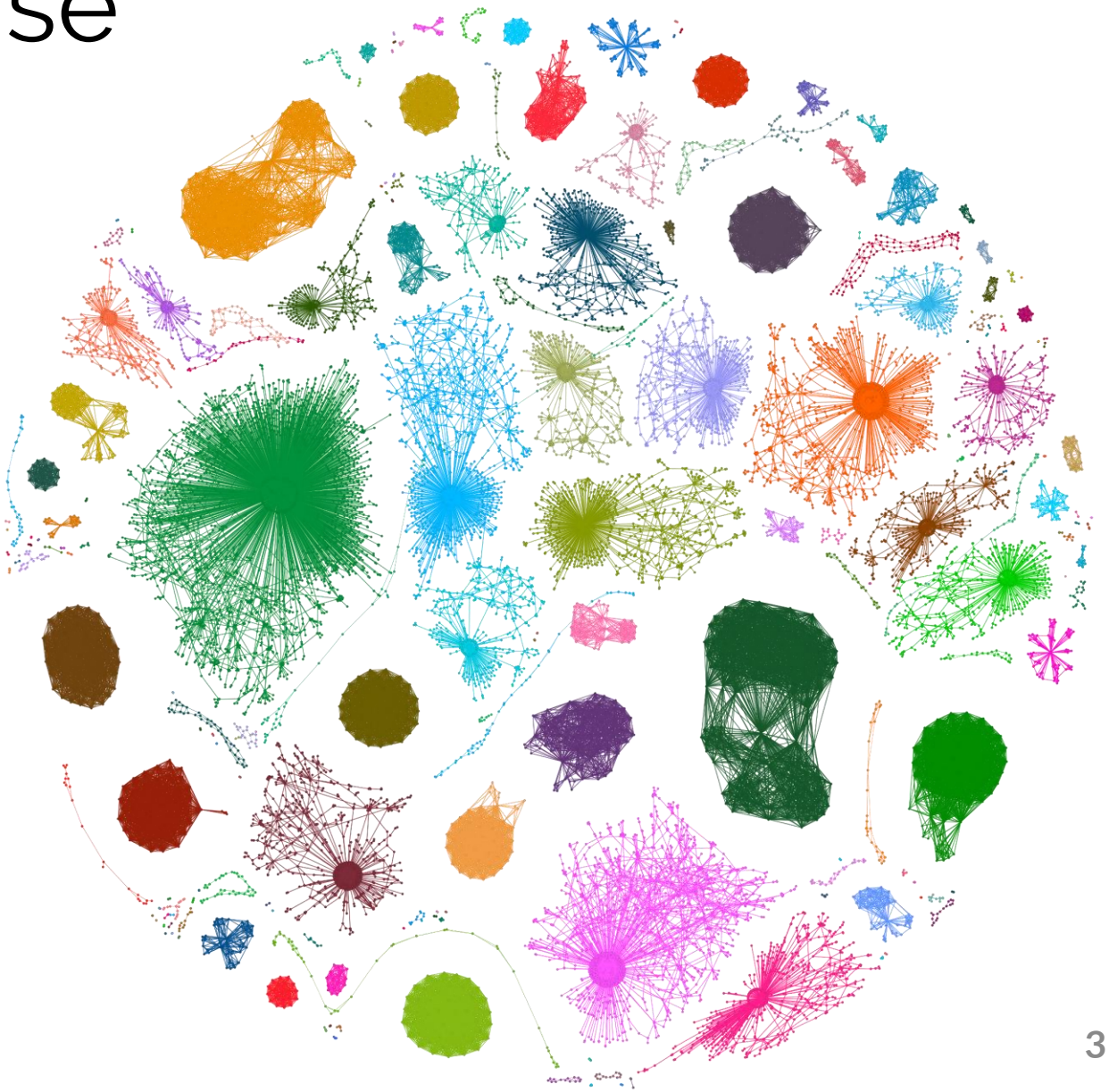
- Information
- Social
- Biological
- Transportation
- …



Graph world is diverse

Different **graph types**:

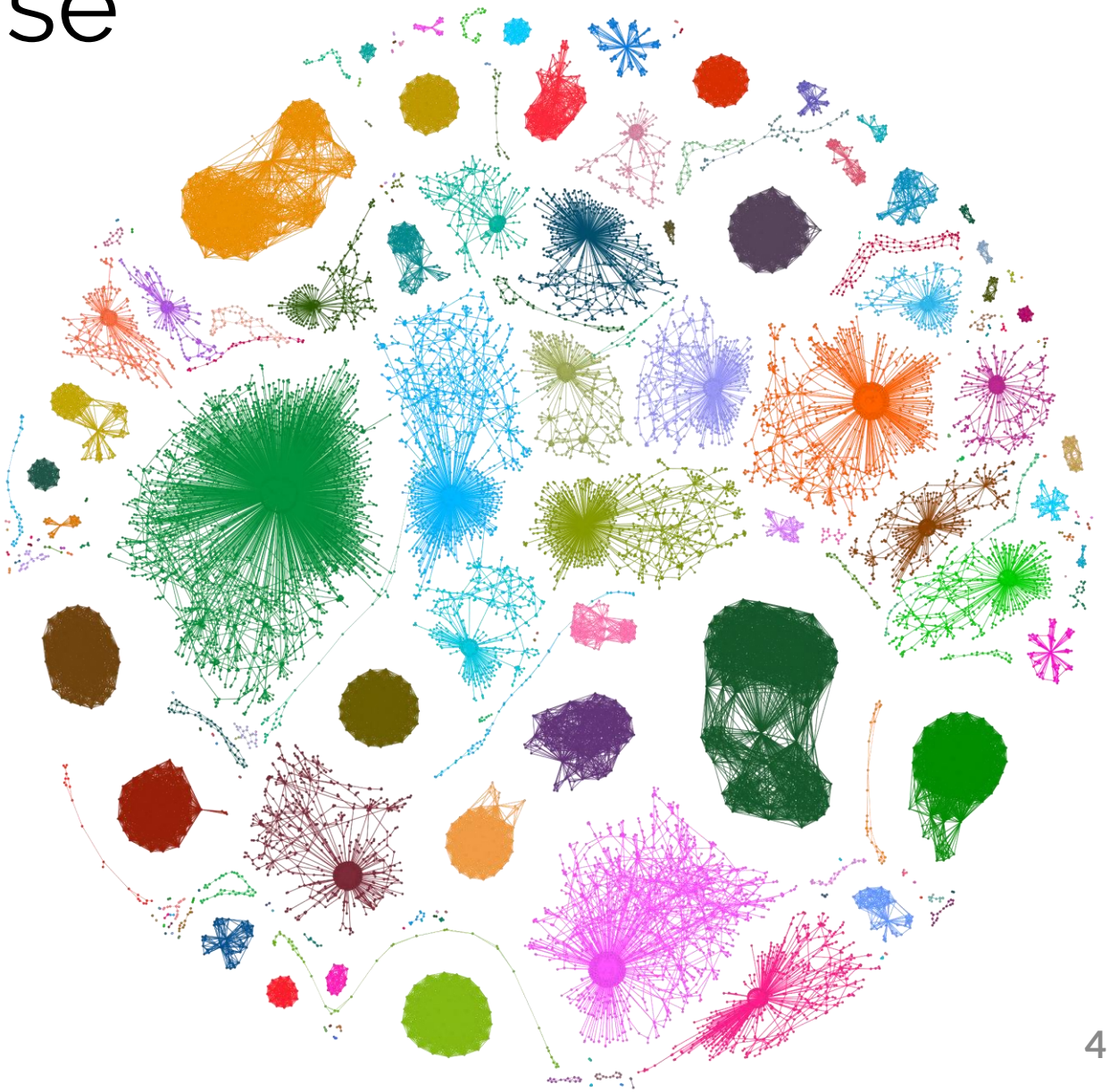
- (Un)directed
- (Un)weighted
- Temporal
- Heterogeneous
- …



Graph world is diverse

Different **modalities**:

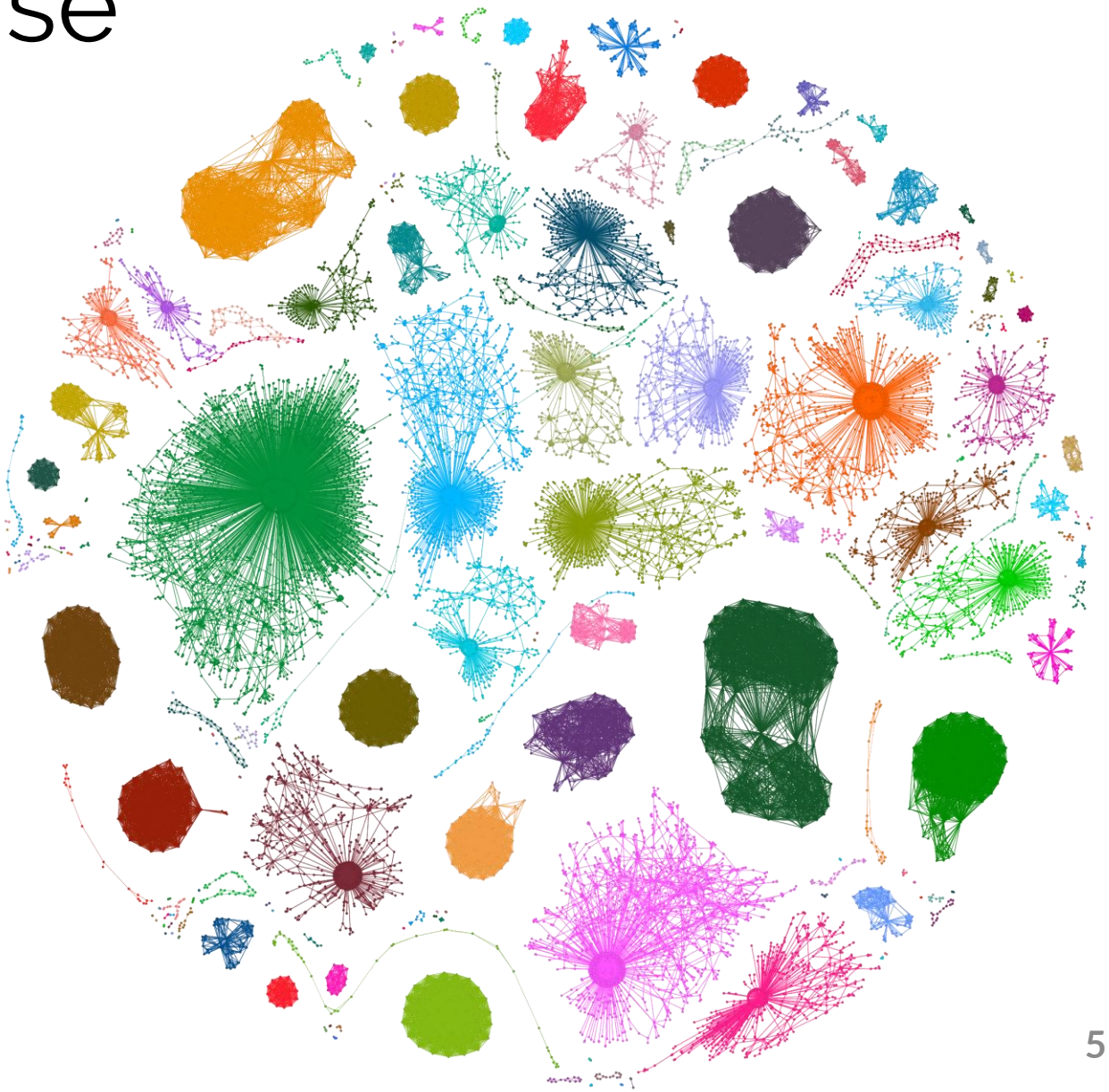
- Nodes
- Edges
- Motifs
- Subgraphs
- Whole graphs
- …



Graph world is diverse

Different **tasks**:

- Classification
- Clustering
- Anomaly detection
- …



Graph world is diverse

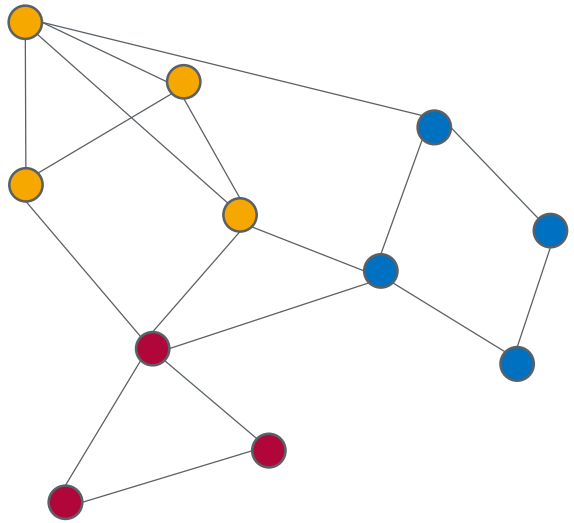
Domains	Graph Types	Modalities	Tasks
Information Social Biological Transportation	(Un)directed (Un)weighted Temporal Heterogeneous	Nodes Edges Subgraphs Whole graphs	Classification Clustering Anomaly detection

Graph world is diverse

Domains	Graph Types	Modalities	Tasks
Information Social Biological Transportation	(Un)directed (Un)weighted Temporal Heterogeneous	Nodes Edges Subgraphs Whole graphs	Classification Clustering Anomaly detection ↓ Embeddings

Why representations?

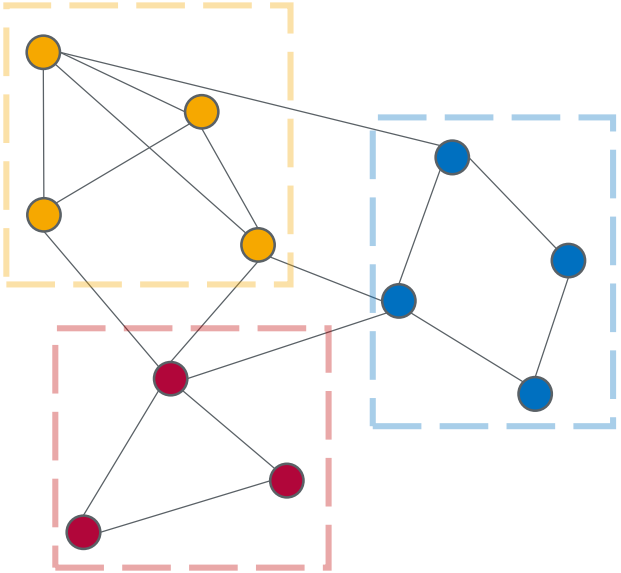
We have fast & good algorithms for mining vector data…



Clustering

Why representations?

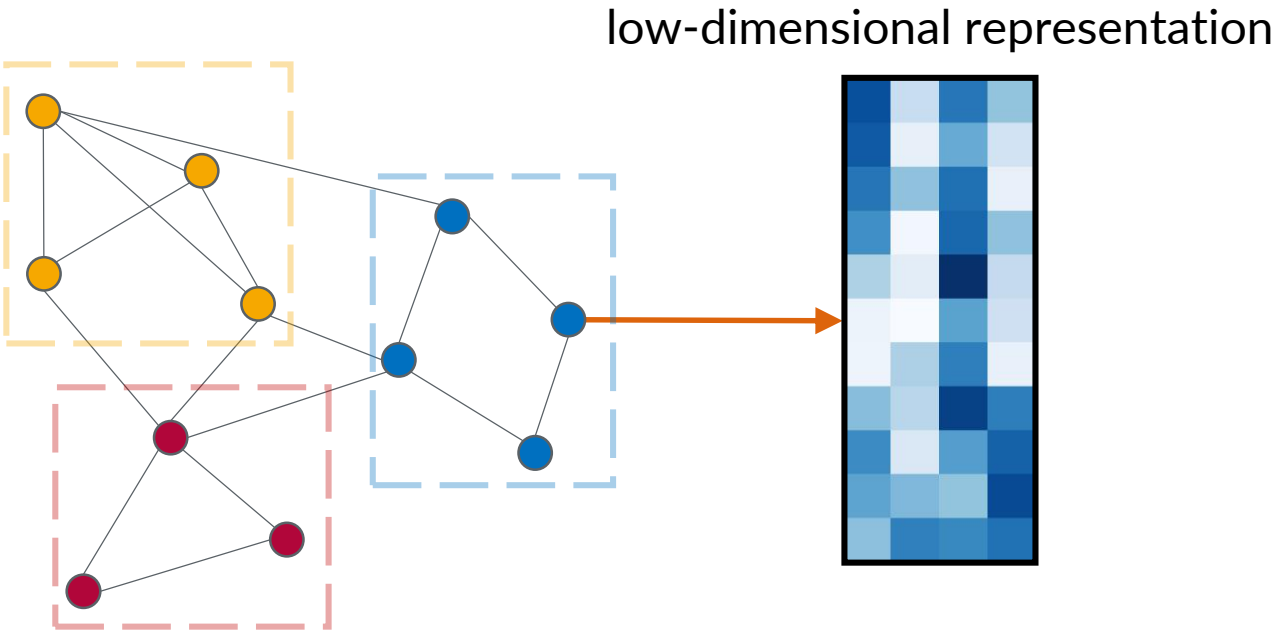
We have fast & good algorithms for mining vector data…



Clustering

Why representations?

