Learning signals defined on graphs with optimal transport and Gaussian process regression

Raphaël Carpintero Perez

Sébastien Da Veiga
Josselin Garnier
Brian Staber

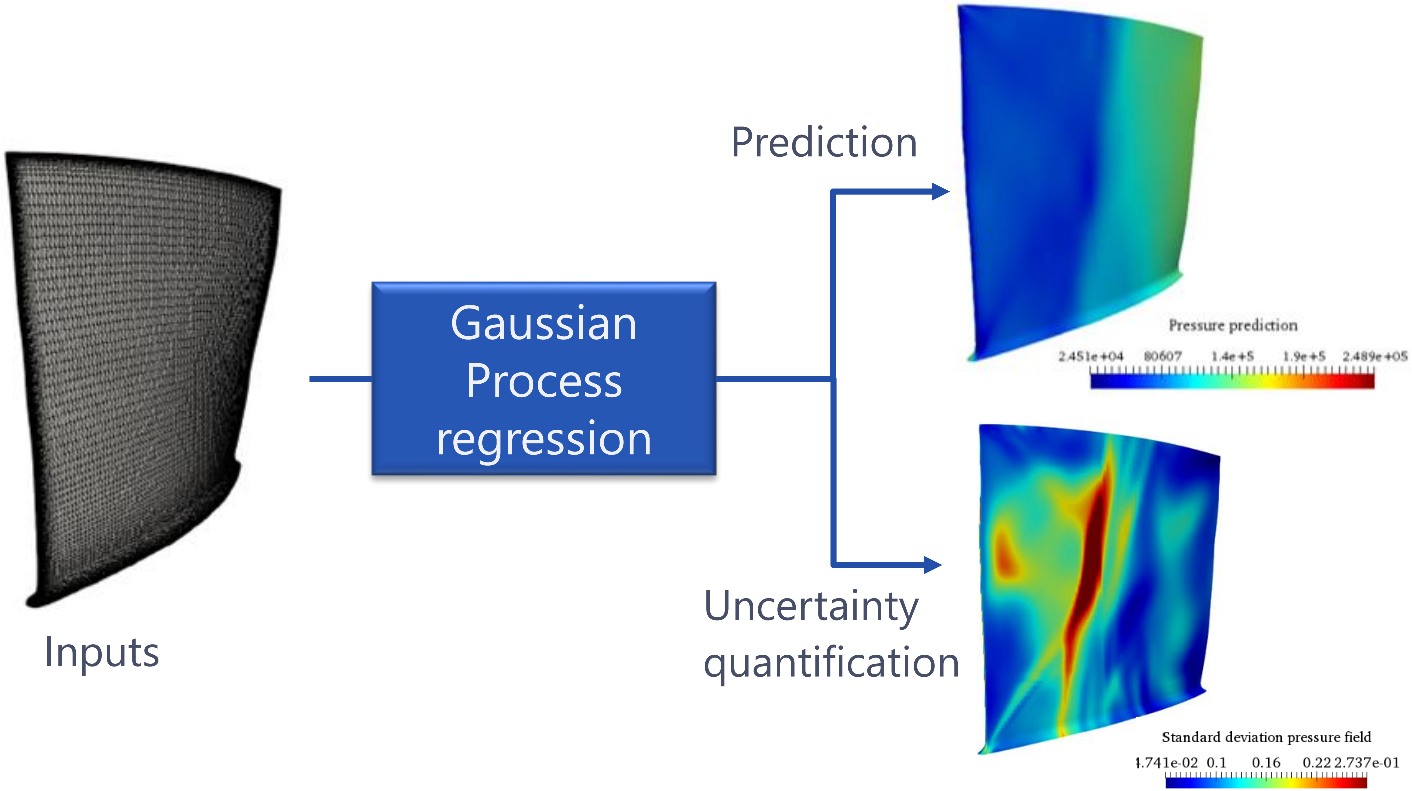
10/12/2024

SAFRAN

CMAP

Objectives

Objectives



Inputs

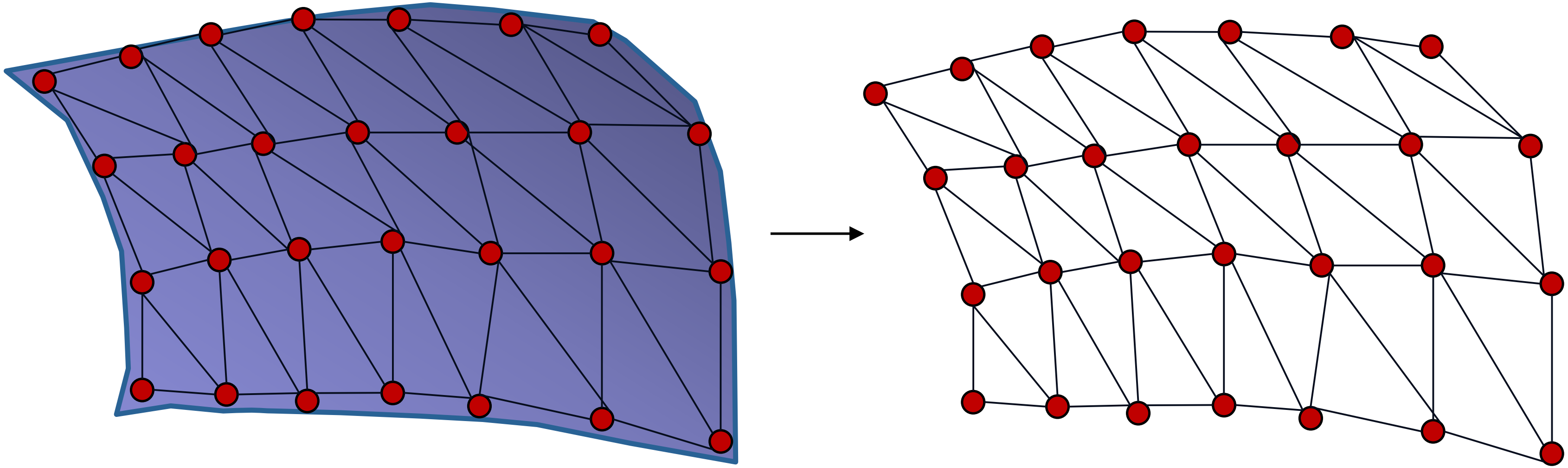
Gaussian Process regression

Prediction

Uncertainty quantification

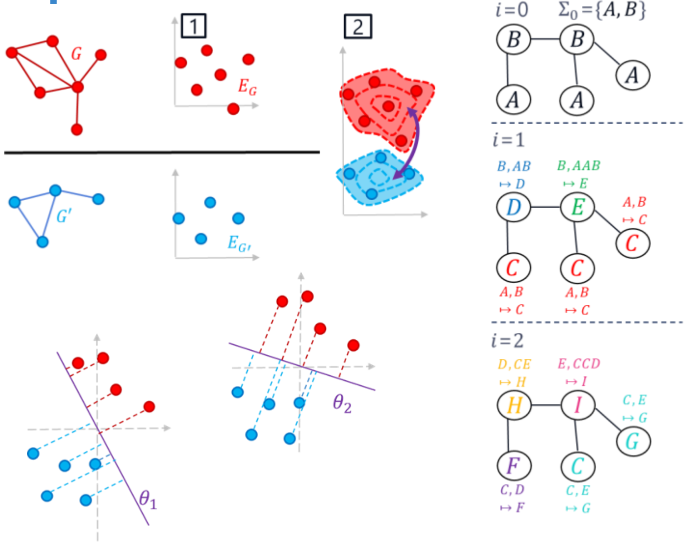
Inputs and outputs

- **Graph inputs**
  - Mesh → Graph structure
  - 2D/3D coordinates for all nodes
- **Scalar inputs**
  - Pressure
  - Speed of rotation
- **Outputs**
  - Physical quantities of interest (scalars)
  - Fields



Outline

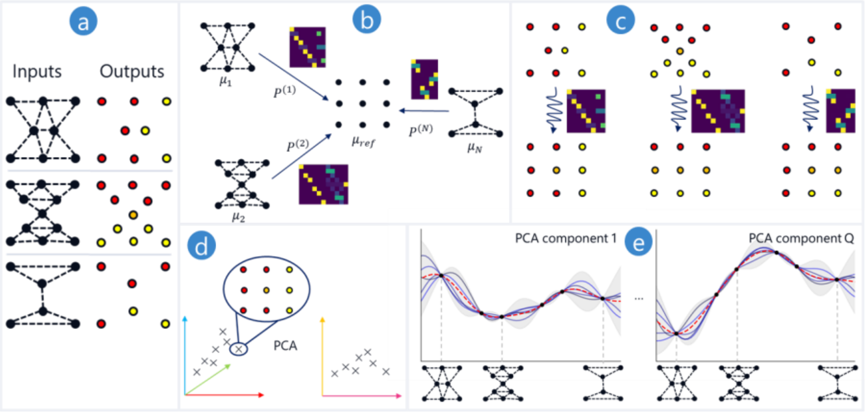
- **Gaussian process regression for graph inputs**



Gaussian process regression with Sliced Wasserstein Weisfeiler Lehman graph kernels

[CP, Da Veiga, Garnier, Staber, 2024]

- **Prediction of output fields**



Learning signals defined on graphs with optimal transport and Gaussian process regression, 2024

[CP, Da Veiga, Garnier, Staber, 2024+]

Gaussian process regression for graph inputs

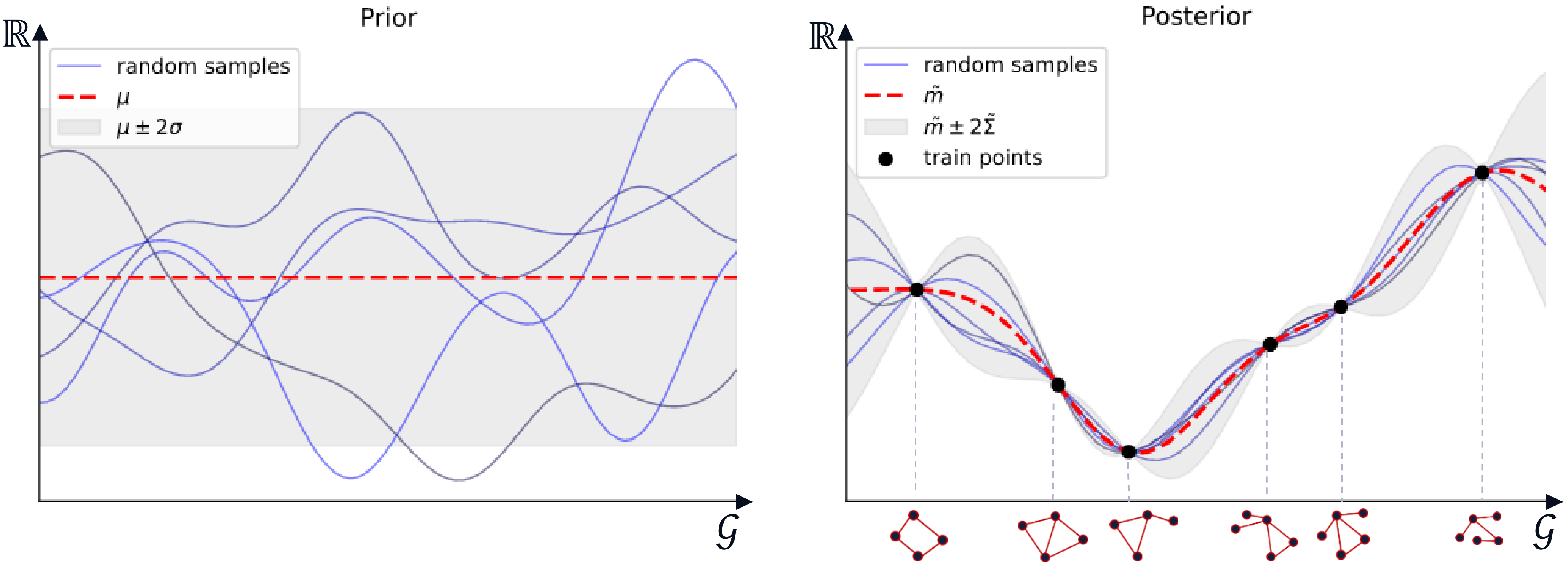
1- Problem statement

2- Classical approaches

3- SWWL kernel

4- Experiments

Gaussian process regression



Gaussian process regression

Noisy observations:
$\mathbf{y} = (y_i)_{i=1}^{N}$ with $\mathrm{y_i} = f(G_i) + \epsilon_i$ where
$\epsilon_i \sim \mathcal{N}(0, \sigma^2)$, $f: \mathcal{X} \to \mathbb{R}$

Gaussian prior over functions:

$f \sim \mathcal{GP}(0, k)$ where $k: \mathcal{G} \times \mathcal{G} \to \mathbb{R}$ is a symmetric **positive definite kernel**

- $\mathcal{X} = \mathcal{G}$ is a set of graphs.
-
- How to choose k ?

k (,) = ?

Test locations:
$\boldsymbol{G}^* = (G_i^*)_{i=1}^{N^*}$
Predictions? $\boldsymbol{f}_* = (f(G_i^*))_{i=1}^{N^*}$?

$\boldsymbol{K}, \boldsymbol{K}_{**}, \boldsymbol{K}_*$: train, test, train/test Gram matrices

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \boldsymbol{K} + \sigma^2 I & \boldsymbol{K}_*^T \\ \boldsymbol{K}_* & \boldsymbol{K}_{**} \end{bmatrix}\right)$$

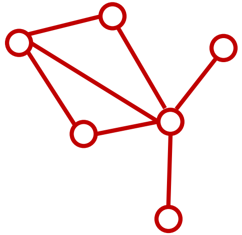
Posterior distribution:

$$\boldsymbol{f}_* \mid \boldsymbol{G}, \boldsymbol{y}, \boldsymbol{G}^* \sim \mathcal{N}(\overline{\boldsymbol{m}}, \overline{\boldsymbol{\Sigma}})$$

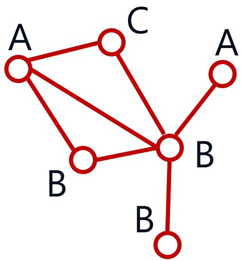
predictive mean $\overline{\boldsymbol{m}} = \boldsymbol{K}_*(\boldsymbol{K} + \sigma^2 I)^{-1}\boldsymbol{y}$

uncertainties $\overline{\boldsymbol{\Sigma}} = \boldsymbol{K}_{**} - \boldsymbol{K}_*(\boldsymbol{K} + \sigma^2 I)^{-1}\boldsymbol{K}_*^T$

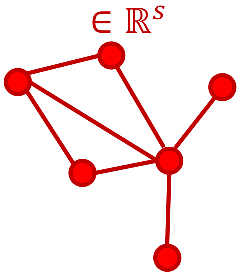
What is a graph ?



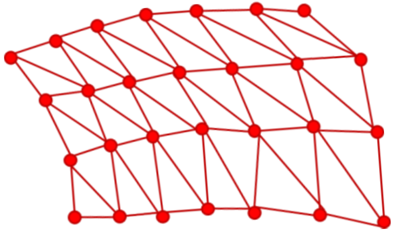
Case 1 :
Vertices + Edges



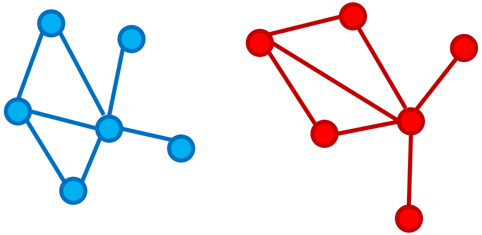
Case 2 :
Vertices + Edges
+ Node labels



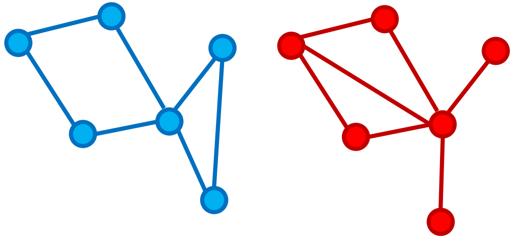
Case 3 :
Vertices + Edges
+ Node attributes



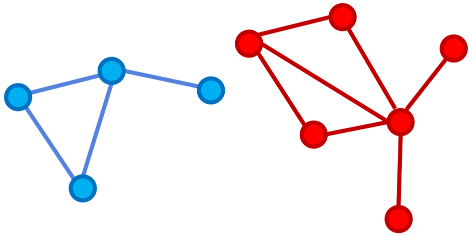
Case 3C+: Varying number of
nodes + structure + attributes
+ large-scale + sparse