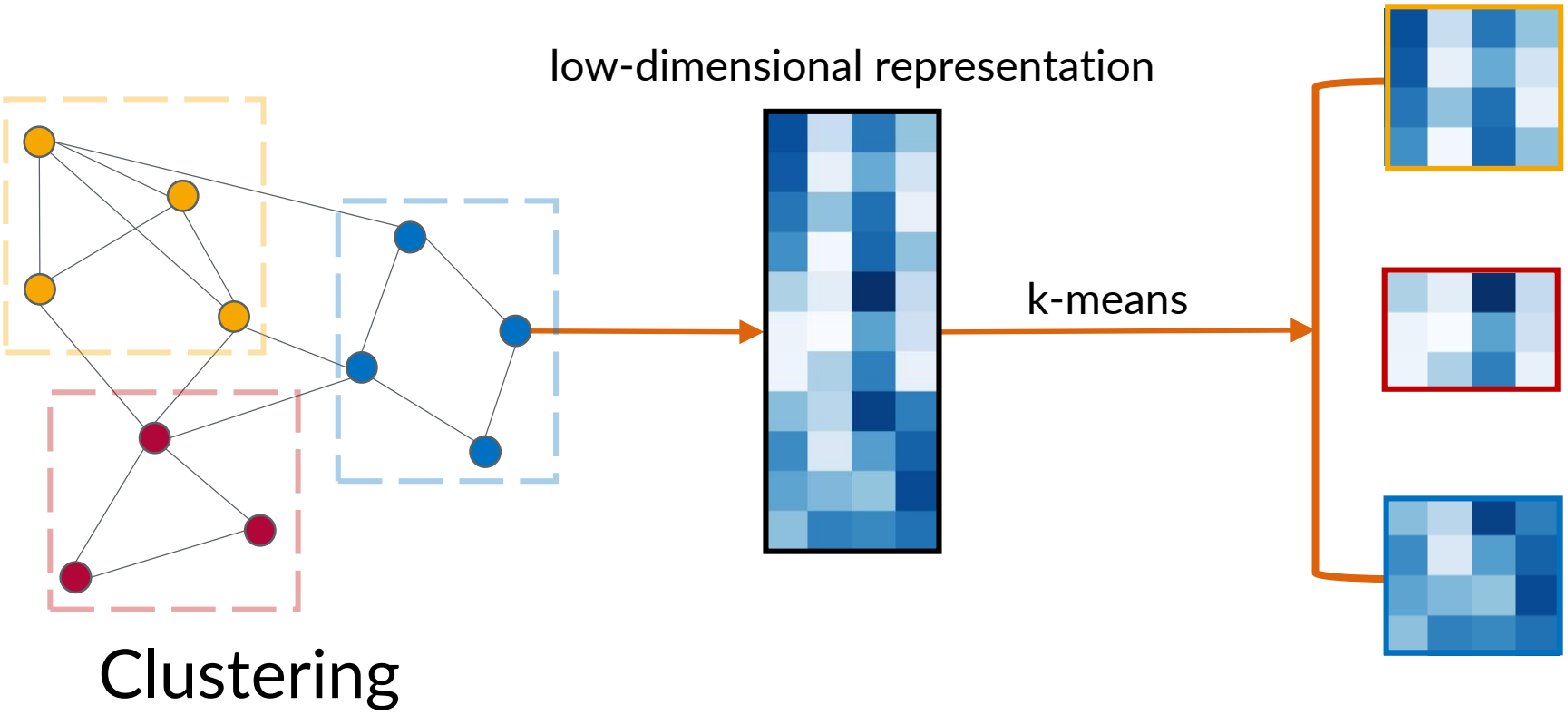
We have fast & good algorithms for mining vector data…



Clustering

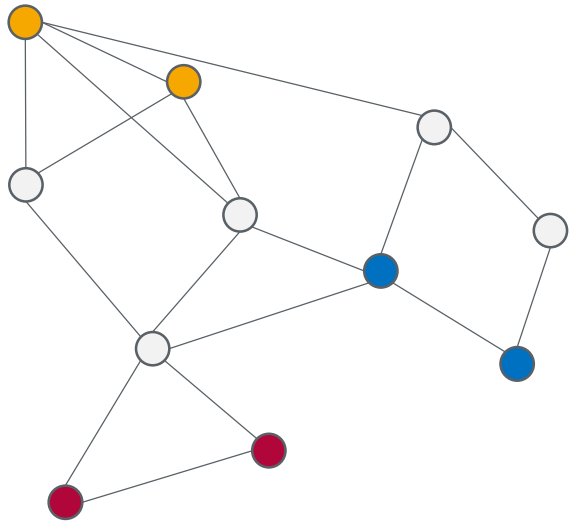
Why representations?

We have fast & good algorithms for mining vector data…



Why representations?

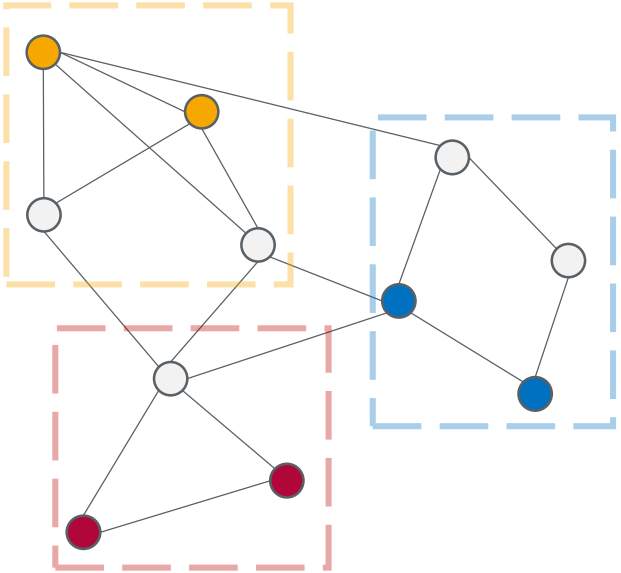
We have fast & good algorithms for mining vector data…



Classification

Why representations?

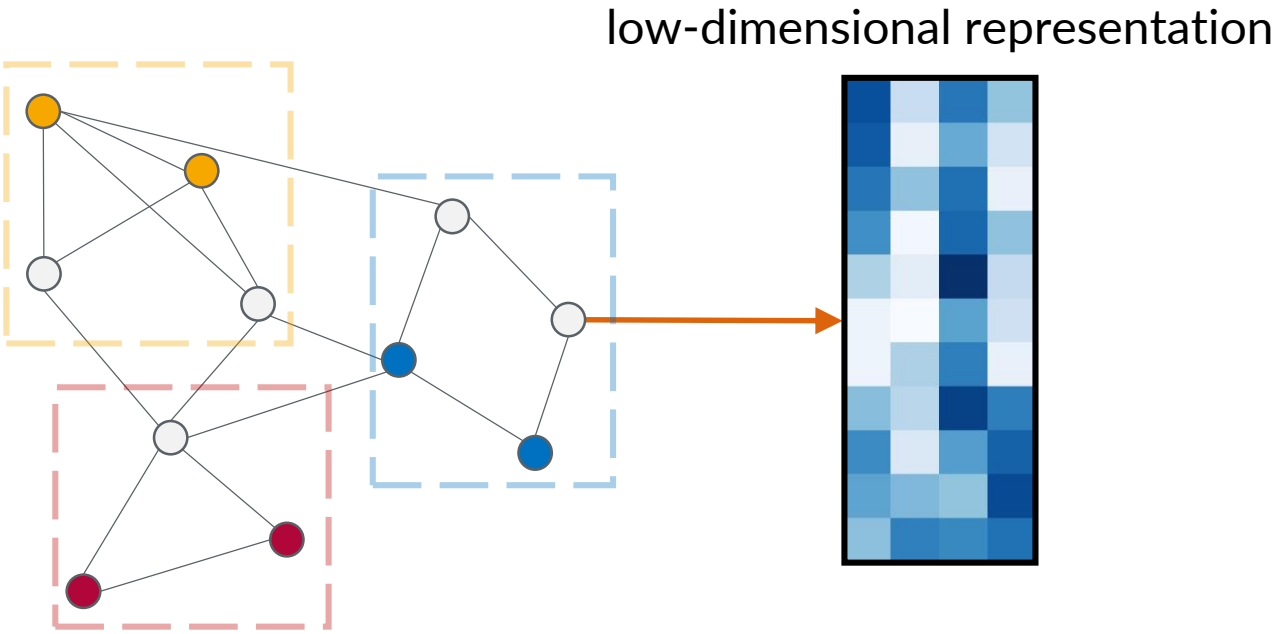
We have fast & good algorithms for mining vector data…



Classification

Why representations?

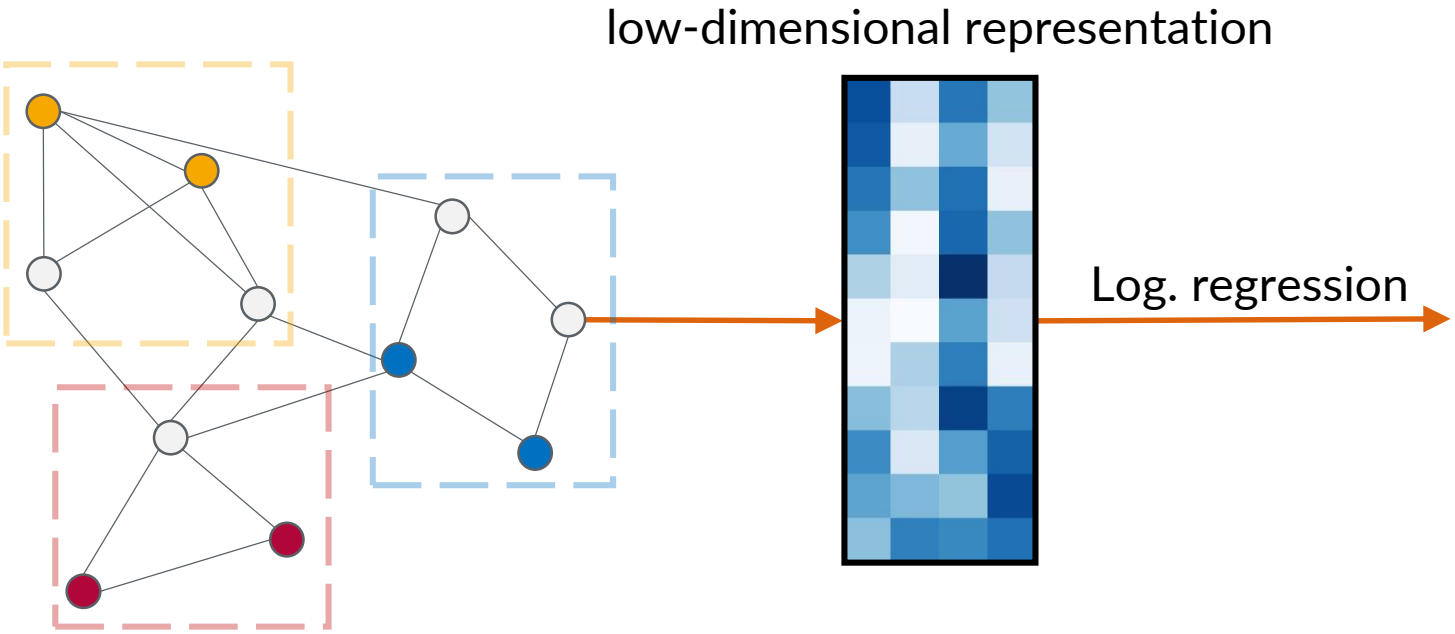
We have fast & good algorithms for mining vector data…



Classification

Why representations?

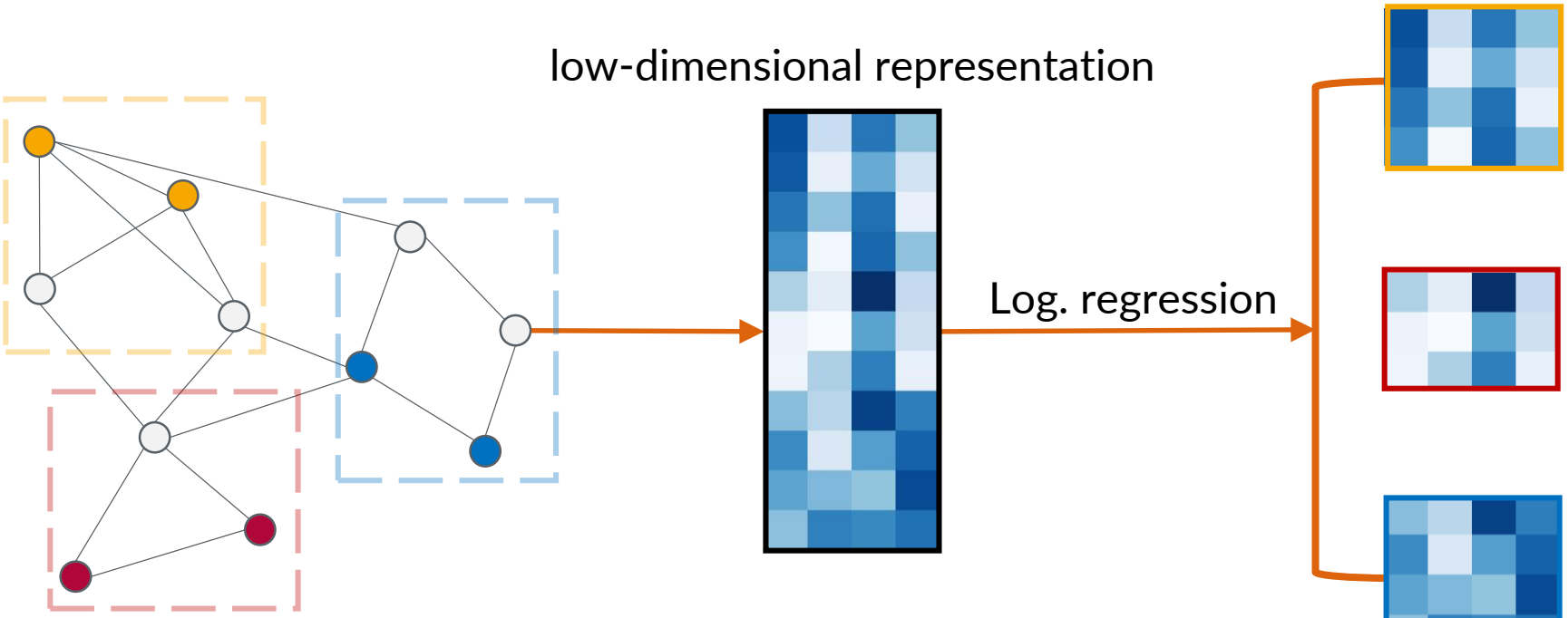
We have fast & good algorithms for mining vector data…



Classification

Why representations?

We have fast & good algorithms for mining vector data…



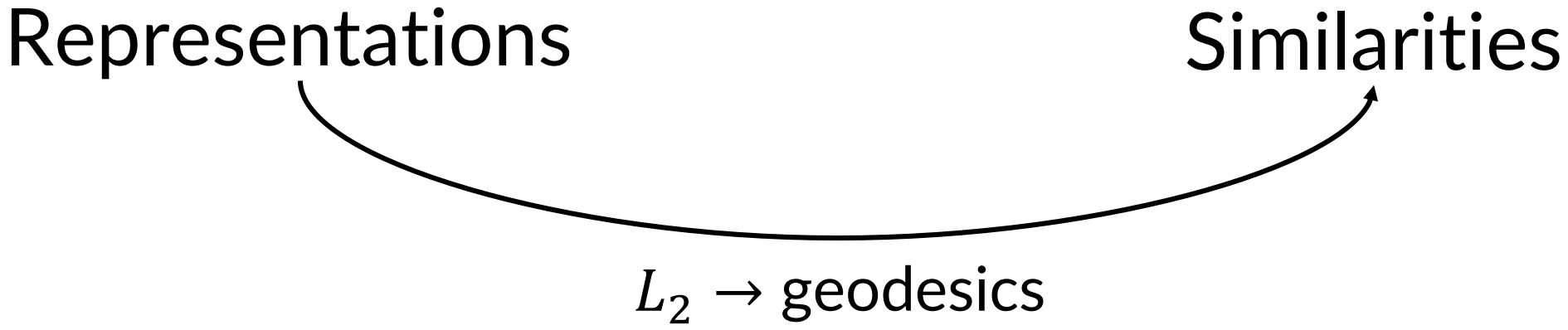
Classification

What makes a Representation?

Good representation **preserves geometry** of the original space.