Case 3A: Fixed structure -> signal



Case 3B: Fixed number of nodes



Case 3C: Varying number of
nodes + structure + attributes

Gaussian process regression for graph inputs

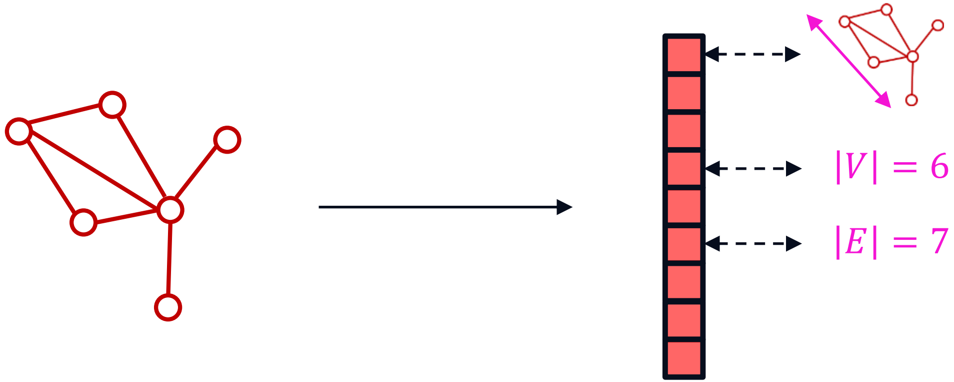
1- Problem statement

2- Classical approaches

3- SWWL kernel

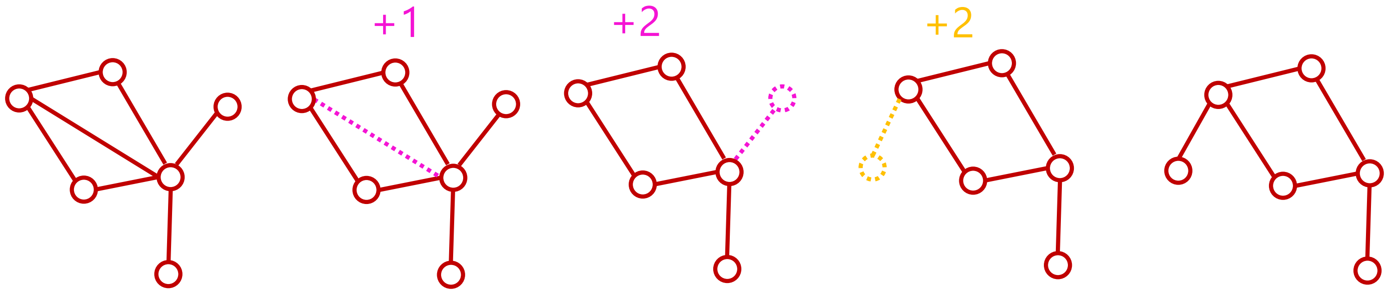
4- Experiments

Invariants / Topological descriptors



- Vectorial representation using quantities invariant to graph isomorphism (diameter, average clustering coefficient, …)
- Complete invariants require exponential time

Graph edit distance



- $d(G_1, G_2)$: minimal number of operations to transfrom G_1 in G_2 (adding/removing an edge/vertex, node relabeling)

- NP-complete
- Not suited for node-attributed graphs…

Taxonomy of graph kernels

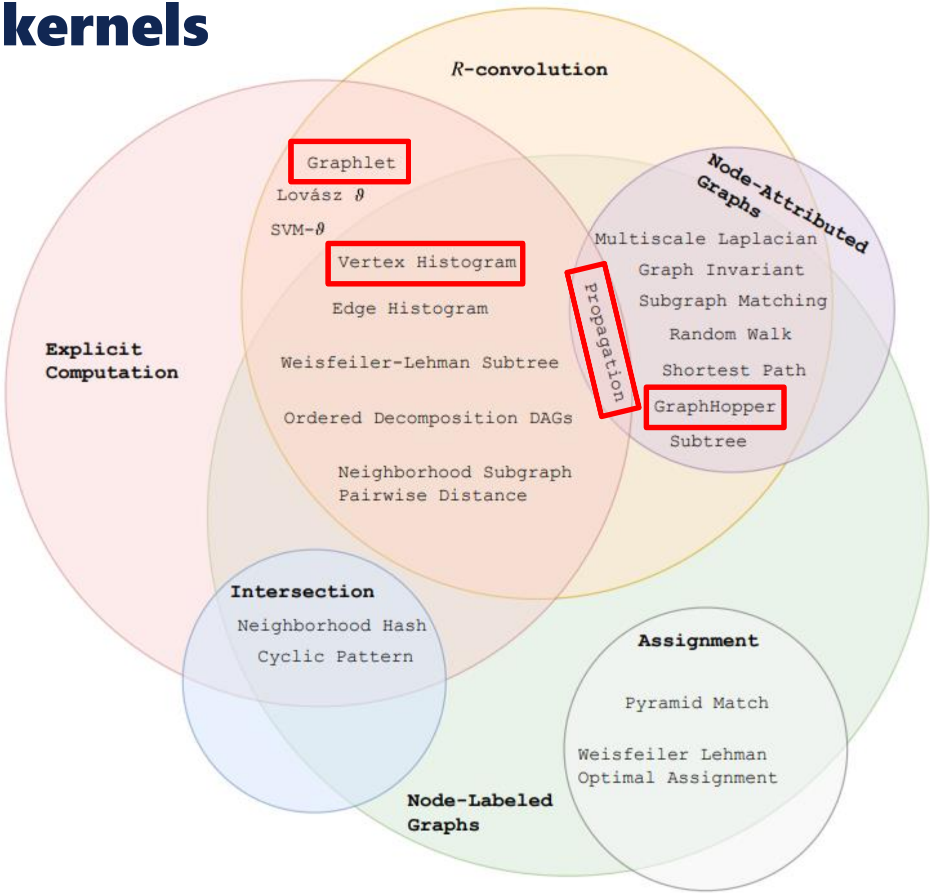
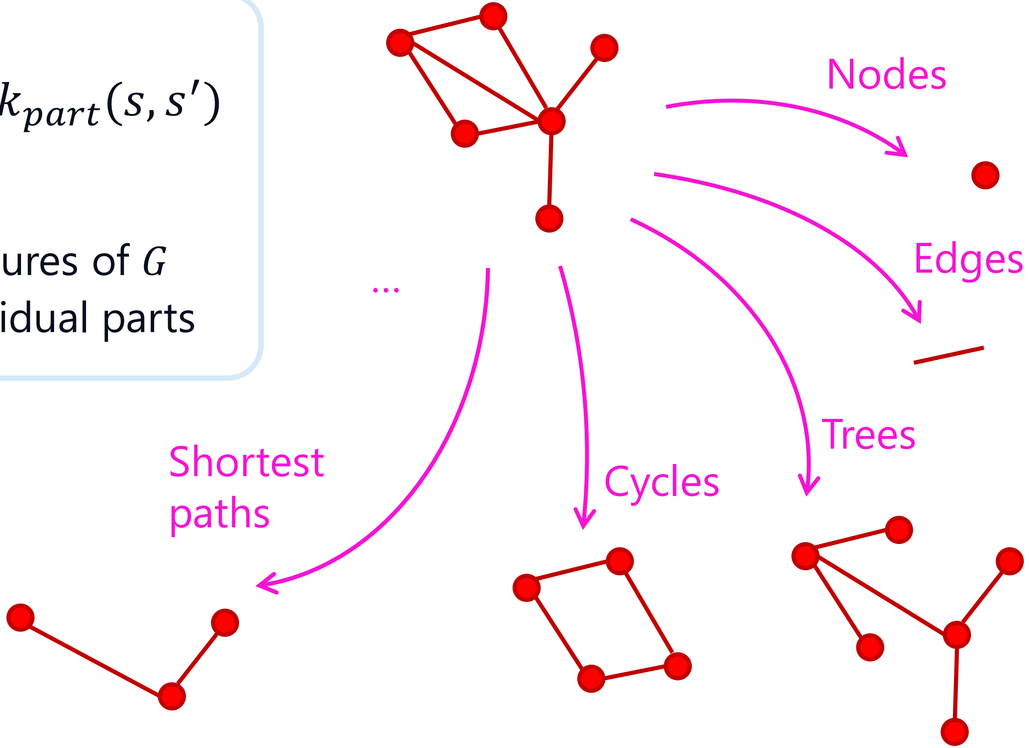


Figure from [Nikolentzos et al., 2021]

$\mathcal{R}$-convolution kernels

$$k(G,G') \coloneqq \sum_{s\in\mathcal{S}(G)} \sum_{s'\in\mathcal{S}(G')} k_{part}(s,s')$$

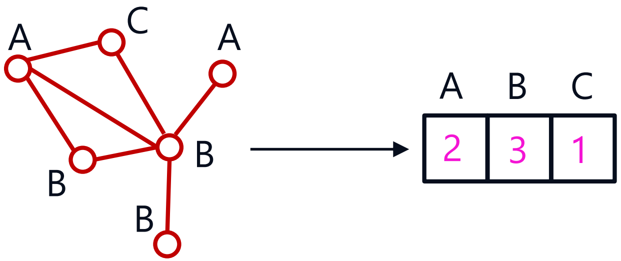
- $S(G)$: set of parts/substructures of G
- k_{part}: kernel between individual parts



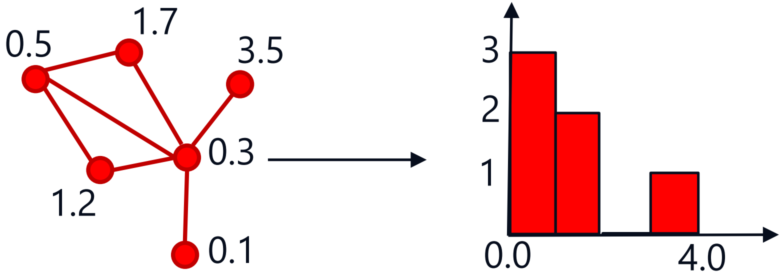
All node-pairs kernel / node histogram kernel

$$k \quad (G,G') := \sum_{\mathrm{v}\in V}\sum_{\mathrm{v}'\in V'} \mathrm{k}_{\mathrm{node}}(\mathrm{v},\mathrm{v}')$$

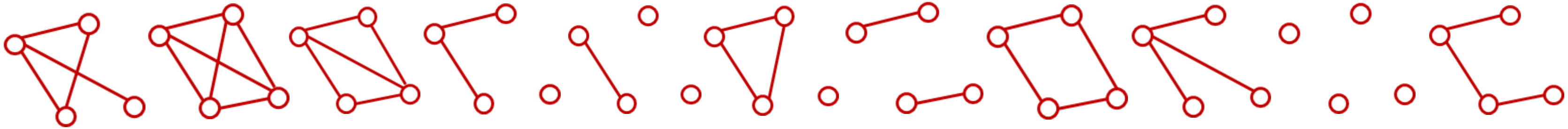
- k_{node} : Dirac kernel ⇒ ϕ_N : unnormalized histogram
- Continuous variant with binning

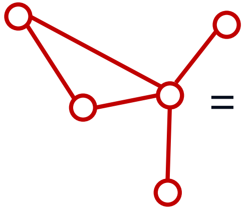


A	B	C
2	3	1



Graphlet kernel





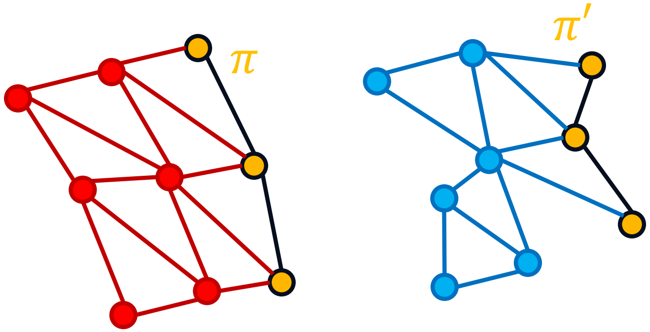
=

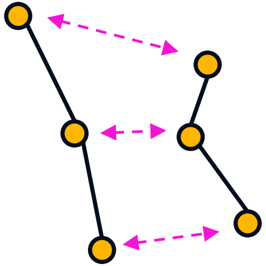
2	0	0	0	1	0	0	0	2	0	0

- Set of k-graphlets of size N_k, $k \geq 3$
- $\phi_{GL}(G)$: vector of the frequencies of all graphlets in G (k-spectrum)
- $k(G, G') \coloneqq \phi_{GL}(G)\phi_{GL}(G')^T$
- Does not take into account labels or attributes

Graph Hopper

[Feragen et al., 2013]



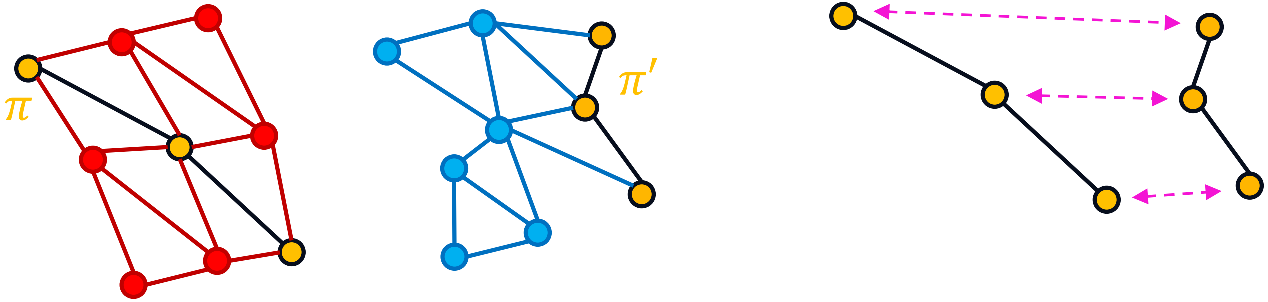


$$k(G,G') \coloneqq \sum_{\pi\in\mathcal{P},\pi'\in\mathcal{P}'} k_p(\pi,\pi') \ \ with \ k_p(\pi,\pi') \coloneqq \begin{cases} \sum_{j=1}^{|\pi|} RBF(\pi_j,\pi'_j) \ if \ |\pi| = |\pi'| \\ 0 \qquad\qquad\qquad\qquad otherwise \end{cases}$$

- $\mathcal{P}$: set of all shortest paths in G, $|\pi|$: discrete length of the path $\pi = (\pi_1, \cdots, \pi_{|\pi|})$

Graph Hopper

[Feragen et al., 2013]

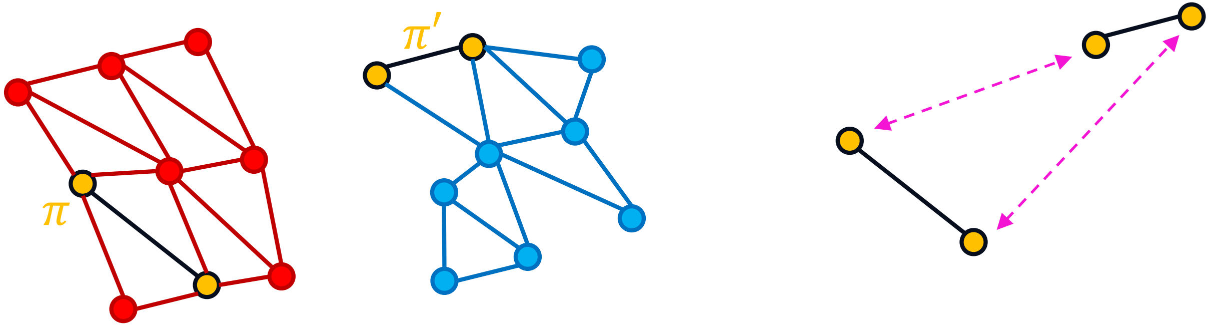


$$k(G,G') \coloneqq \sum_{\pi\in\mathcal{P},\pi'\in\mathcal{P}'} k_p(\pi,\pi') \ \ with\ k_p(\pi,\pi') \coloneqq \begin{cases} \sum_{j=1}^{|\pi|} RBF(\pi_j,\pi'_j)\ \ if\ |\pi| = |\pi'| \\ 0 \qquad\qquad\qquad\qquad otherwise \end{cases}$$

- $\mathcal{P}$: set of all shortest paths in G, $|\pi|$: discrete length of the path $\pi = (\pi_1,\cdots,\pi_{|\pi|})$

Graph Hopper

[Feragen et al., 2013]



$$k(G,G') \coloneqq \sum_{\pi\in\mathcal{P},\pi'\in\mathcal{P}'} k_p(\pi,\pi') \;\; with\; k_p(\pi,\pi') \coloneqq \begin{cases} \sum_{j=1}^{|\pi|} RBF(\pi_j,\pi'_j) \;\; if\; |\pi| = |\pi'| \\ 0 \qquad\qquad\qquad\qquad otherwise \end{cases}$$

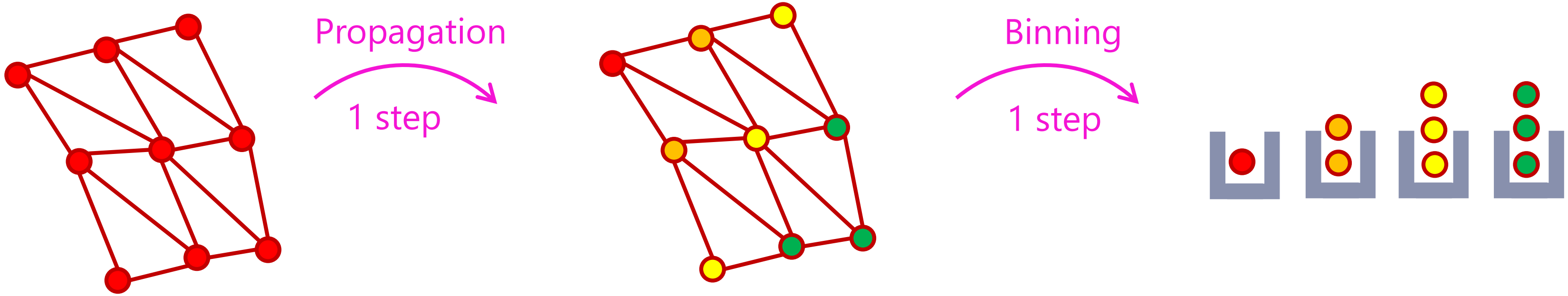
- $\mathcal{P}$: set of all shortest paths in G, $|\pi|$: discrete length of the path $\pi = (\pi_1, \cdots, \pi_{|\pi|})$

Propagation kernel

- $k(G,G') = \sum_t^T k_t(G,G')$ T iterations

- <u>Binning + counting</u>: $k_t(G,G') = \sum_{u \in G} \sum_{u' \in G'} \delta(\, h(F_t),\ h(F'_t))$
  δ: Kronecker, h: Locally Sensitive Hashing function