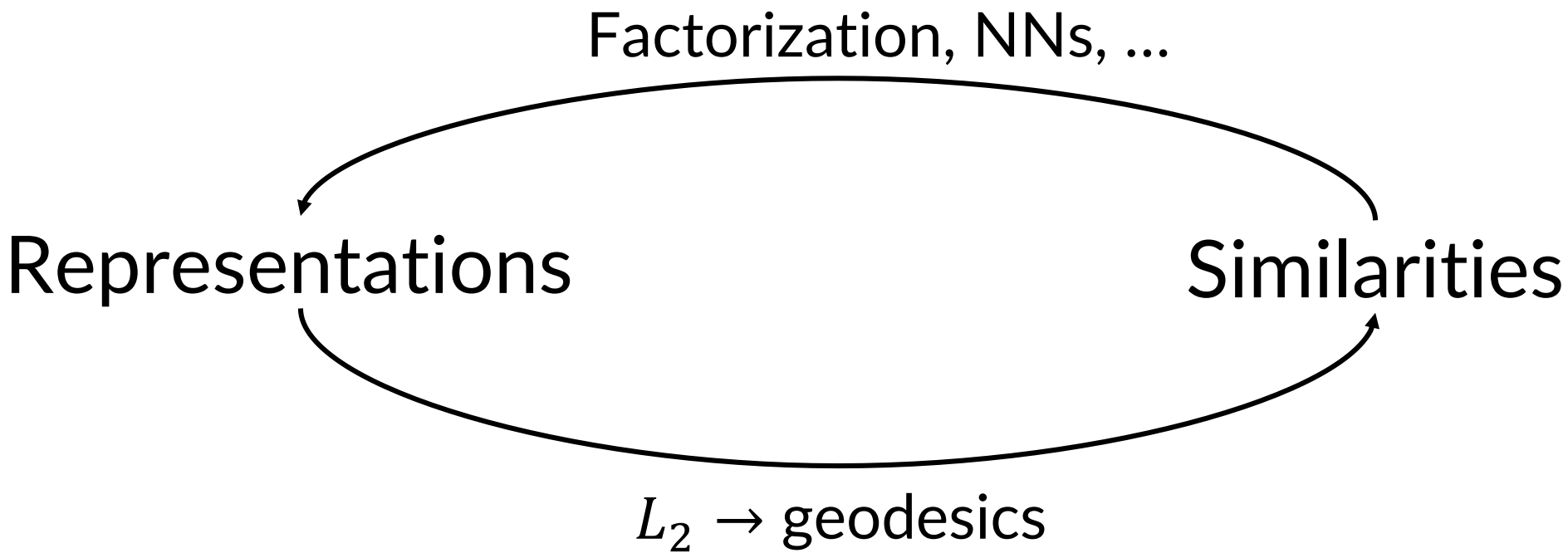
What makes a Representation?

Good representation **preserves geometry** of the original space.



What makes a Representation?

Good representation **preserves geometry** of the original space.



Part I: node representations

VERSE: **Vers**atile Graph **E**mbeddings from Similarity Measures

Anton Tsitsulin[1] Davide Mottin[1] Panagiotis Karras[2] Emmanuel Müller[1]

HPI Hasso Plattner Institut

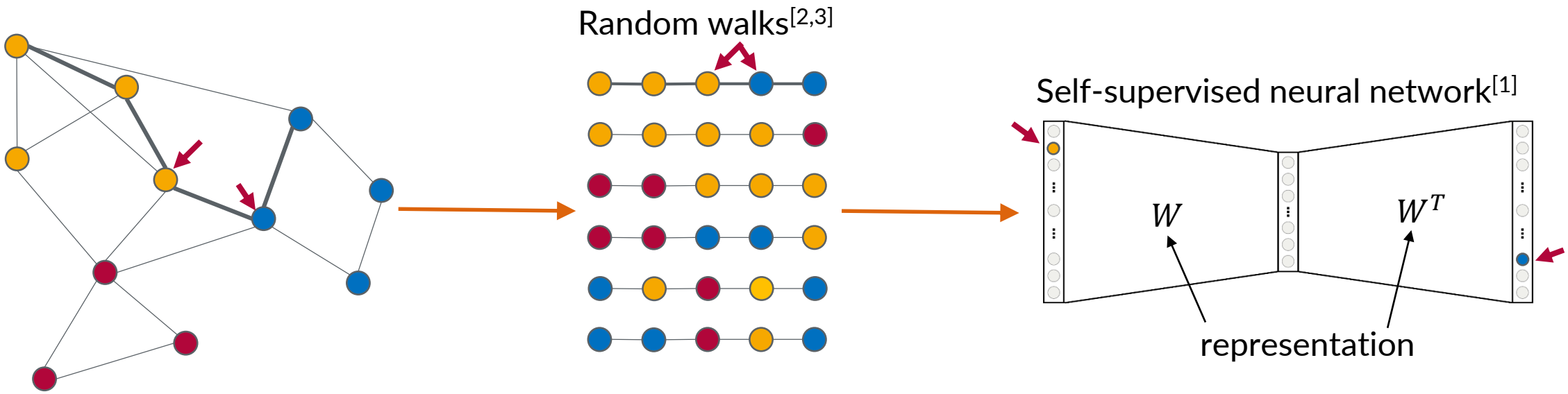
1

AARHUS UNIVERSITY

2

Neural node representations

Nodes in random walks ≈ words in sentences → word2vec



[1] Efficient Estimation of Word Representations in Vector Space, Mikolov et al., NIPS 2013
[2] DeepWalk: Online Learning of Social Representations, Perozzi et al., KDD 2014
[3] node2vec: Scalable Feature Learning for Networks, Grover & Leskovec, KDD 2016

Wait, what?

Do we know what do these walks mean?

- What do parameters change?
- What does the model optimize?

word2vec can be understood as matrix factorization!

[1] Metric recovery from directed unweighted graphs, Hashimoto et al., AISTATS 2015
[2] Neural Word Embedding as Implicit Matrix Factorization , Levy & Goldberg, NIPS 2014

Wait, what?

Do we know what do these walks mean?

- What do parameters change?
- What does the model optimize?

word2vec can be understood as matrix factorization!

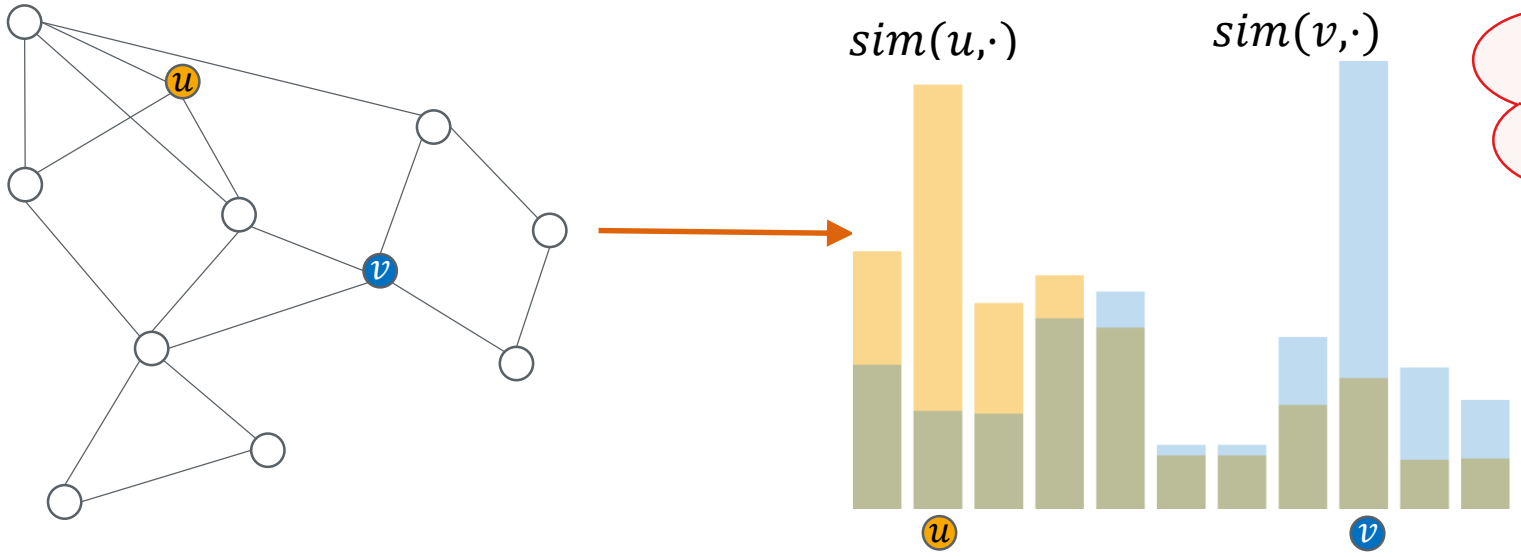
Yes, but the assumptions are too strict!

(dimensionality = number of nodes)

[1] Metric recovery from directed unweighted graphs, Hashimoto et al., AISTATS 2015
[2] Neural Word Embedding as Implicit Matrix Factorization , Levy & Goldberg, NIPS 2014

Key observation

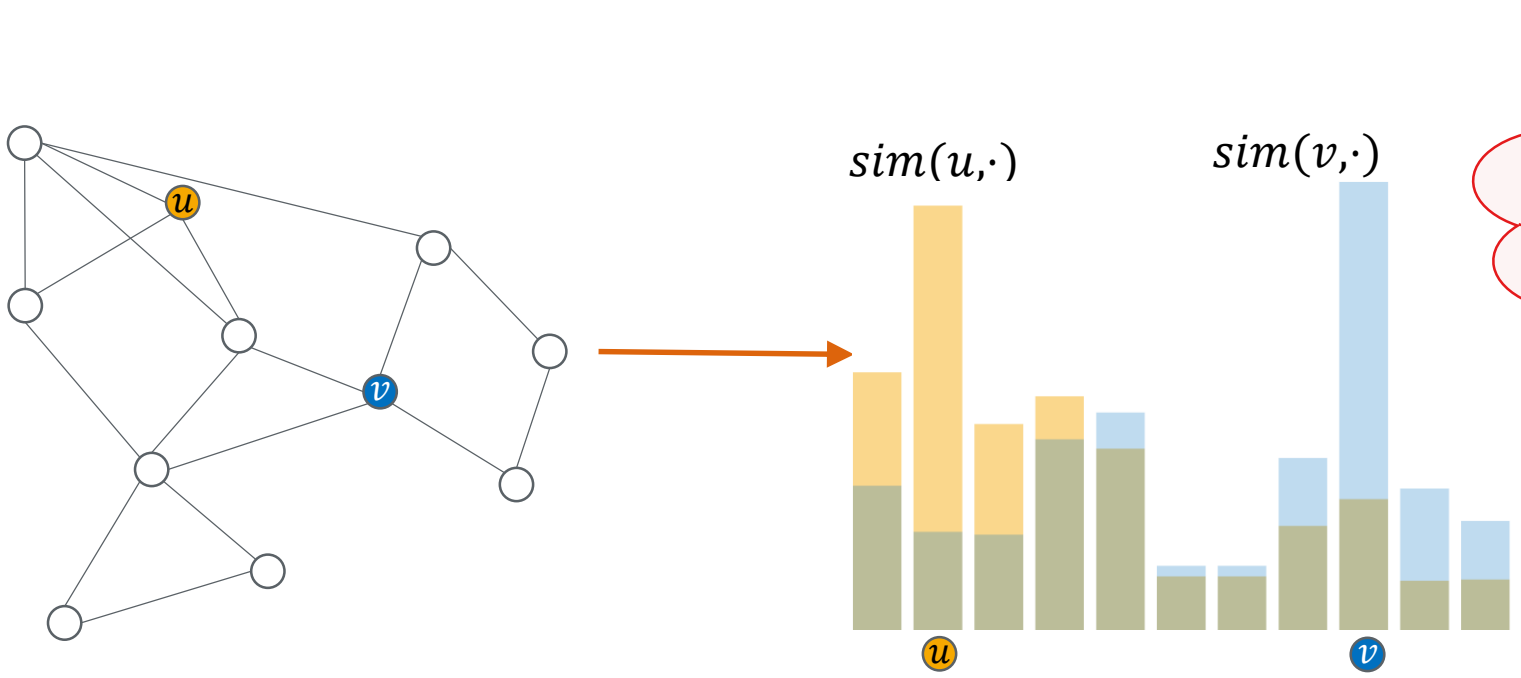
Random walks define node similarity distributions!



Random walks converge to Personalized PageRank

Key observation

Random walks define node similarity distributions!

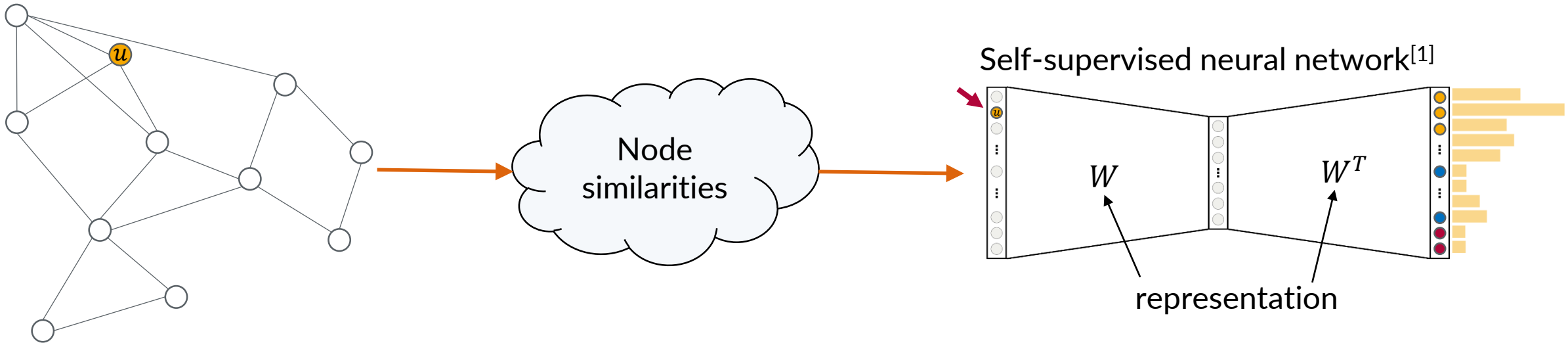


Random walks converge to Personalized PageRank

Q: Can we inject similarities fully into the model?

Yes, we can!

VERSE can learn similarity distributions



Q1: Which similarities can we possibly represent?

Q2: What other methods have to do with similarities?

Why similarities?

- We can explicitly <u>measure the quality</u>

Why similarities?

- We can measure the quality
- We can adapt the similarity to the data/task
  - Examples in the paper: PageRank, SimRank, adjacency

Why similarities?

- We can measure the quality
- We can adapt the similarity to the data/task
  - Examples in the paper: PageRank, SimRank, adjacency
- Thinking about similarities provides insight:
  - We show how DeepWalk & node2vec ≈ PPR
  - VERSE uses 1 parameter instead of 5

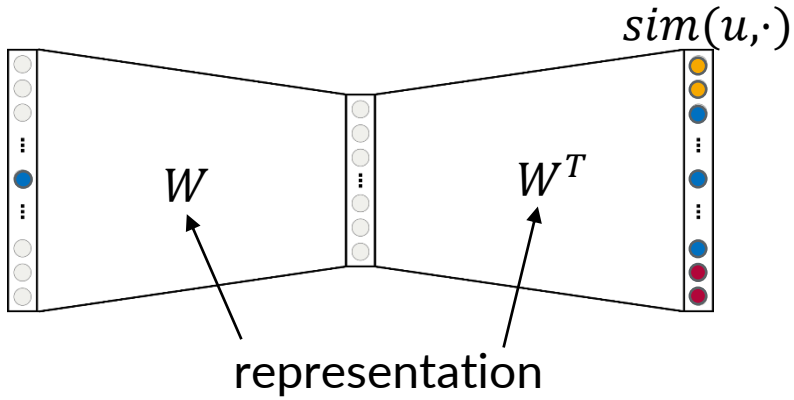
VERSE graph embedding

Algorithm for given $sim(u,\cdot)$:

- Initialize $W \sim \mathcal{N}(0,1)$
- For $u \in V$ optimize W for softmax $sim(u,\cdot)$ by gradient descent

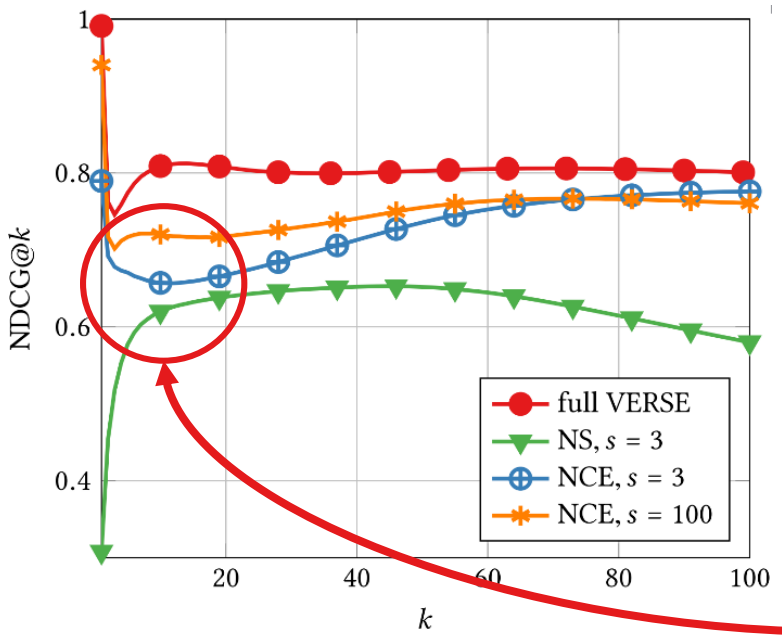
Full updates are <u>too expensive</u> - $O(n^2)$

We make it faster via sampling!



Sampling in VERSE

We use Noise Contrastive Estimation



$$\mathcal{L}_{NCE} = \sum_{\substack{u \sim \mathcal{P} \\ v \sim sim_{\mathrm{G}}(u,\cdot)}} \Big[\log \Pr_W(D = 1 | sim_{\mathrm{E}}(u, v)) + k\mathbb{E}_{\widetilde{v} \sim Q(u)} \log \Pr_W(D = 0 | sim_{\mathrm{E}}(u, \widetilde{v})) \Big]$$

Negative Sampling does not preserve similarities!

Experimental setup

- Goal: diverse tasks & datasets
- PPR as default similarity
- Max. one day on 10-core server (try that at home!)
- Code @ GitHub, C for performance (don’t try that at home!)