- <u>Propagation</u>: $F_t = P\ F_t$ where P is a transition matrix (e.g. $P = Diag\left(\sum_j A_{1j}, \cdots, \sum_j A_{nj}\right)^{-1} A$)



Propagation

1 step

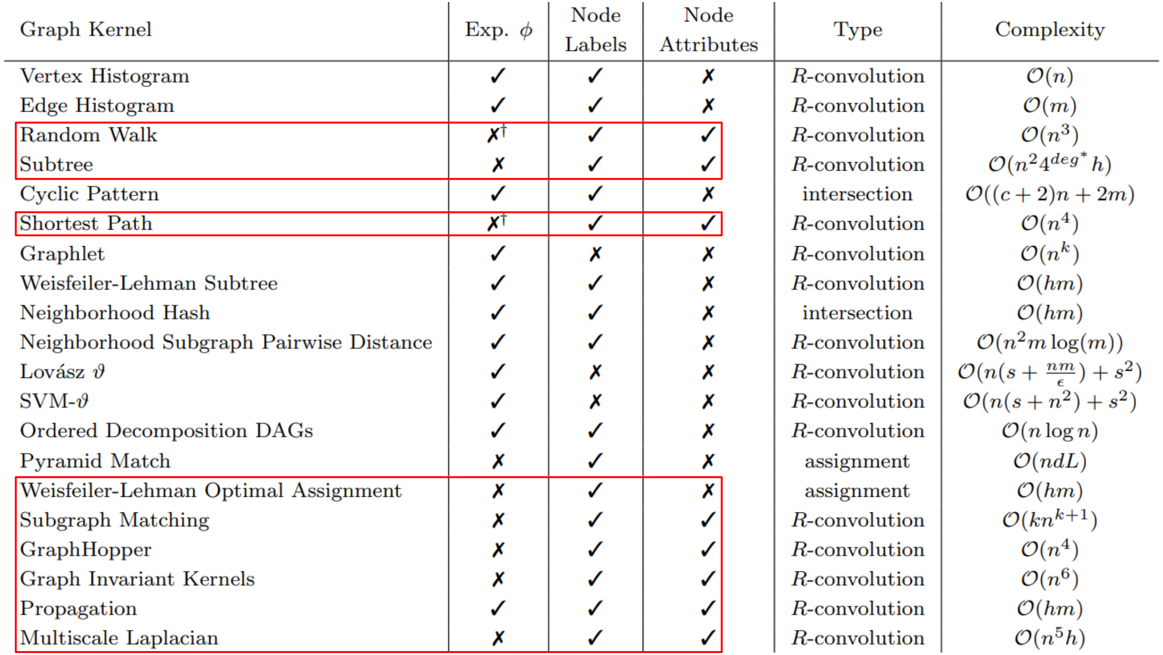
Binning

1 step

Graph kernels

Checklist:

✓ continuous node attributes



Graph Kernel	Exp. ϕ	Node Labels	Node Attributes	Type	Complexity
Vertex Histogram	✓	✓	✗	R-convolution	$\mathcal{O}(n)$
Edge Histogram	✓	✓	✗	R-convolution	$\mathcal{O}(m)$
Random Walk	✗†	✓	✓	R-convolution	$\mathcal{O}(n^3)$
Subtree	✗	✓	✓	R-convolution	$\mathcal{O}(n^2 4^{deg^*} h)$
Cyclic Pattern	✓	✓	✗	intersection	$\mathcal{O}((c+2)n+2m)$
Shortest Path	✗†	✓	✓	R-convolution	$\mathcal{O}(n^4)$
Graphlet	✓	✗	✗	R-convolution	$\mathcal{O}(n^k)$
Weisfeiler-Lehman Subtree	✓	✓	✗	R-convolution	$\mathcal{O}(hm)$
Neighborhood Hash	✓	✓	✗	intersection	$\mathcal{O}(hm)$
Neighborhood Subgraph Pairwise Distance	✓	✓	✗	R-convolution	$\mathcal{O}(n^2 m \log(m))$
Lovász ϑ	✓	✗	✗	R-convolution	$\mathcal{O}(n(s+\frac{nm}{\epsilon})+s^2)$
SVM-ϑ	✓	✗	✗	R-convolution	$\mathcal{O}(n(s+n^2)+s^2)$
Ordered Decomposition DAGs	✓	✓	✗	R-convolution	$\mathcal{O}(n \log n)$
Pyramid Match	✗	✓	✗	assignment	$\mathcal{O}(ndL)$
Weisfeiler-Lehman Optimal Assignment	✗	✓	✗	assignment	$\mathcal{O}(hm)$
Subgraph Matching	✗	✓	✓	R-convolution	$\mathcal{O}(kn^{k+1})$
GraphHopper	✗	✓	✓	R-convolution	$\mathcal{O}(n^4)$
Graph Invariant Kernels	✗	✓	✓	R-convolution	$\mathcal{O}(n^6)$
Propagation	✓	✓	✓	R-convolution	$\mathcal{O}(hm)$
Multiscale Laplacian	✗	✓	✓	R-convolution	$\mathcal{O}(n^5 h)$

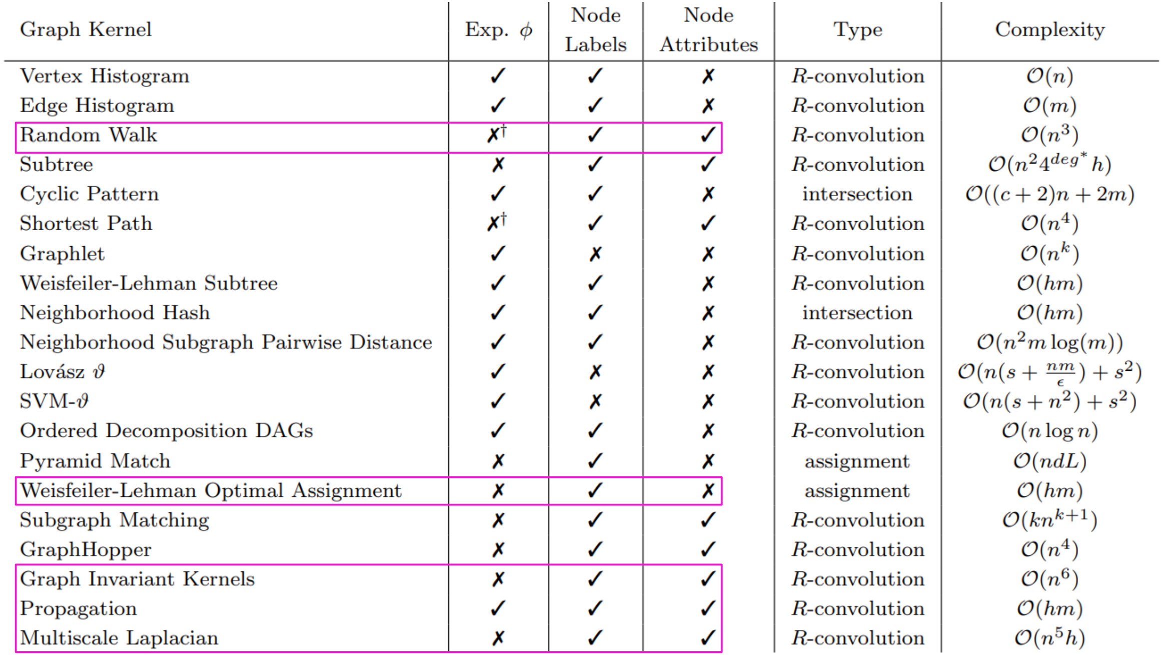
[Nikolentzos et al., 2021]

Graph kernels

Checklist:

✓ continuous node attributes

✓ no relying heavily on the graph structure



Graph Kernel	Exp. ϕ	Node Labels	Node Attributes	Type	Complexity
Vertex Histogram	✓	✓	✗	R-convolution	$\mathcal{O}(n)$
Edge Histogram	✓	✓	✗	R-convolution	$\mathcal{O}(m)$
Random Walk	✗†	✓	✓	R-convolution	$\mathcal{O}(n^3)$
Subtree	✗	✓	✓	R-convolution	$\mathcal{O}(n^2 4^{deg^*} h)$
Cyclic Pattern	✓	✓	✗	intersection	$\mathcal{O}((c+2)n+2m)$
Shortest Path	✗†	✓	✓	R-convolution	$\mathcal{O}(n^4)$
Graphlet	✓	✗	✗	R-convolution	$\mathcal{O}(n^k)$
Weisfeiler-Lehman Subtree	✓	✓	✗	R-convolution	$\mathcal{O}(hm)$
Neighborhood Hash	✓	✓	✗	intersection	$\mathcal{O}(hm)$
Neighborhood Subgraph Pairwise Distance	✓	✓	✗	R-convolution	$\mathcal{O}(n^2 m \log(m))$
Lovász ϑ	✓	✗	✗	R-convolution	$\mathcal{O}(n(s+\frac{nm}{\epsilon})+s^2)$
SVM-ϑ	✓	✗	✗	R-convolution	$\mathcal{O}(n(s+n^2)+s^2)$
Ordered Decomposition DAGs	✓	✓	✗	R-convolution	$\mathcal{O}(n \log n)$
Pyramid Match	✗	✓	✗	assignment	$\mathcal{O}(ndL)$
Weisfeiler-Lehman Optimal Assignment	✗	✓	✗	assignment	$\mathcal{O}(hm)$
Subgraph Matching	✗	✓	✓	R-convolution	$\mathcal{O}(kn^{k+1})$
GraphHopper	✗	✓	✓	R-convolution	$\mathcal{O}(n^4)$
Graph Invariant Kernels	✗	✓	✓	R-convolution	$\mathcal{O}(n^6)$
Propagation	✓	✓	✓	R-convolution	$\mathcal{O}(hm)$
Multiscale Laplacian	✗	✓	✓	R-convolution	$\mathcal{O}(n^5 h)$

[Nikolentzos et al., 2021]

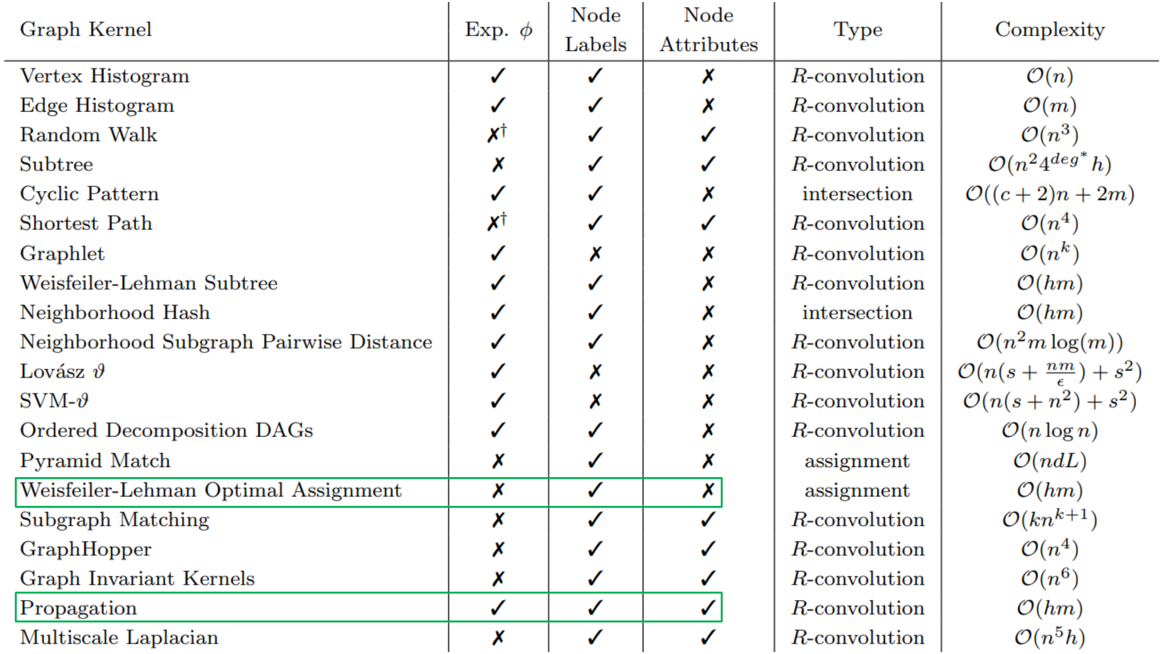
Graph kernels

Checklist:

✓ continuous node attributes

✓ no relying heavily on the graph structure

✓ tractable



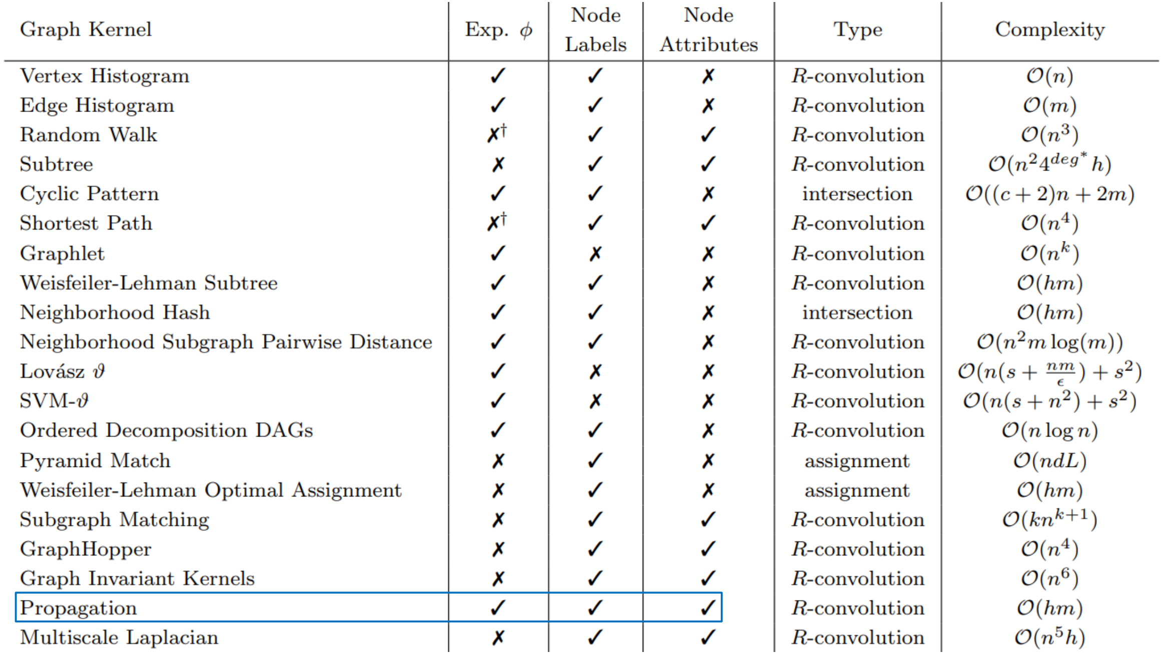
Graph Kernel	Exp. ϕ	Node Labels	Node Attributes	Type	Complexity
Vertex Histogram	✓	✓	✗	R-convolution	$\mathcal{O}(n)$
Edge Histogram	✓	✓	✗	R-convolution	$\mathcal{O}(m)$
Random Walk	✗†	✓	✓	R-convolution	$\mathcal{O}(n^3)$
Subtree	✗	✓	✓	R-convolution	$\mathcal{O}(n^2 4^{deg^*} h)$
Cyclic Pattern	✓	✓	✗	intersection	$\mathcal{O}((c+2)n+2m)$
Shortest Path	✗†	✓	✓	R-convolution	$\mathcal{O}(n^4)$
Graphlet	✓	✗	✗	R-convolution	$\mathcal{O}(n^k)$
Weisfeiler-Lehman Subtree	✓	✓	✗	R-convolution	$\mathcal{O}(hm)$
Neighborhood Hash	✓	✓	✗	intersection	$\mathcal{O}(hm)$
Neighborhood Subgraph Pairwise Distance	✓	✓	✗	R-convolution	$\mathcal{O}(n^2 m \log(m))$
Lovász ϑ	✓	✗	✗	R-convolution	$\mathcal{O}(n(s+\frac{nm}{\epsilon})+s^2)$
SVM-ϑ	✓	✗	✗	R-convolution	$\mathcal{O}(n(s+n^2)+s^2)$
Ordered Decomposition DAGs	✓	✓	✗	R-convolution	$\mathcal{O}(n \log n)$
Pyramid Match	✗	✓	✗	assignment	$\mathcal{O}(ndL)$
Weisfeiler-Lehman Optimal Assignment	✗	✓	✗	assignment	$\mathcal{O}(hm)$
Subgraph Matching	✗	✓	✓	R-convolution	$\mathcal{O}(kn^{k+1})$
GraphHopper	✗	✓	✓	R-convolution	$\mathcal{O}(n^4)$
Graph Invariant Kernels	✗	✓	✓	R-convolution	$\mathcal{O}(n^6)$
Propagation	✓	✓	✓	R-convolution	$\mathcal{O}(hm)$
Multiscale Laplacian	✗	✓	✓	R-convolution	$\mathcal{O}(n^5 h)$

[Nikolentzos et al., 2021]

Graph kernels

Checklist:

- ✓ continuous node attributes
- ✓ no relying heavily on the graph structure
- ✓ tractable
- ✓ positive definite