Social graph × link prediction

method	*edge representation*				
	Average	Concat	Hadamard	L1	L2
VERSE	73.78	73.66	79.71	74.11	74.56
DEEPWALK	70.05	69.92	69.79	78.38	77.37
LINE	75.17	75.13	72.54	63.77	64.47
HOPE	71.89	71.90	70.22	71.22	70.63
HSVERSE	74.14	74.02	80.26	73.04	73.53
NODE2VEC	71.29	71.22	72.43	78.38	78.66
Feature Eng.			78.84		

Link prediction

(accuracy)

Different graphs × node clustering

method	CoCit	CoAuthor	VK	YouTube	Orkut
VERSE	69.43	79.25	45.78	67.63	42.64
DeepWalk	70.04	73.83	43.30	58.08	44.66
LINE	60.02	71.58	39.65	63.40	42.59
GraRep	67.61	77.40	—	—	—
HOPE	42.45	69.57	21.70	37.94	—
hsVERSE	69.81	79.31	45.84	69.13	—
Node2vec	70.06	75.78	44.27	—	—
Louvain	72.05	84.29	46.60	71.06	—

Node clustering

(modularity)

Web graph × node classification

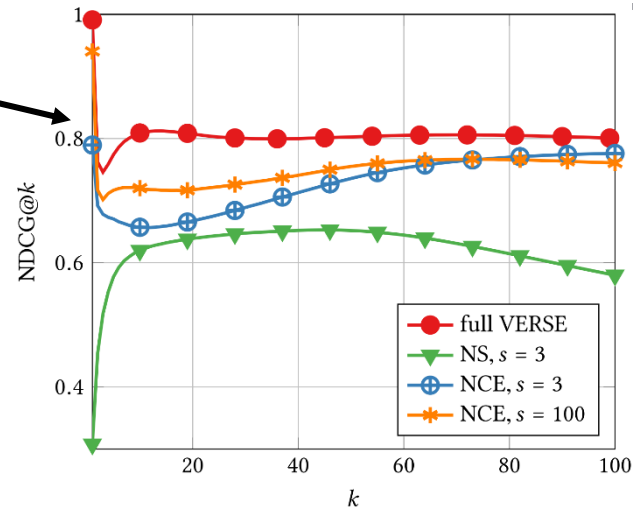
method	*labelled nodes, %*				
	1%	3%	5%	7%	9%
VERSE	17.92	22.26	24.07	25.07	25.99
DEEPWALK	18.16	21.55	22.89	23.64	24.54
LINE	13.71	17.36	18.69	19.84	20.64
HOPE	9.22	13.80	15.09	16.18	16.78
HSVERSE	18.16	22.84	25.40	27.38	29.09

Classification

(accuracy)

Take home messages

- We provide new useful abstraction: node similarities
- We create **VERSE** to explicitly work with similarities
- We develop a scalable approximation technique with NCE
- There is a room for improvement!



Part II: graph representations

NetLSD: Hearing the Shape of a Graph

Anton Tsitsulin[1] Davide Mottin[1] Panagiotis Karras[2]
Alex Bronstein[3] Emmanuel Müller[1]

[1] HPI
Germany

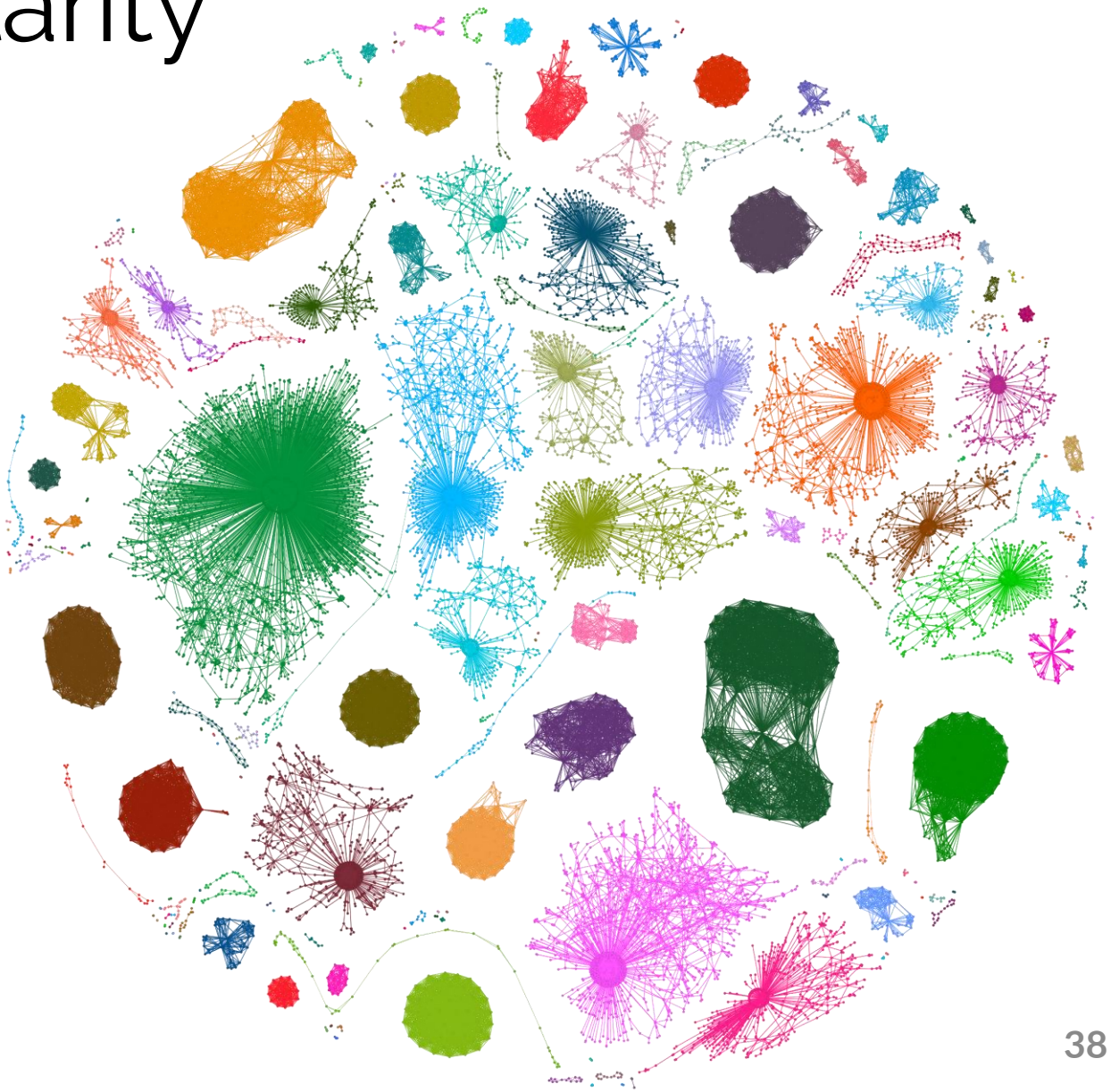
[2] Aarhus university
Denmark

[3] Technion
Israel

Defining graph similarity

With it, we can do:

- Classification
- Clustering
- Anomaly detection
- …

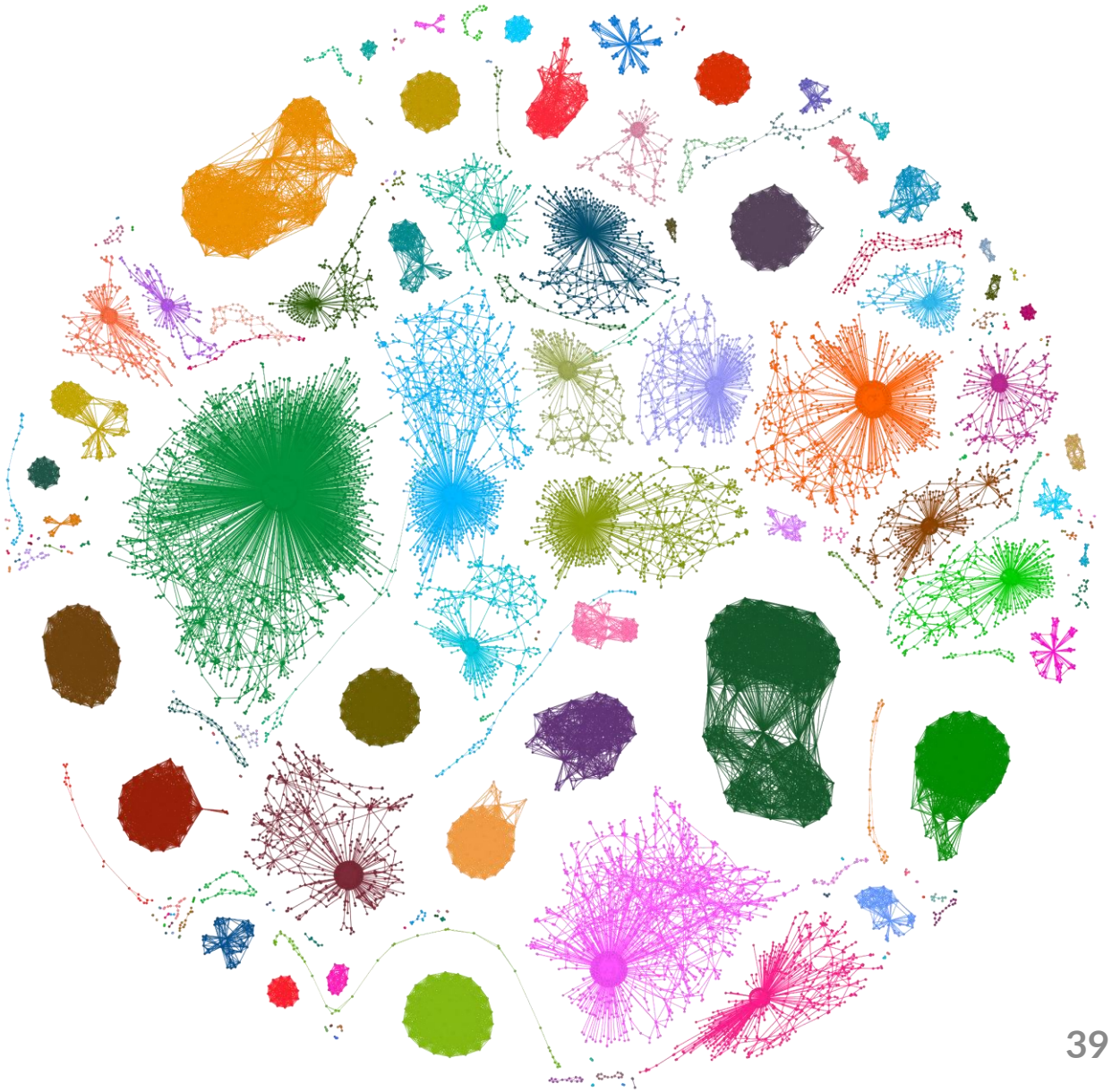


Scalability is key!

Two problem sources:

- Big graphs
- Many graphs

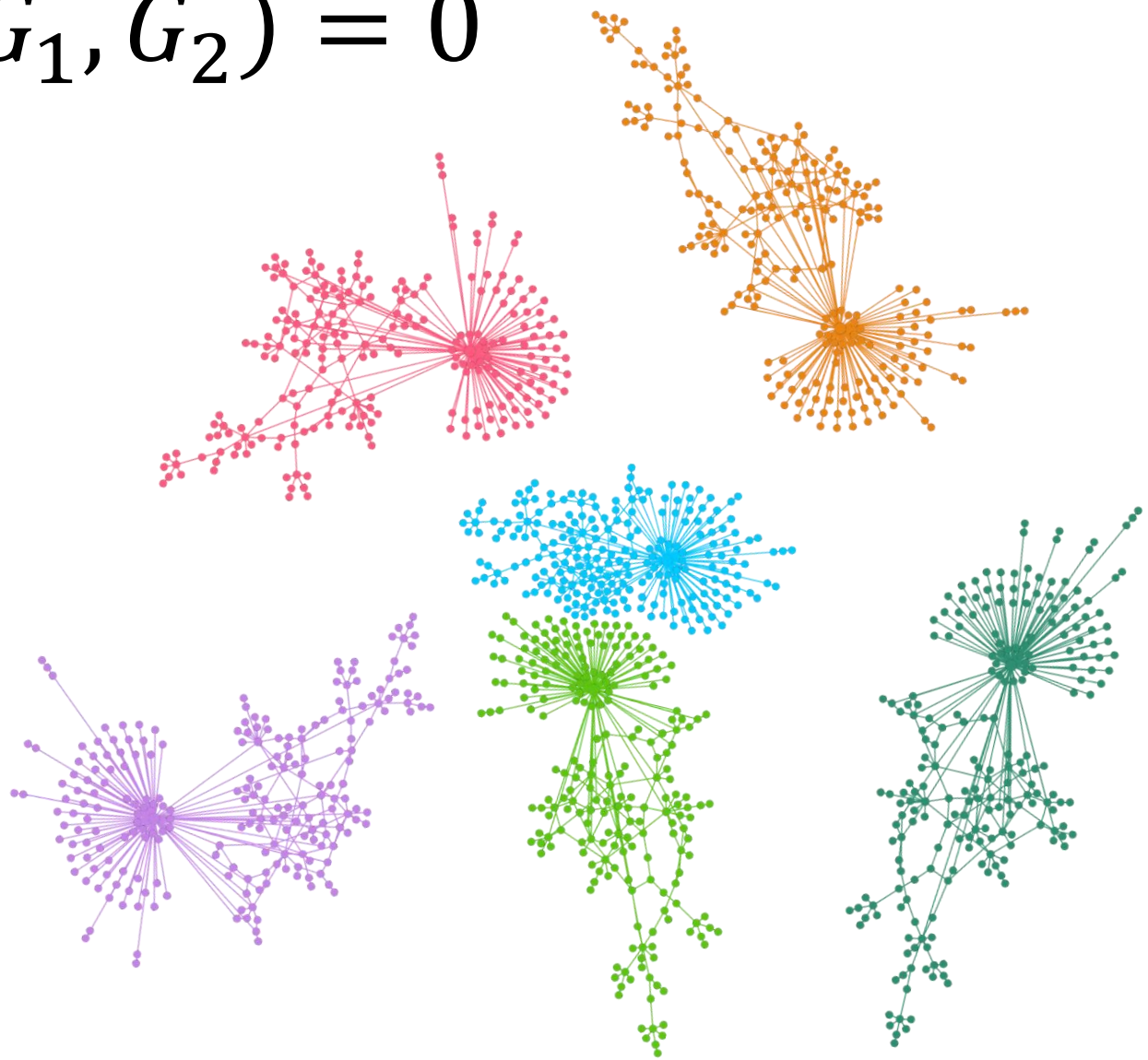
Solution: graph descriptors



Isomorphism $\Rightarrow d(G_1, G_2) = 0$

3 key properties:

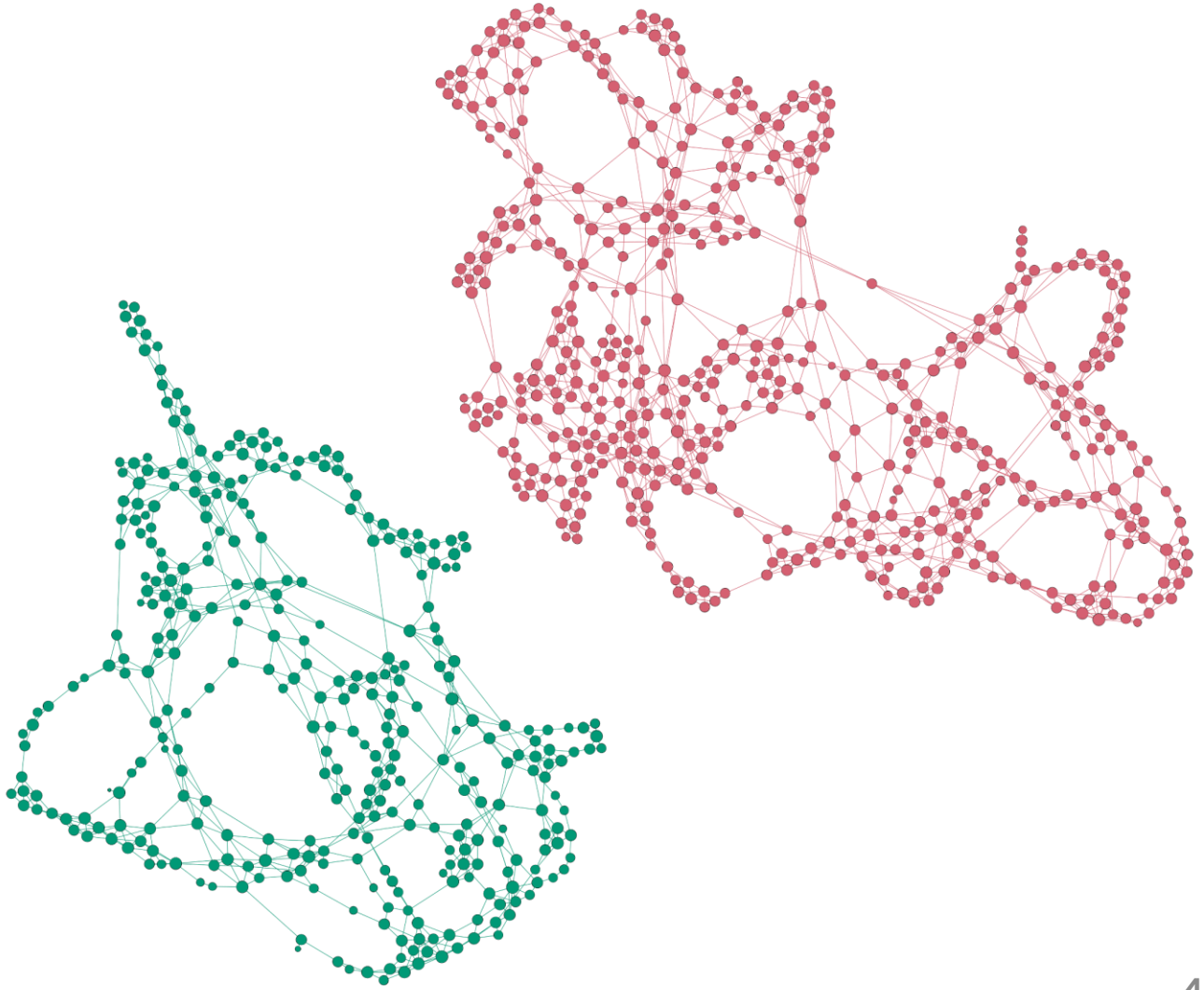
- Permutation invariance
- Scale-adaptivity
- Size invariance