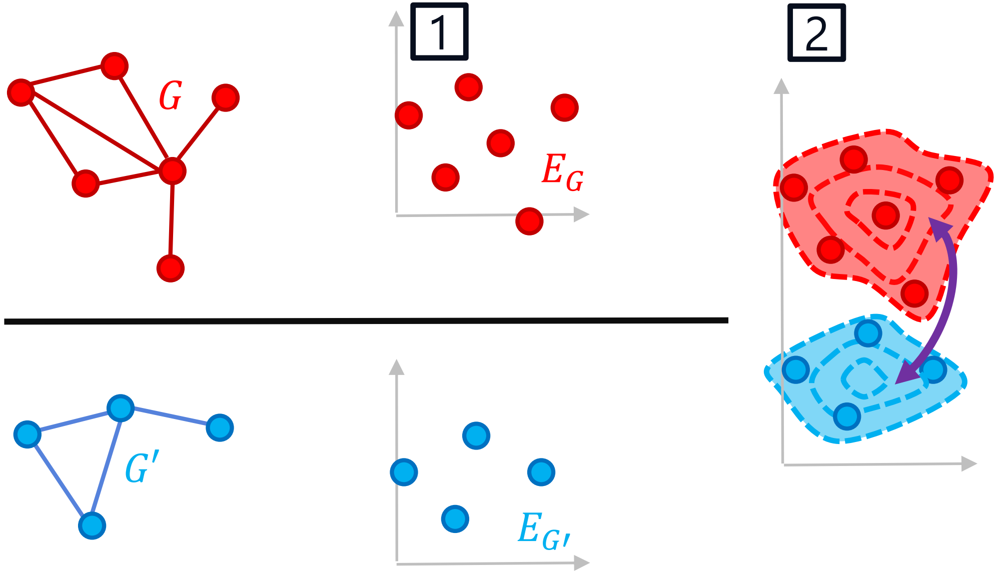
Graph Kernel	Exp. ϕ	Node Labels	Node Attributes	Type	Complexity
Vertex Histogram	✓	✓	✗	R-convolution	$\mathcal{O}(n)$
Edge Histogram	✓	✓	✗	R-convolution	$\mathcal{O}(m)$
Random Walk	✗†	✓	✓	R-convolution	$\mathcal{O}(n^3)$
Subtree	✗	✓	✓	R-convolution	$\mathcal{O}(n^2 4^{deg^*} h)$
Cyclic Pattern	✓	✓	✗	intersection	$\mathcal{O}((c+2)n+2m)$
Shortest Path	✗†	✓	✓	R-convolution	$\mathcal{O}(n^4)$
Graphlet	✓	✗	✗	R-convolution	$\mathcal{O}(n^k)$
Weisfeiler-Lehman Subtree	✓	✓	✗	R-convolution	$\mathcal{O}(hm)$
Neighborhood Hash	✓	✓	✗	intersection	$\mathcal{O}(hm)$
Neighborhood Subgraph Pairwise Distance	✓	✓	✗	R-convolution	$\mathcal{O}(n^2 m \log(m))$
Lovász ϑ	✓	✗	✗	R-convolution	$\mathcal{O}(n(s+\frac{nm}{\epsilon})+s^2)$
SVM-ϑ	✓	✗	✗	R-convolution	$\mathcal{O}(n(s+n^2)+s^2)$
Ordered Decomposition DAGs	✓	✓	✗	R-convolution	$\mathcal{O}(n \log n)$
Pyramid Match	✗	✓	✗	assignment	$\mathcal{O}(ndL)$
Weisfeiler-Lehman Optimal Assignment	✗	✓	✗	assignment	$\mathcal{O}(hm)$
Subgraph Matching	✗	✓	✓	R-convolution	$\mathcal{O}(kn^{k+1})$
GraphHopper	✗	✓	✓	R-convolution	$\mathcal{O}(n^4)$
Graph Invariant Kernels	✗	✓	✓	R-convolution	$\mathcal{O}(n^6)$
Propagation	✓	✓	✓	R-convolution	$\mathcal{O}(hm)$
Multiscale Laplacian	✗	✓	✓	R-convolution	$\mathcal{O}(n^5 h)$

[Nikolentzos et al., 2021]

Gaussian process regression for graph inputs

1- Problem statement
2- Classical approaches
3- SWWL kernel
4- Experiments

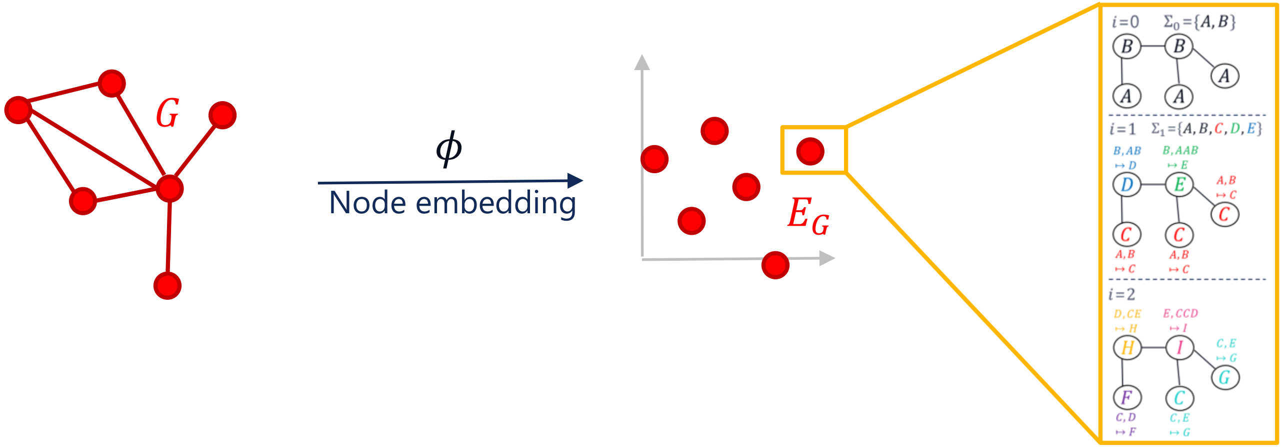
Node embeddings + Optimal transport approaches



Step 1: continuous WL embeddings

[Togninalli et al., 2019]

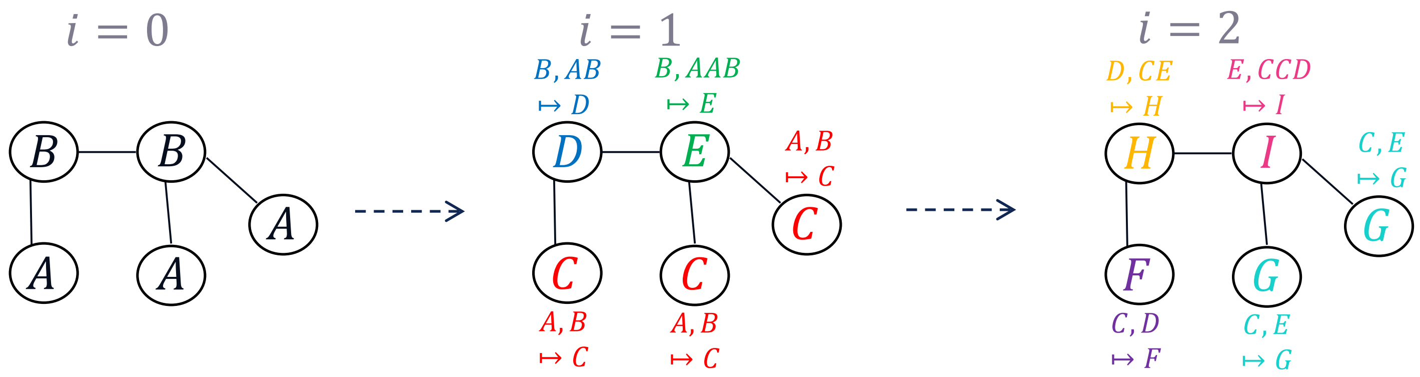
$\boxed{1}$



Weisfeiler-Lehman embeddings

Figure From [Kriege et al., 2020]

- WL relabeling (discrete case)



$\Sigma = \{A, B\}$ $\quad$ $\Sigma = \{A, B, C, D, E\}$ $\quad$ $\Sigma = \{A, B, C, D, E, F, G, H, I\}$

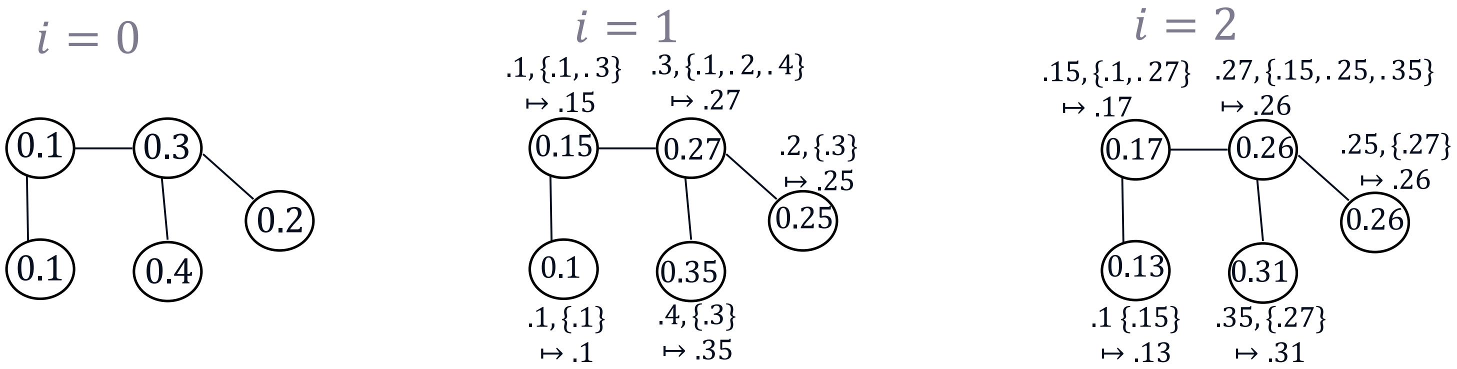
$$l^{(i+1)}(v) = Hash\left(l^i(v), \{l^i(u)\,, u \in \mathcal{N}(v)\}\right)$$

$$X_G^{(i)} = \left[l^{(i)}(v), v \in V_G\right] \qquad X_G = Concatenate(X_G^{(0)}, \cdots, X_G^{(H)})$$