Local structures are important

3 key properties:

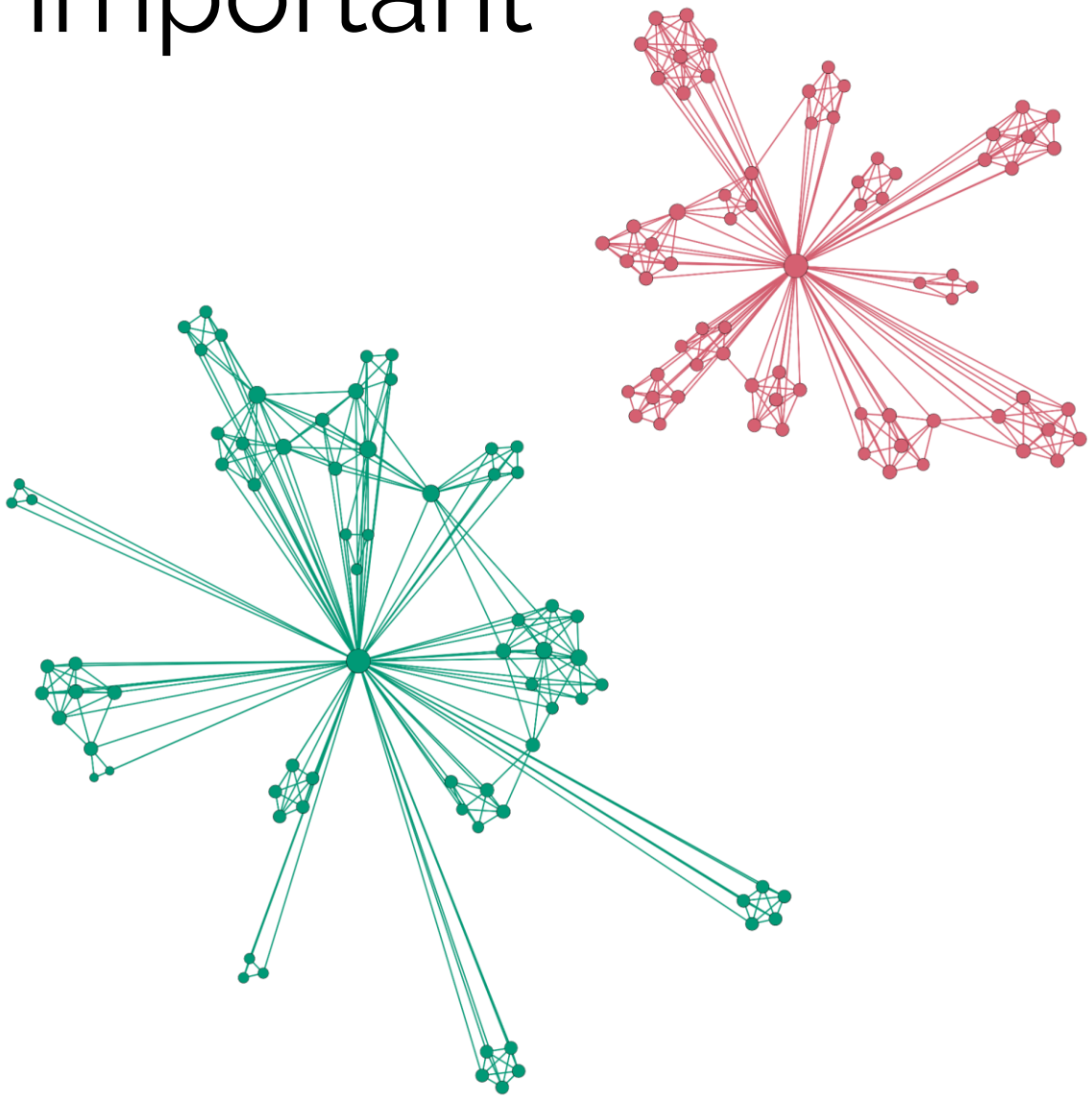
- Permutation invariance
- Scale-adaptivity
- Size invariance



Global structure is important

3 key properties:

- Permutation invariance
- Scale-adaptivity
- Size invariance



We may need to disregard the size

3 key properties:

- Permutation invariance
- Scale-adaptivity
- Size invariance

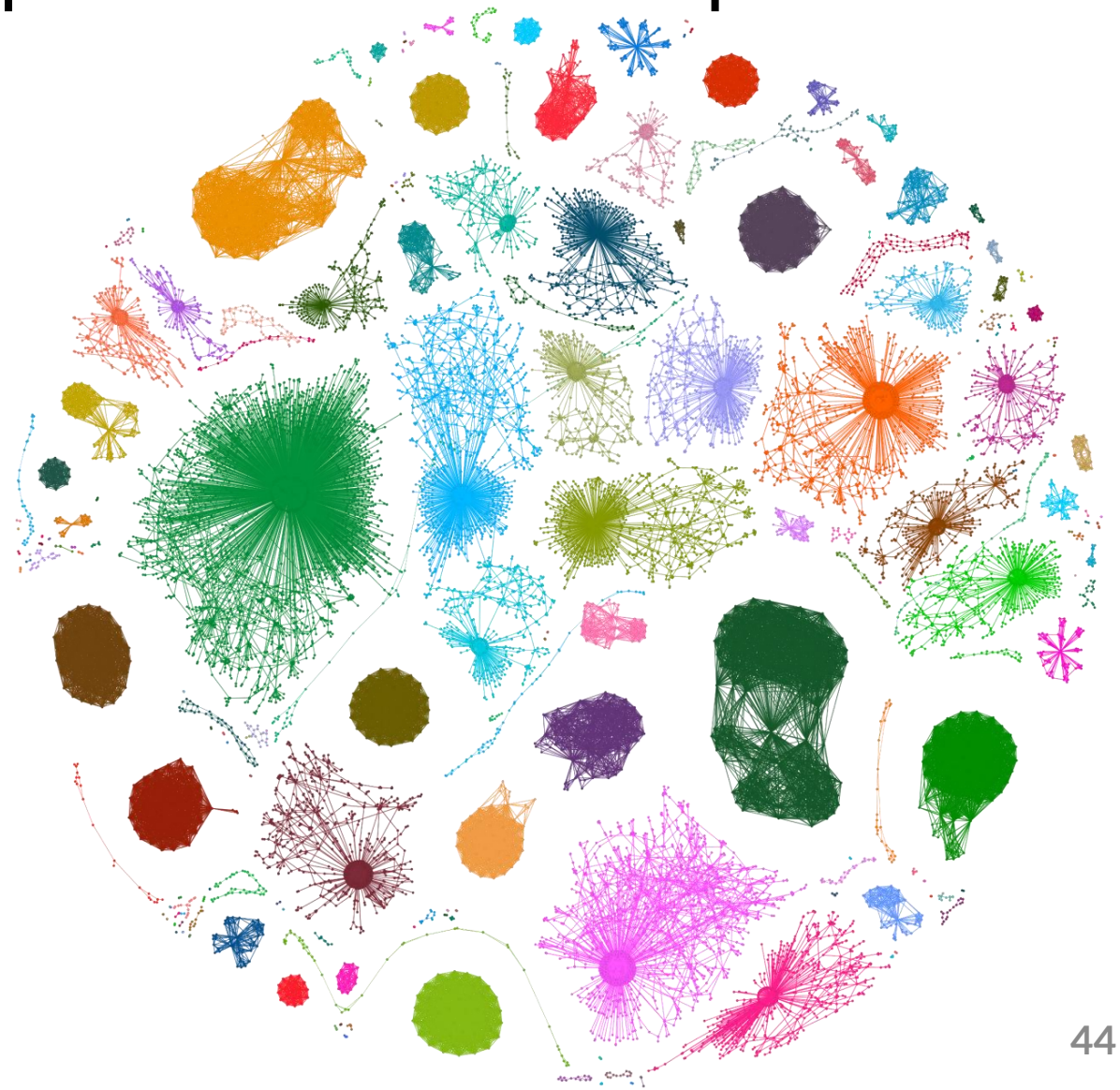
Network Laplacian Spectral Descriptors

3 key properties:

- Permutation invariance
- Scale-adaptivity
- Size invariance

+ Scalability

= **NetLSD**



Optimal Transport

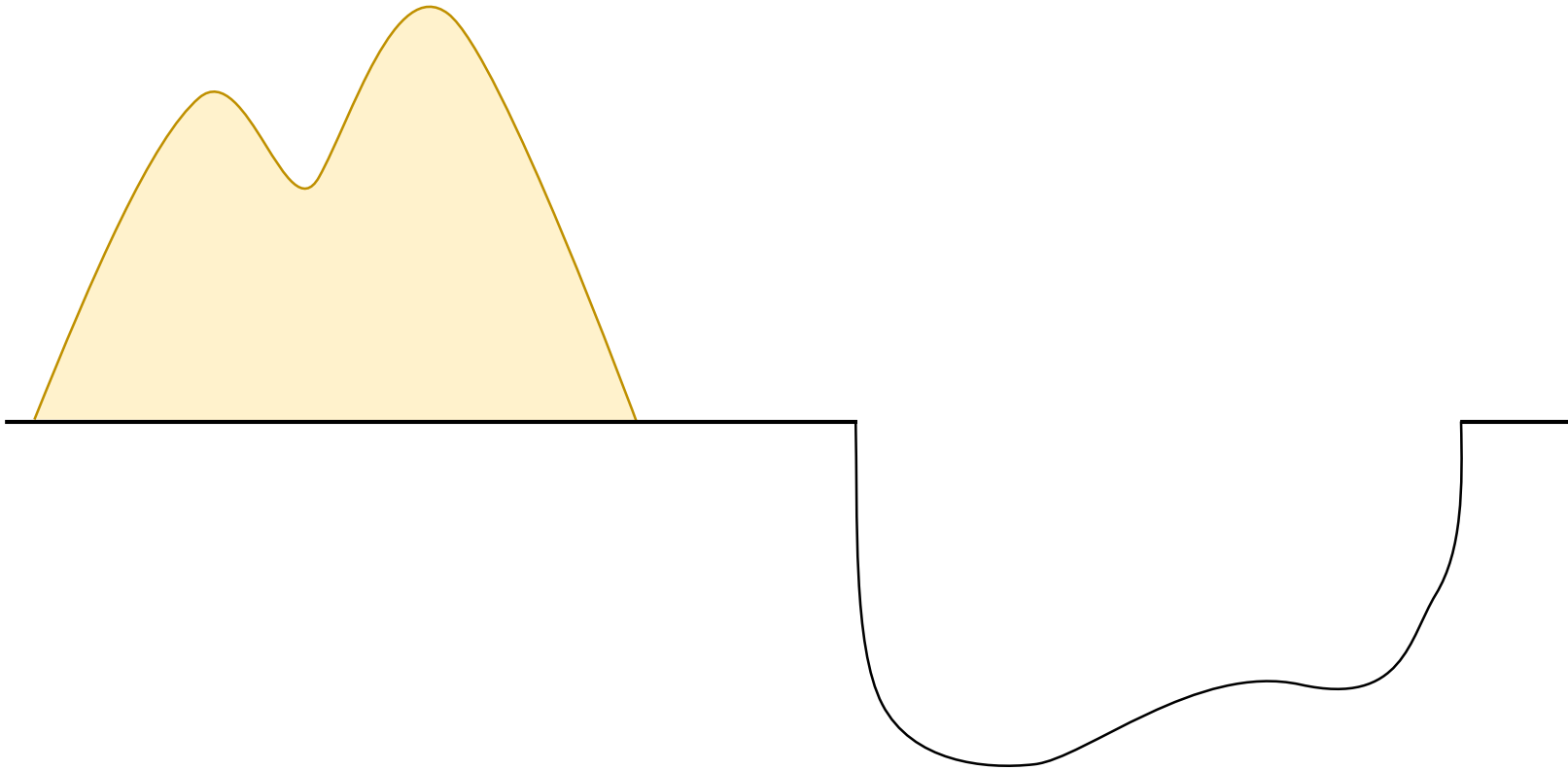
Geometry for probability measures supported on a space.

Optimal Transport

Geometry for probability measures supported on a space.

G. Monge
1781

L. Kantorovich
1939



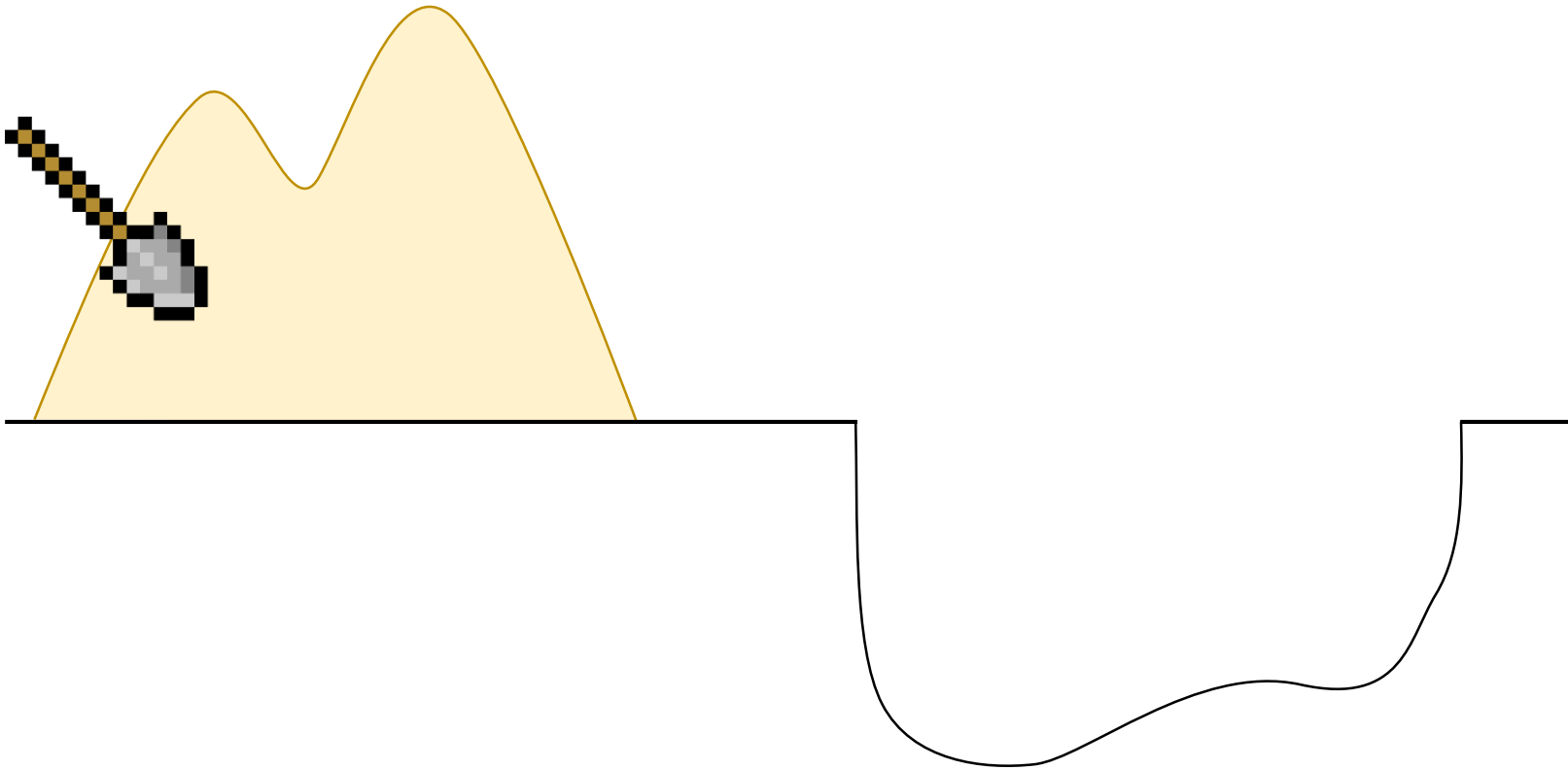
Optimal Transport

Geometry for probability measures supported on a space.

G. Monge
1781

L. Kantorovich
1939

C. Villani
2003

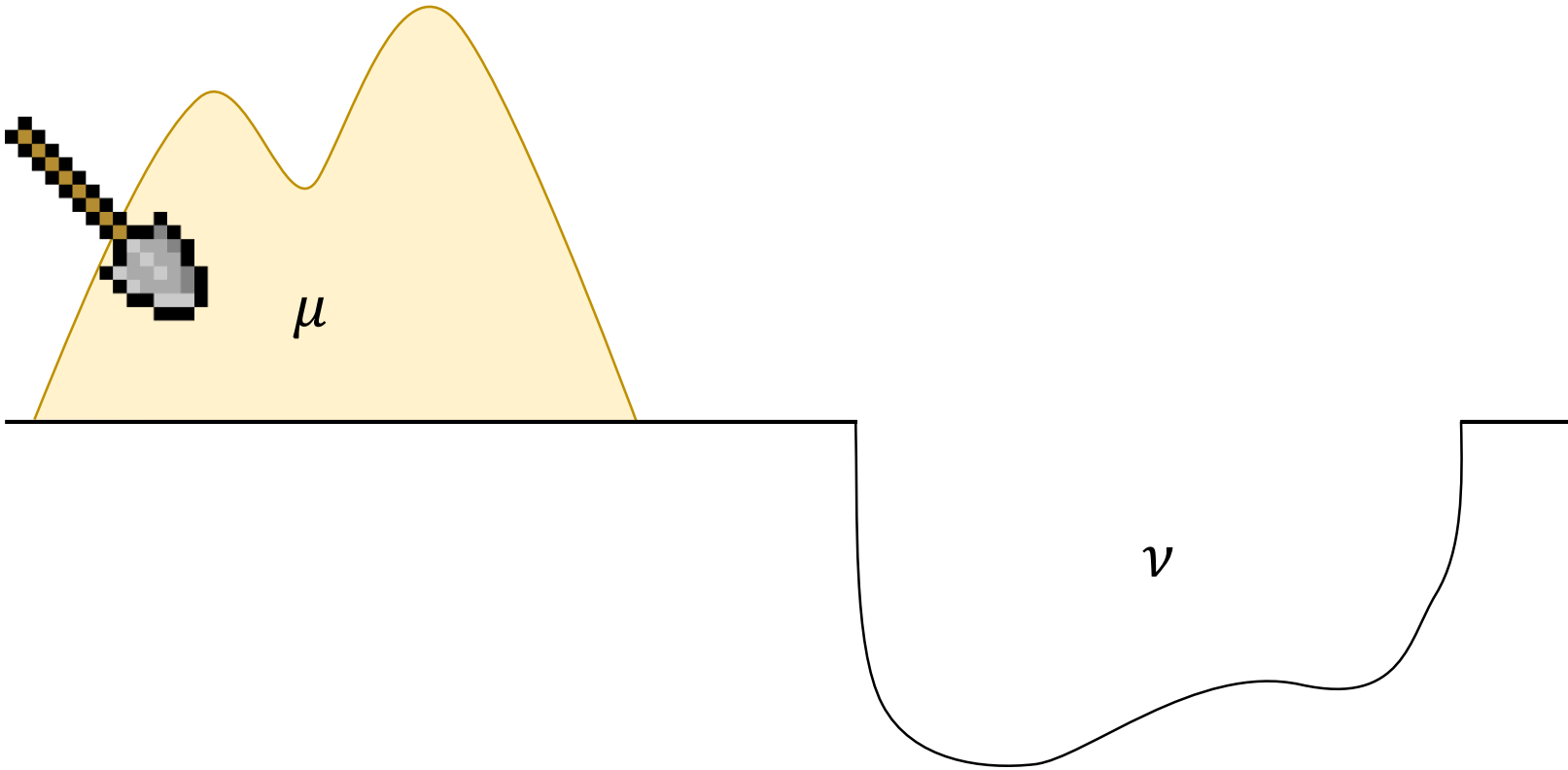


Optimal Transport

Geometry for probability measures supported on a space.

G. Monge
1781

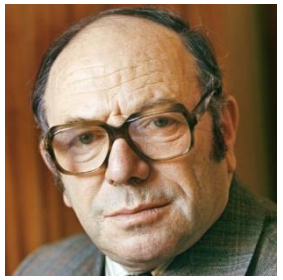
L. Kantorovich
1939



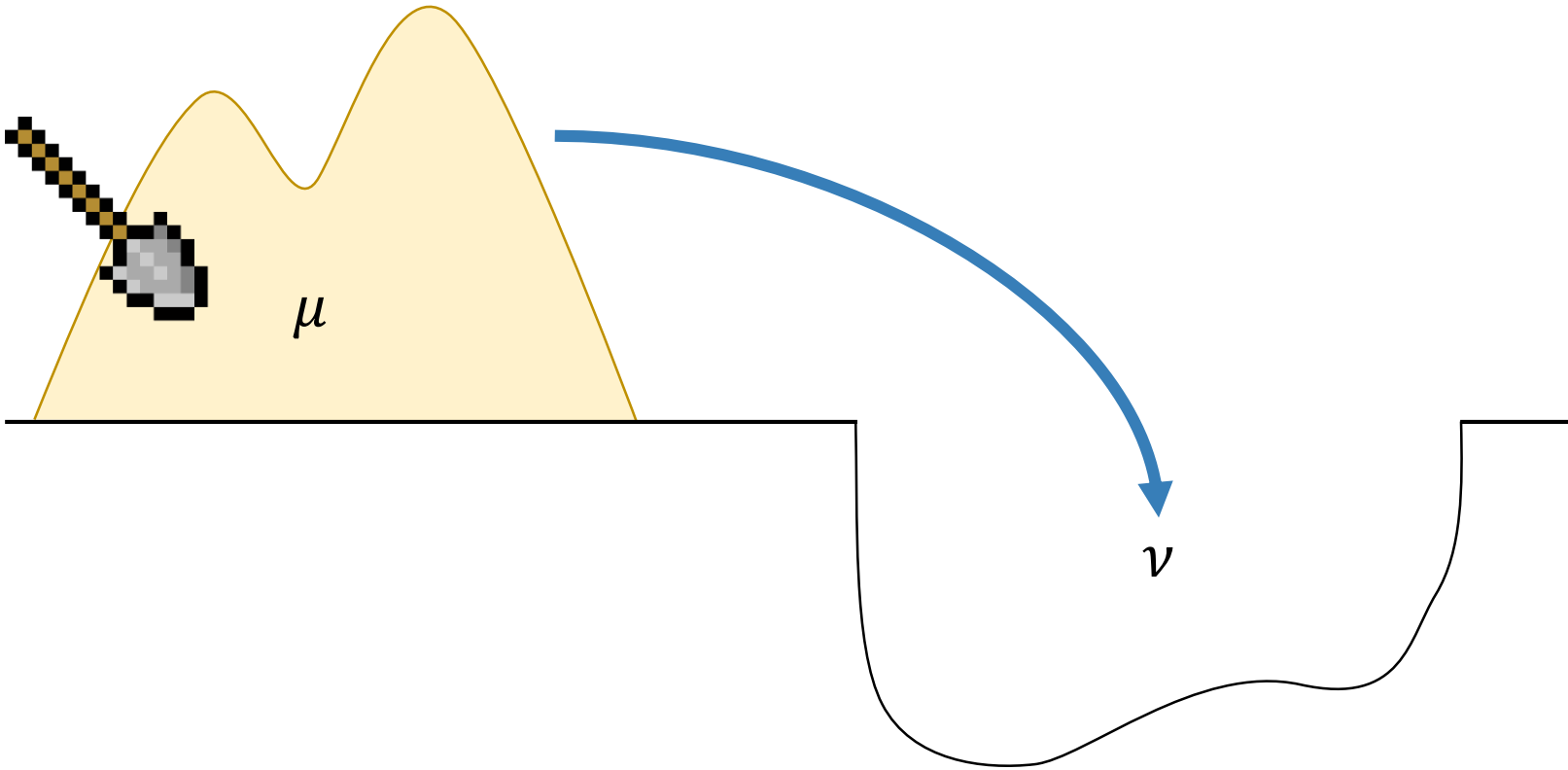
Optimal Transport

Geometry for probability measures supported on a space.

G. Monge
1781



L. Kantorovich
1939



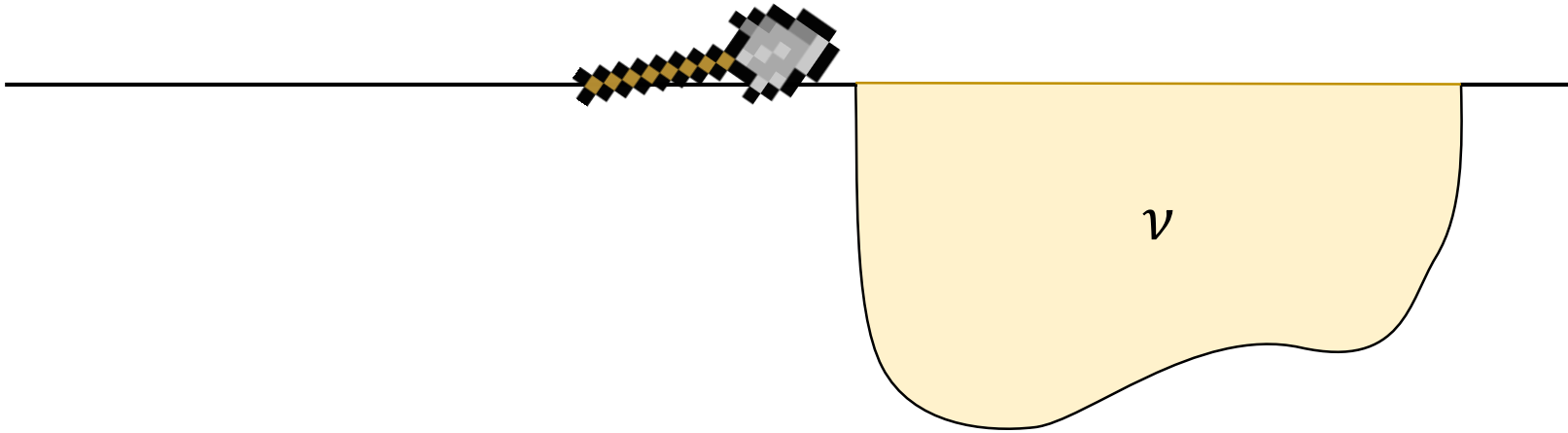
Optimal Transport

Geometry for probability measures supported on a space.

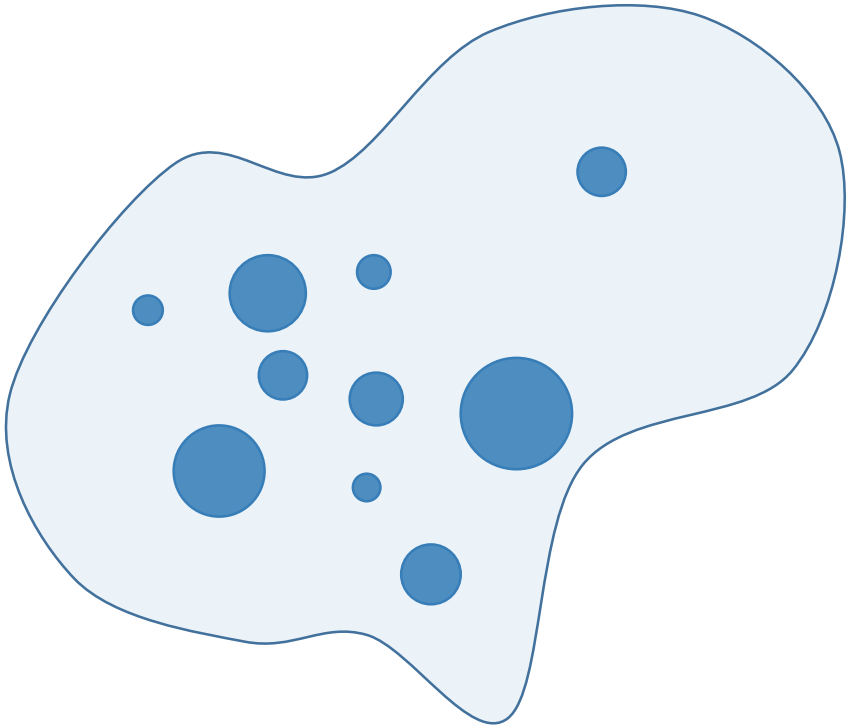
G. Monge
1781

L. Kantorovich
1939

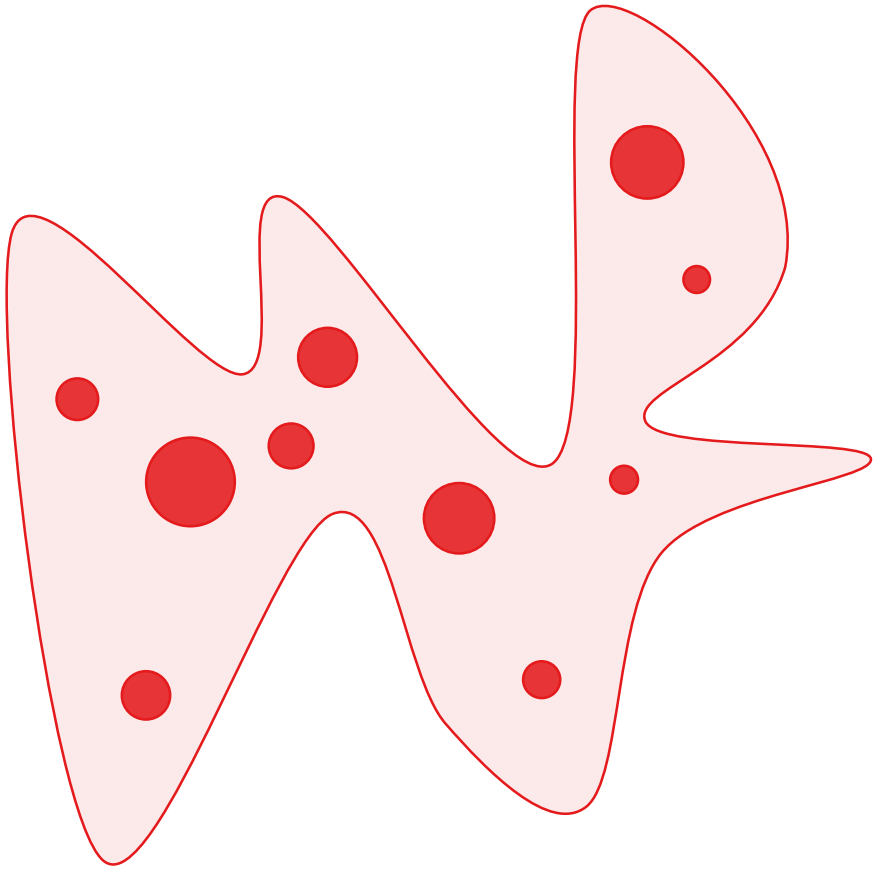
Discrete case →
Linear programming



Gromov-Wasserstein distance

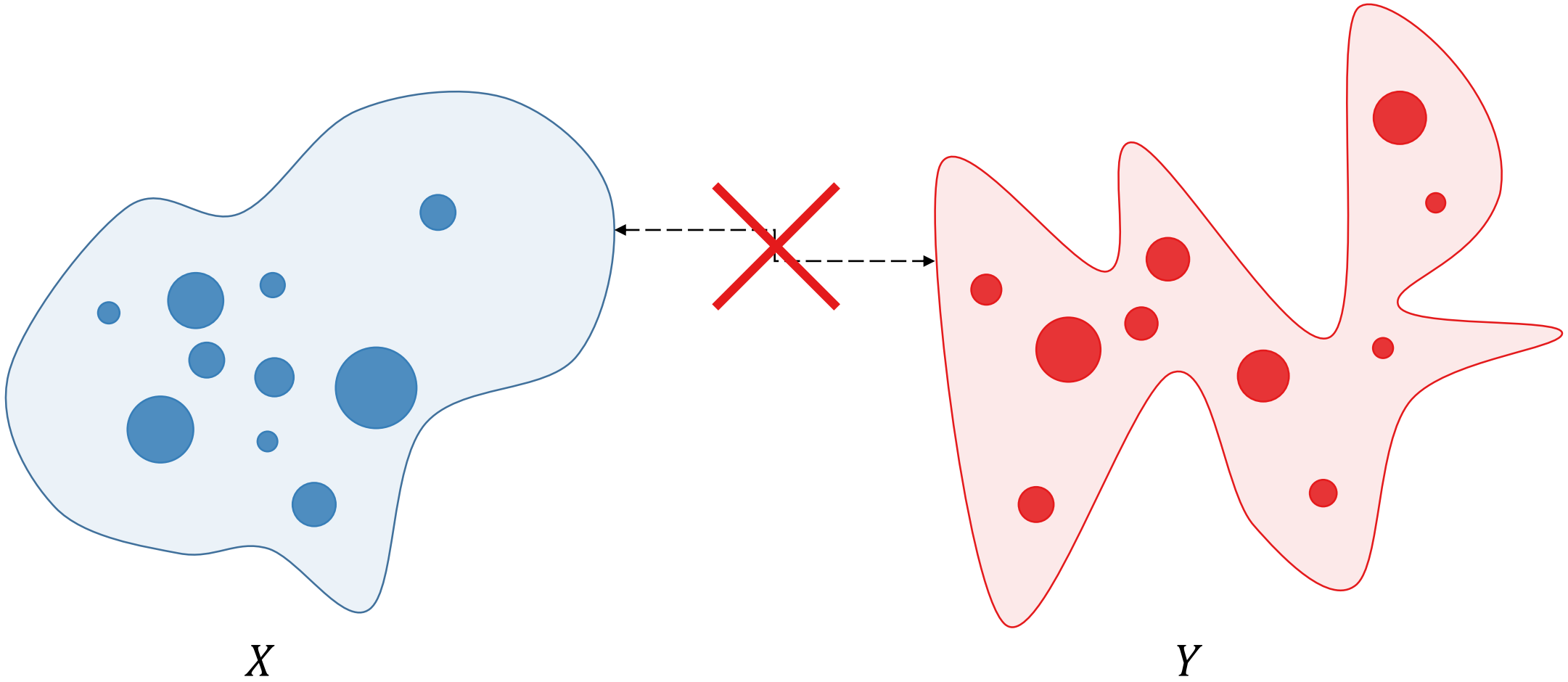


X

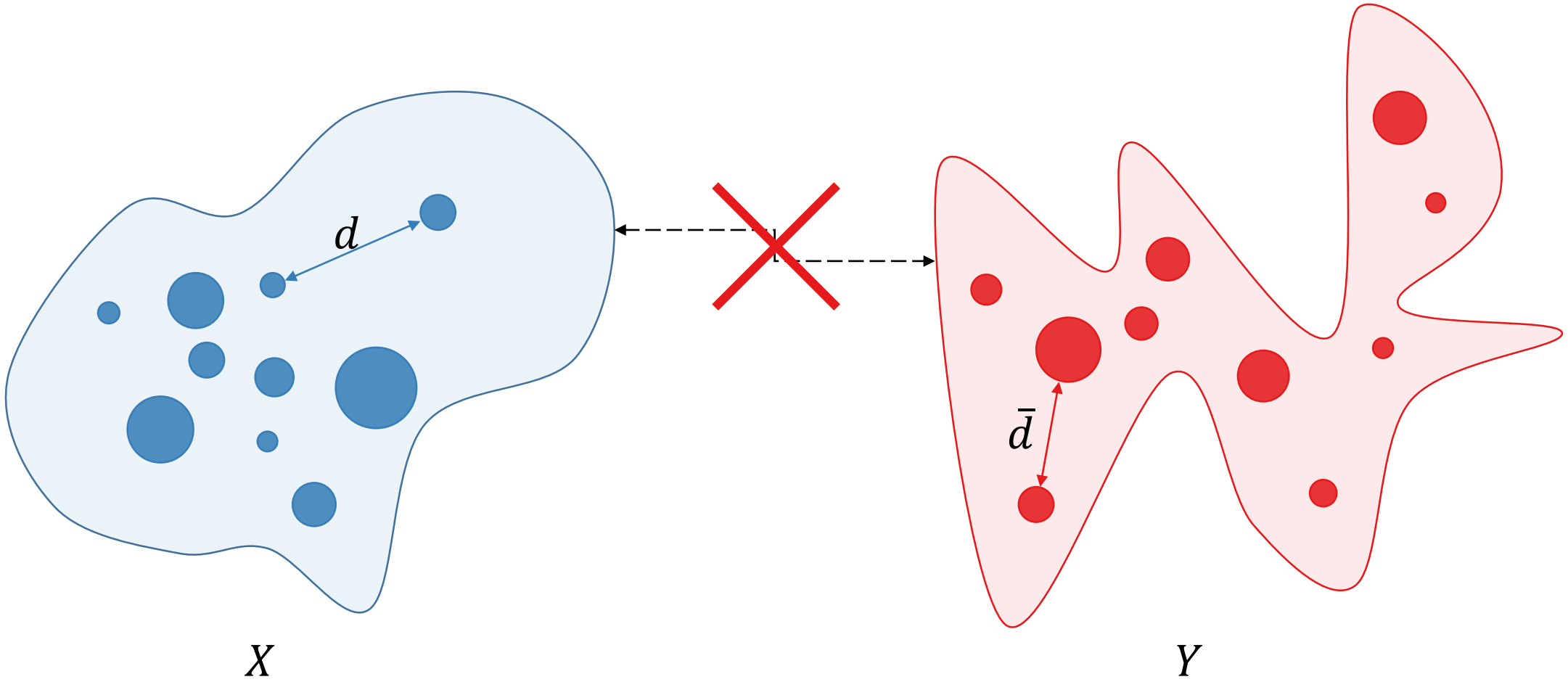


Y

Gromov-Wasserstein distance

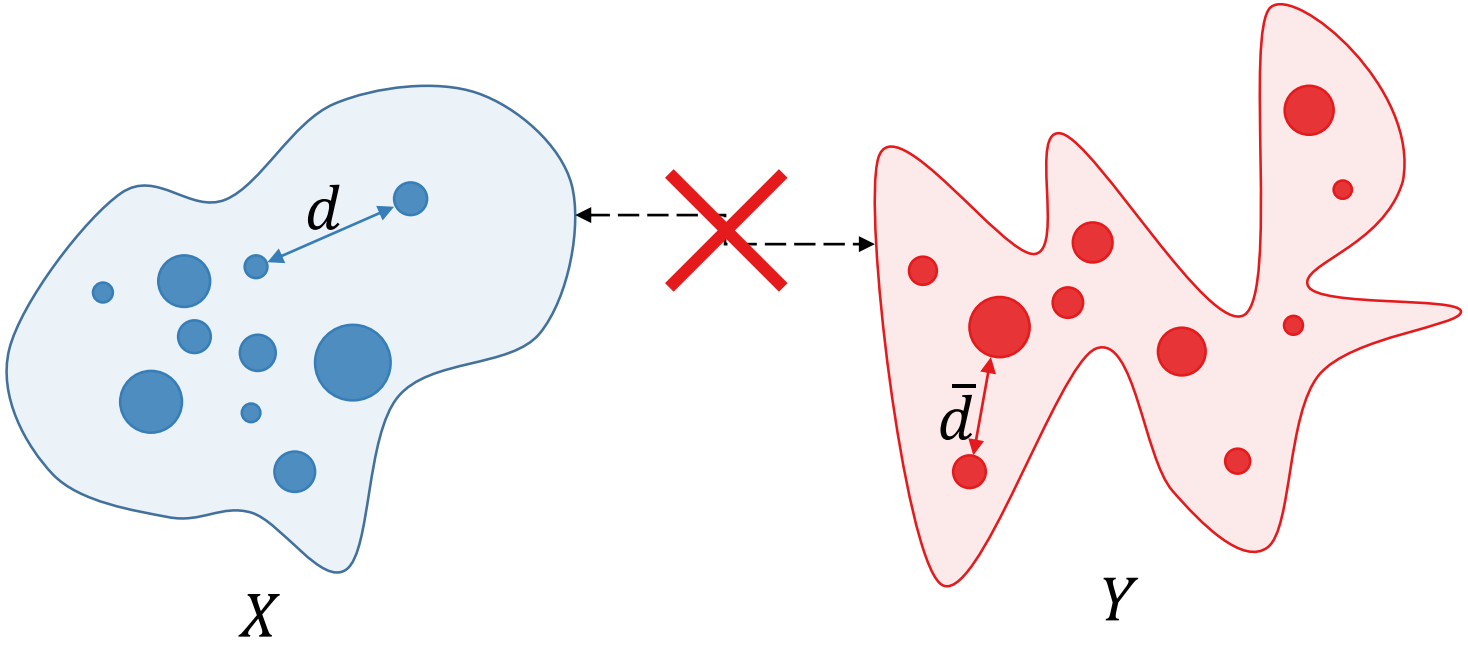


Gromov-Wasserstein distance



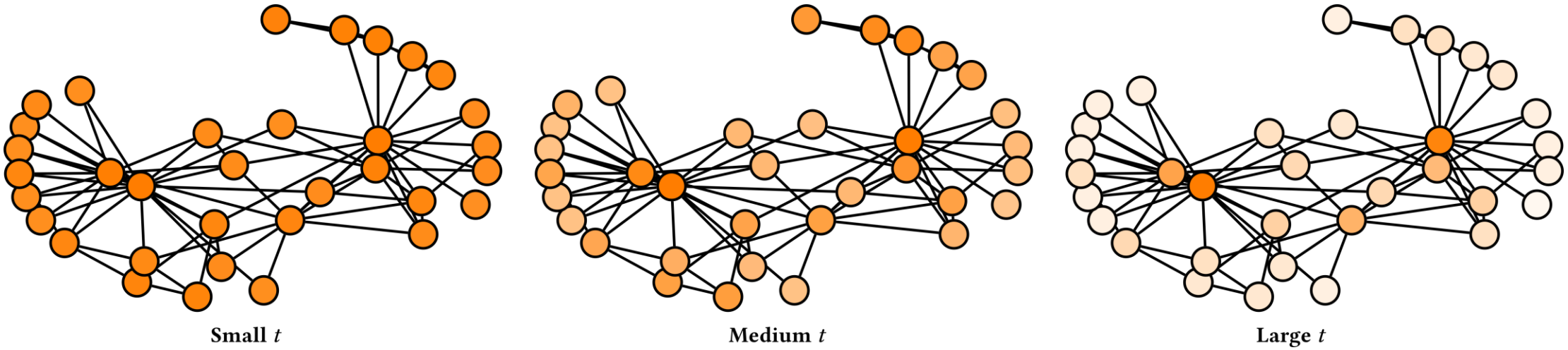
Gromov-Wasserstein distance

$$d_{\mathcal{GW},p}(X,Y) = \frac{1}{2}\left(\inf_{M}\sum_{i,j}\sum_{i',j'}\left|d(x_i,x_{i'}) - \bar{d}(y_j,y_{j'})\right|^p m_{ij}m_{i'j'}\right)^{1/p}$$



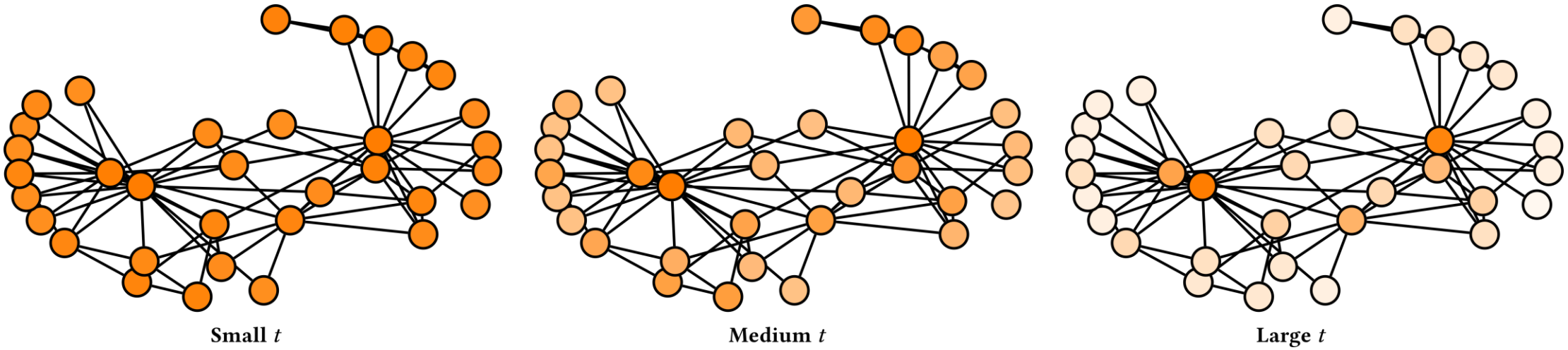
Heat diffusion has an explicit notion of scale

$$\frac{\partial u_t}{\partial t} = -\mathcal{L} u_t$$



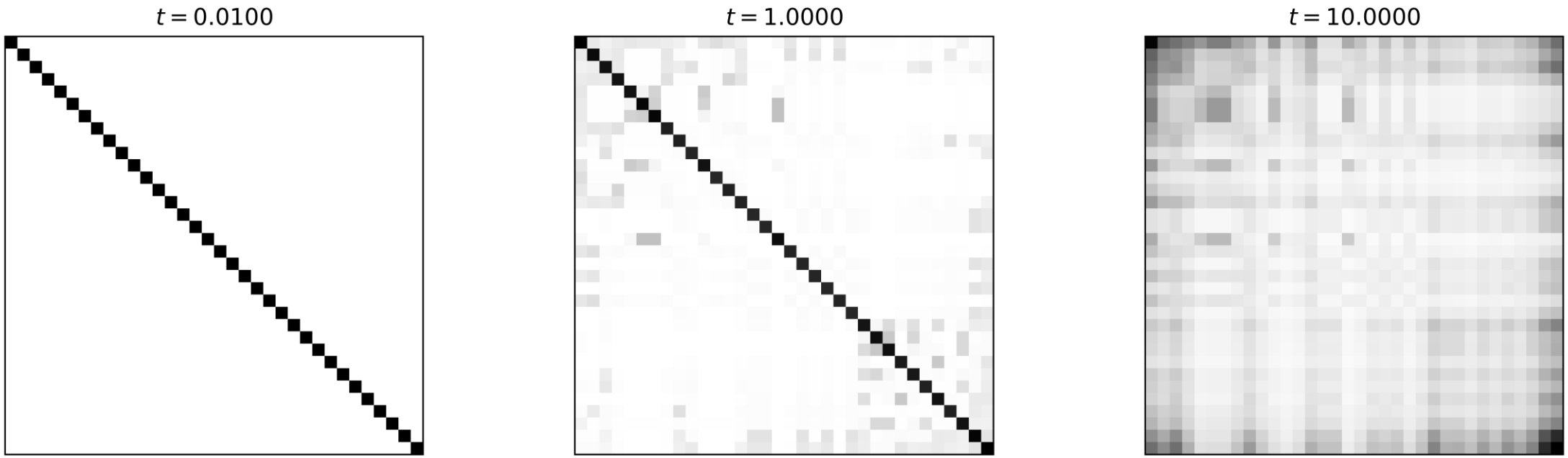
Heat kernel has an explicit notion of scale

$$H_t = e^{-t\mathcal{L}} = \Phi e^{-t\Lambda}\Phi^\top = \sum_{j=1}^{n} e^{-t\lambda_j}\phi_j\,\phi_j^\top$$



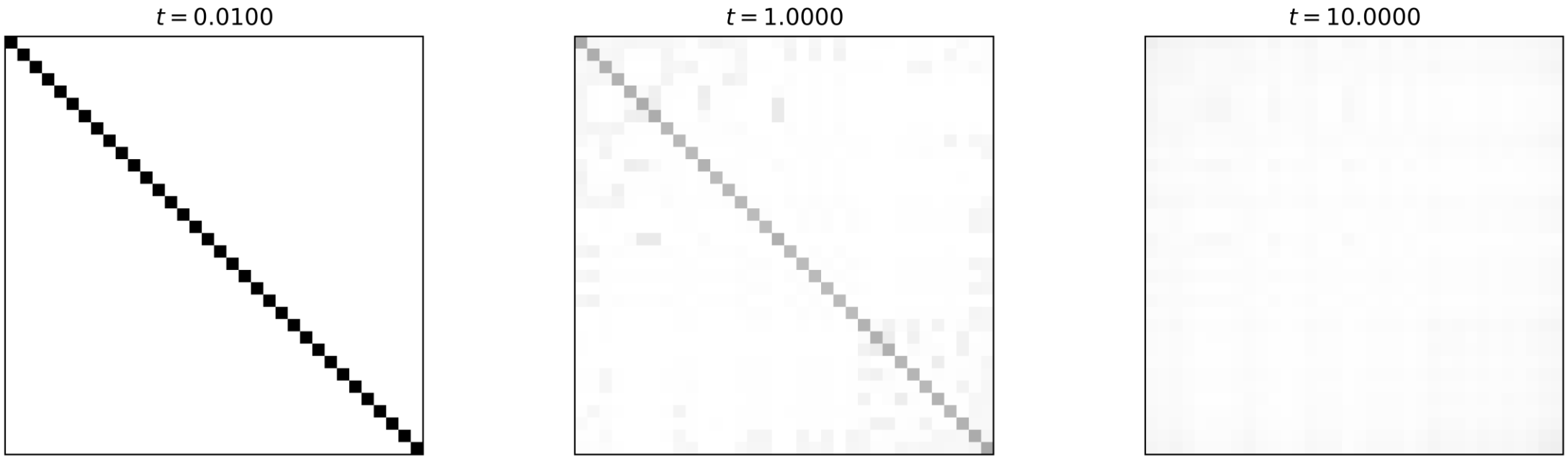
Scale corresponds to locality

$$H_t = e^{-t\mathcal{L}} = \Phi e^{-t\Lambda}\Phi^\top = \sum_{j=1}^{n} e^{-t\lambda_j}\phi_j\,\phi_j^\top$$



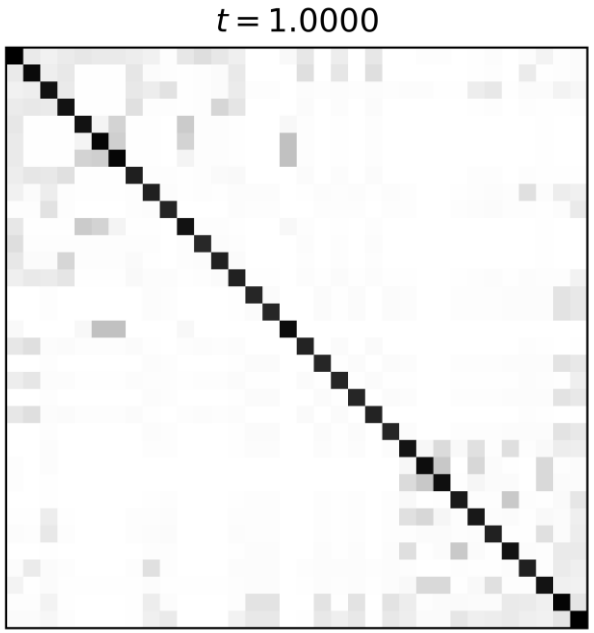
Scale corresponds to locality

$$H_t = e^{-t\mathcal{L}} = \Phi e^{-t\Lambda}\Phi^\top = \sum_{j=1}^{n} e^{-t\lambda_j}\phi_j\,\phi_j^\top$$



Spectral Gromov-Wasserstein = Gromov-Wasserstein + heat kernel

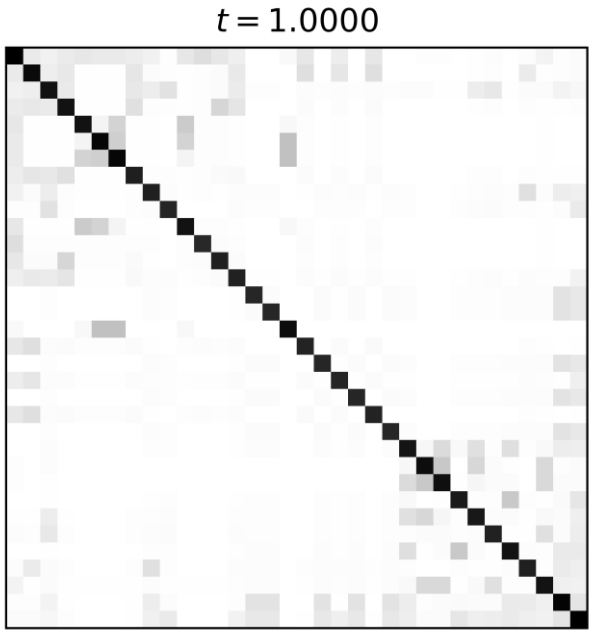
$$d^{\mathrm{spec}}_{\mathcal{GW},p}(X,Y) = \inf_{M} \sup_{t>0} e^{-2\left(t+t^{-1}\right)} \cdot \left(\sum_{i,j} \sum_{i',j'} \left| H_t^X(x_i, x_{i'}) - H_t^Y(y_j, y_{j'}) \right|^p m_{ij} m_{i'j'} \right)^{1/p}$$



Using heat kernel at all t as a distance doesn't make our task any easier

Spectral Gromov-Wasserstein has a useful lower bound!

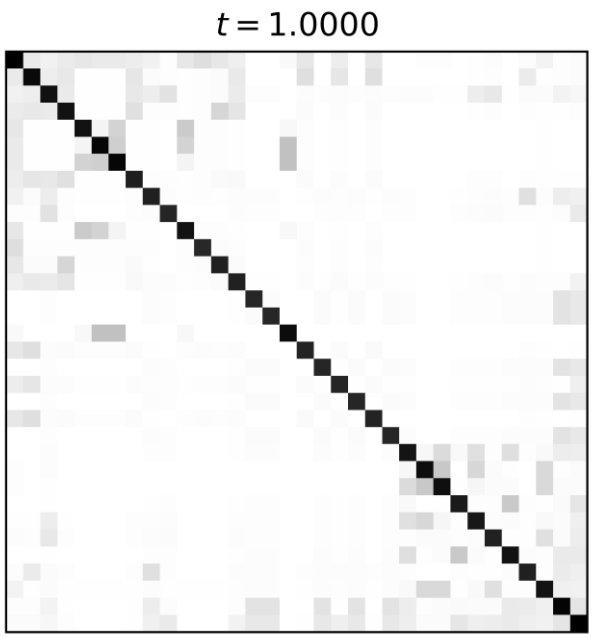
$$d^{\text{spec}}_{\mathcal{GW},p}(X,Y) = \inf_{M} \sup_{t>0} e^{-2\left(t+t^{-1}\right)} \cdot \left(\sum_{i,j} \sum_{i',j'} \left| H_t^X(x_i, x_{i'}) - H_t^Y(y_j, y_{j'}) \right|^p m_{ij} m_{i'j'} \right)^{1/p} \geq \sup_{t>0} e^{-2\left(t+t^{-1}\right)} \cdot \left| \operatorname{tr}(H^X) - \operatorname{tr}(H^Y) \right|$$



Using heat kernel at all t as a distance **does** make our task **way** easier!

Spectral Gromov-Wasserstein has a useful lower bound!

$$d^{\text{spec}}_{\mathcal{GW},p}(X,Y) = \inf_M \sup_{t>0} e^{-2\left(t+t^{-1}\right)} \cdot \left(\sum_{i,j} \sum_{i',j'} \left| H_t^X(x_i, x_{i'}) - H_t^Y(y_j, y_{j'}) \right|^p m_{ij} m_{i'j'} \right)^{1/p} \geq \sup_{t>0} e^{-2\left(t+t^{-1}\right)} \cdot \left| \operatorname{tr}(H^X) - \operatorname{tr}(H^Y) \right|$$



Using heat kernel at all t as a distance **does** make our task **way** easier!

We can just compare heat traces!

Network Laplacian Spectral Descriptors

$$h_t = \mathrm{tr}(H_t) = \sum_j e^{-t\lambda_j}$$

We sample t logarithmically, and compare h_t with L_2 distance
However, h_t is size-dependent!

Size invariance = normalization

$$h_t = \mathrm{tr}(H_t) = \sum_j e^{-t\lambda_j}$$

We can normalize by h_t of the complete (K) or empty graph $\overline{K}$
Computation of all λ is still expensive: $O(n^3)$

Scalability

We propose two options:

1. Use local Taylor expansion: $h_t = \mathrm{tr}(e^{-t\mathcal{L}}) = \sum_{k=0}^{\infty} \frac{\mathrm{tr}((-t\mathcal{L})^k)}{k!} \approx n - t\ \mathrm{tr}(\mathcal{L}) + \frac{t^2}{2}\mathrm{tr}(\mathcal{L}^2) + \ldots$

   Second term is degree distribution; third is weighted triangle count

Scalability

We propose two options:

1. Use local Taylor expansion: $h_t = \mathrm{tr}(e^{-t\mathcal{L}}) = \sum_{k=0}^{\infty} \frac{\mathrm{tr}((-t\mathcal{L})^k)}{k!} \approx n - t\ \mathrm{tr}(\mathcal{L}) + \frac{t^2}{2}\mathrm{tr}(\mathcal{L}^2) + \dots$

   Second term is degree distribution; third is weighted triangle count

2. Compute top + bottom eigenvalues, approximate the rest

   Linear extrapolation = explicit assumption on the manifold (Weyl's law)

Scalability

We propose two options:

1. Use local Taylor expansion: $h_t = \mathrm{tr}(e^{-t\mathcal{L}}) = \sum_{k=0}^{\infty} \frac{\mathrm{tr}((-t\mathcal{L})^k)}{k!} \approx n - t\ \mathrm{tr}(\mathcal{L}) + \frac{t^2}{2}\mathrm{tr}(\mathcal{L}^2) + \ldots$

   Second term is degree distribution; third is weighted triangle count

2. Compute top + bottom eigenvalues, approximate the rest

   Linear extrapolation = explicit assumption on the manifold (Weyl's law)

Other spectrum approximators can be even more efficient!

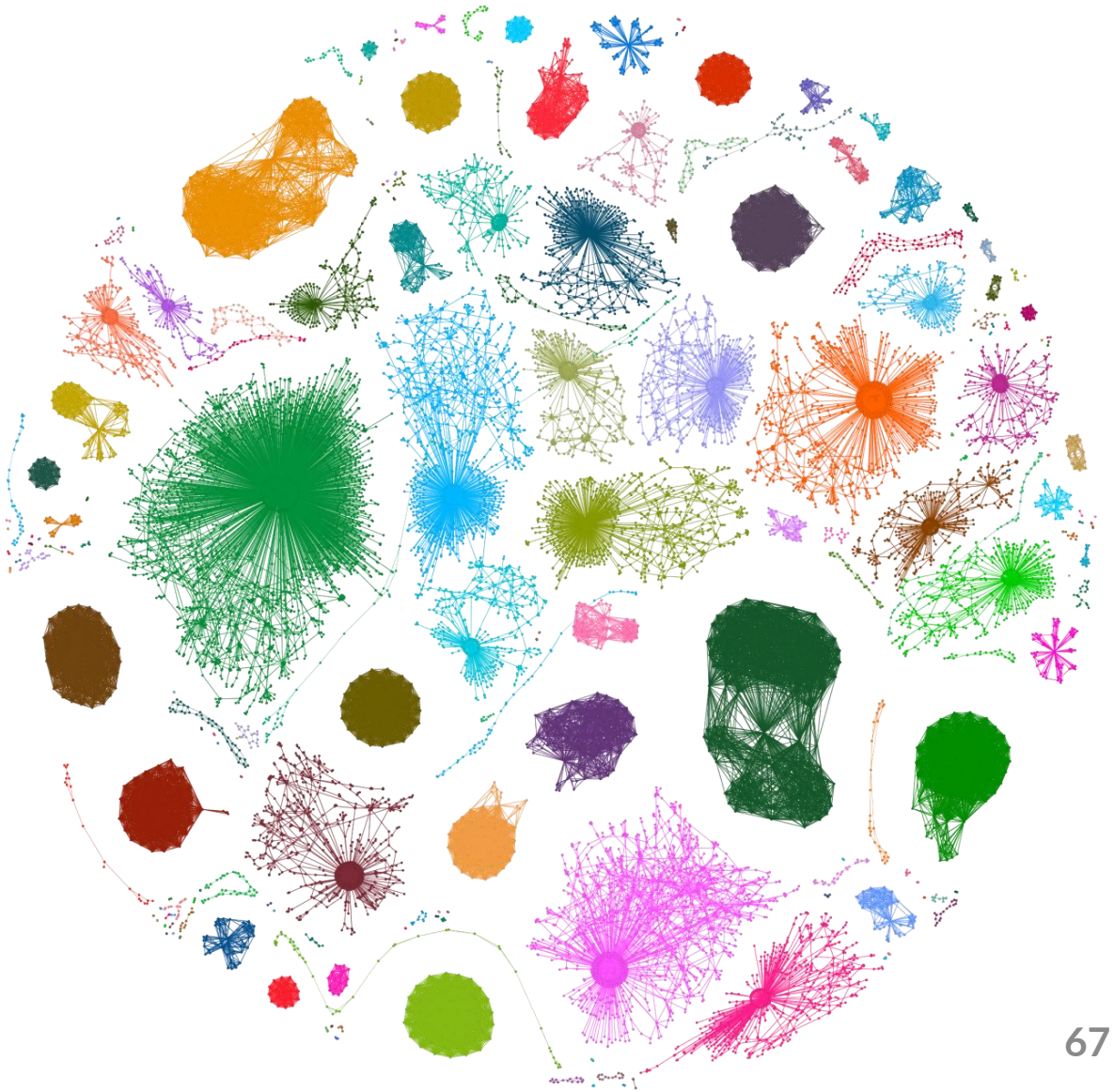
[Cohen-Steiner et al. | KDD 2018]

[Adams et al. | arXiv 1802.03451]

Experimental design

3 key properties:

- Permutation invariance
- Scale-adaptivity
- Size invariance



Detecting graphs with communities

3 key properties:

- Permutation invariance
- Scale-adaptivity
- Size invariance

NetLSD

NIPS'17
ASONAM'13

	$n \sim \mathcal{P}(\lambda)$				
Method	64	128	256	512	1024
$h(G)$	54.39	59.01	60.82	57.99	53.80
$h(G)/h(\bar{K})$	54.53	62.27	70.83	76.45	78.40
$w(G)$	56.23	63.77	69.57	71.66	70.34
$w(G)/w(\bar{K})$	55.51	63.85	72.12	77.59	79.39
FGSD	55.44	54.99	53.86	52.74	50.92
NetSimile	59.55	56.57	59.41	66.23	60.58

Accuracy of classification of SBM vs Erdős–Rényi graphs

Detecting rewired graphs

3 key properties:

- Permutation invariance
- Scale-adaptivity
- Size invariance

NetLSD

NIPS'17
ASONAM'13

	dataset					
Method	MUTAG	PROTEINS	NCI1	ENZYMES	COLLAB	IMDB-B
$h(G)$	76.03	91.81	69.74	92.51	59.82	67.18
$h(G)/h(\bar{K})$	79.12	94.90	74.55	95.20	65.85	70.58
$w(G)$	78.18	93.04	70.54	94.03	69.01	75.26
$w(G)/w(\bar{K})$	79.72	89.00	74.14	90.77	70.35	75.54
FGSD	77.79	60.11	64.08	53.93	55.18	56.23
NetSimile	77.11	85.73	58.58	87.38	54.43	54.44

Accuracy of classification of real vs rewired graphs

Classifying real graphs

3 key properties:

- Permutation invariance
- Scale-adaptivity
- Size invariance

NetLSD

NIPS'17
ASONAM'13

	dataset					
Method	MUTAG	PROTEINS	NCI1	ENZYMES	COLLAB	IMDB-B
$h(G)$	86.47	64.89	66.49	31.99	68.00	68.04
$h(G)/h(\bar{K})$	85.32	65.73	67.44	33.31	69.42	70.17
$w(G)$	83.35	66.80	70.78	40.41	75.77	68.63
$w(G)/w(\bar{K})$	81.72	65.58	67.67	35.78	77.24	69.33
FGSD	84.90	65.30	75.77	41.58	73.96	69.54
NETSIMILE	84.09	62.45	66.56	33.23	73.10	69.20

Accuracy of graph classification

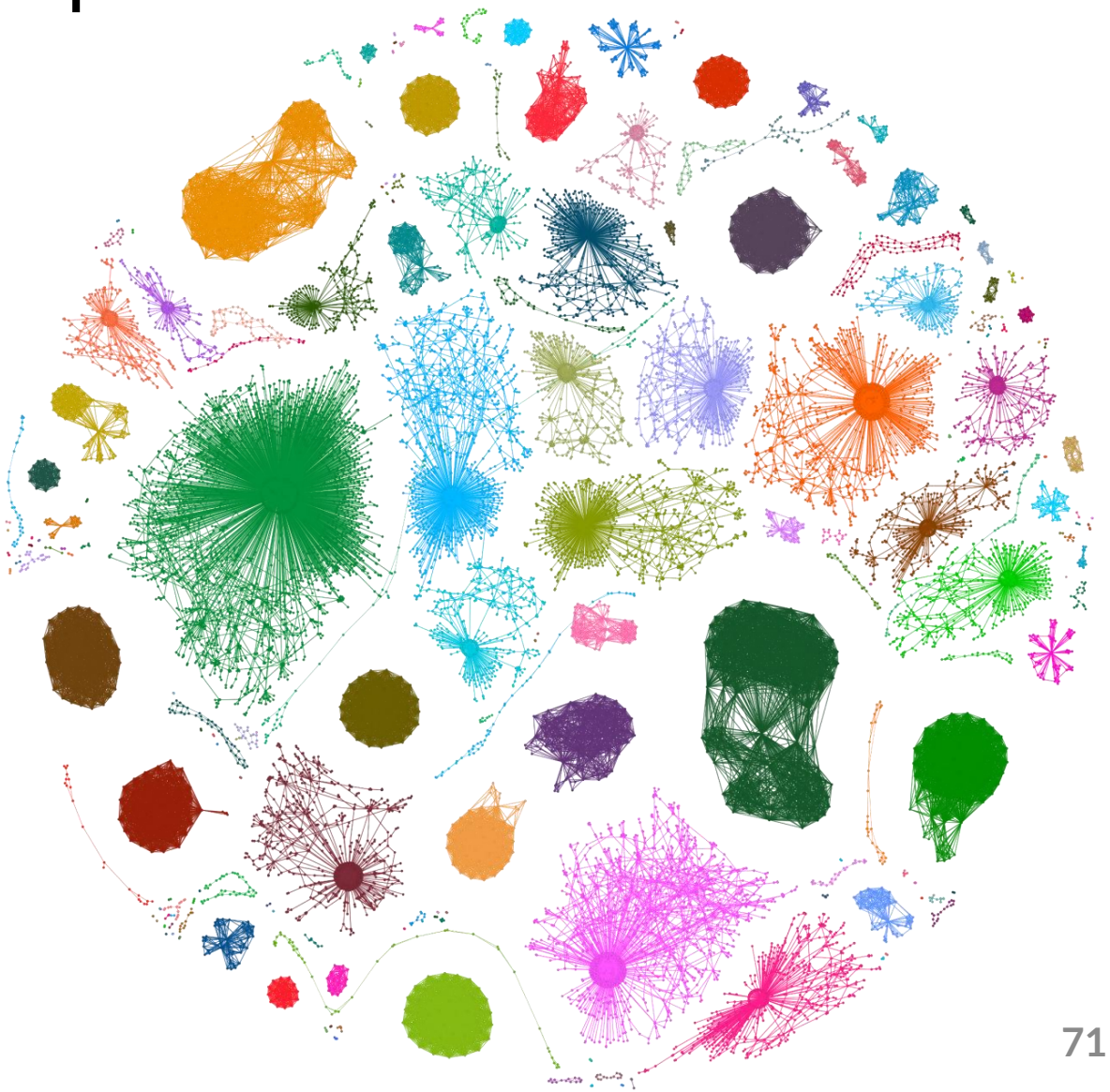
Expressive graph comparison

3 key properties:

- Permutation invariance
- Scale-adaptivity
- Size invariance

+ Scalability

= **NetLSD**



Questions?

code + data	github.com/xgfs	
website	tsitsul.in	← presentation will be there
write me	anton@tsitsul.in	

Network Laplacian Spectral Descriptors: wave kernel trace

$$w_t = \mathrm{tr}(W_t) = \sum_j e^{-it\lambda_j}$$

We sample t logarithmically, and compare $\mathrm{Re}(w_t)$ with L_2 distance
w_t detects symmetries!
≈quantum random walks

Hearing the Shape of a Graph

"Can One Hear the Shape of a Drum?" – Kac 1966

No, as there are co-spectral drums (graphs)

Conjecture: # of co-spectral graphs $\to 0$ as # of nodes $\to \infty$

[Dufree, Martin 2015]