Continuous Weisfeiler-Lehman embeddings

[Togninalli et al., 2019]

- WL relabeling (continuous case)

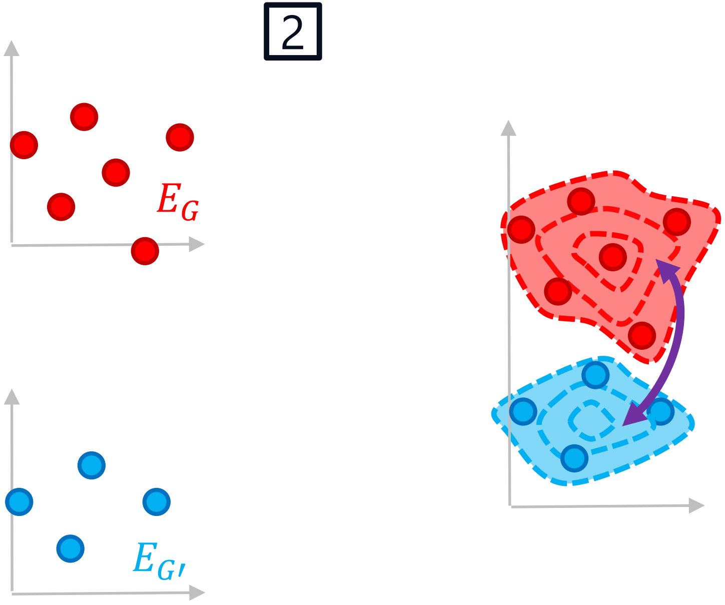


$i = 0$
0.1
0.3
0.2
0.1
0.4
$i = 1$
$.1, \{.1, .3\} \mapsto .15$
$.3, \{.1, .2, .4\} \mapsto .27$
$.2, \{.3\} \mapsto .25$
0.15
0.27
0.25
0.1
0.35
$.1, \{.1\} \mapsto .1$
$.4, \{.3\} \mapsto .35$
$i = 2$
$.15, \{.1, .27\} \mapsto .17$
$.27, \{.15, .25, .35\} \mapsto .26$
$.25, \{.27\} \mapsto .26$
0.17
0.26
0.26
0.13
0.31
$.1\ \{.15\} \mapsto .13$
$.35, \{.27\} \mapsto .31$

$$a^{(i+1)}(v) = \frac{1}{2}\left(a^{(i)}(v) + \frac{1}{\deg(v)} \sum_{u \in \mathcal{N}(v)} w(v,u)\, a^{(i)}(u)\right)$$

$$X_G^{(i)} = \left[a^{(i)}(v), v \in V_G\right] \qquad X_G = Concatenate\left(X_G^{(0)}, \cdots, X_G^{(H)}\right)$$

Step 2: optimal transport



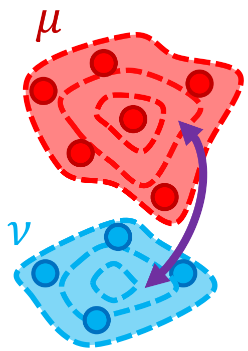
Wasserstein distance

Wasserstein distance (continuous case)

$$\mathcal{W}_r^r(\mu,\nu) = \inf_{\gamma\in\Pi(\mu,\nu)} \int_{\mathbb{R}^s\times\mathbb{R}^s} \|x-y\|^r d\gamma(x,y),$$

Where:

- $r \in [1,+\infty)$, $s \in [1,+\infty)$,
- $\mathcal{P}_r(\mathbb{R}^s)$: probability measures on $\mathbb{R}^s$ with finite moments of order r,
- $\|.\|$: Euclidean norm,
- $\Pi(\mu,\nu) = \{\pi \in \mathcal{P}_r(\mathbb{R}^s \times \mathbb{R}^s):\ (Proj_1)_{\#\pi} = \mu,\ (Proj_2)_{\#\pi} = \nu\}$



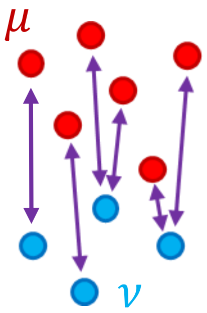
Wasserstein distance

Wasserstein distance (discrete case)

$$\mathcal{W}_r^r(\mu,\nu) = \min_{P\in U(n,n')} \langle C^{\mu,\nu}, P\rangle$$

Where:

- $\mu = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i} \qquad \nu = \frac{1}{n'}\sum_{i=1}^{n'} \delta_{y_i}$
- $U(n,n') = \left\{P \in \mathbb{R}_+^{n\times n'} : P1_{n'} = \frac{1}{n}1_n, P1_n = \frac{1}{n'}1_{n'}\right\}$
- $C^{\mu,\nu} = \left[\left\|x_i - y_j\right\|^r\right]_{i=1\ldots n,\ j=1\ldots n\prime}$



Wasserstein distance: issues

Substitution 'kernels' ✗ [Peyré, Cuturi, 2019]

$k(x,y) := k_I(\|x-y\|)$ be an isotropic kernel
→ replace $\|.\|$ by $\mathcal{W}_2$ or $\mathcal{W}_1$

$$\mathrm{k}_1(\mu,\nu) = k_I\left(\sqrt{\mathcal{W}_1\ (\mu,\nu))}\right)$$

$$\mathrm{k}_2(\mu,\nu) = k_I\left(\mathcal{W}_2\ (\mu,\nu)\right)$$

$k_1$and k_2 are not **positive definite** kernels in dimension ≥ 2.

Complexity ✗

1 pair: $O(n^3\log(n))$, Gram matrix: $O(N^2n^3\log(n))$

Computation time for the Rotor37 dataset

N = 1000
n ≃30000
500 000 Wasserstein distances to compute

→ **400 days** to build the 'Gram' matrix…

Sliced Wasserstein distance

Wasserstein (dimension 1)

$$\mathcal{W}_r^r(\mu,\nu) = \int_0^1 |\mathrm{F}^{-1}(\mu) - F^{-1}(\nu)|^{\mathrm{r}}\, \mathrm{d}t$$

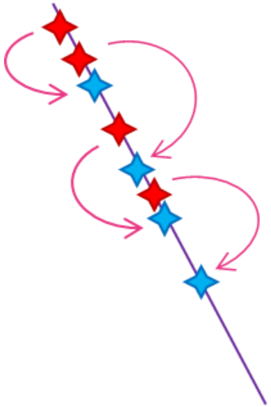
Quantile function

Empirical case:

$$\mu = \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i} \qquad \nu = \frac{1}{n}\sum_{i=1}^{n}\delta_{y_i}$$

$$\mathcal{W}_r^r(\mu,\nu) = \frac{1}{n}\sum_{i=1}^{n}|x_{(i)} - y_{(i)}\ |^r$$

Order statistics



This document and the information therein are the property of Safran. They must not be copied or communicated to a third party without the prior written authorization of Safran

SAFRAN

Sliced Wasserstein distance

Wasserstein (dimension 1)

$$\mathcal{W}_r^r(\mu,\nu) = \int_0^1 |\mathrm{F}^{-1}(\mu) - F^{-1}(\nu)|^{\mathrm{r}}\,\mathrm{d}t$$

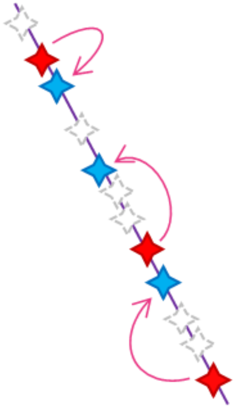
Quantile function

Empirical case:

$$\mu = \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i} \qquad \nu = \frac{1}{n'}\sum_{i=1}^{n'}\delta_{y_i}$$

$$\widehat{\mathcal{W}_r^r}(\mu,\nu) = \frac{1}{Q}\sum_{i=1}^{Q}\left|x_{(i)} - y_{(i)}\right|^r$$

Approximation with $Q \ll \max(n, n')$ quantiles



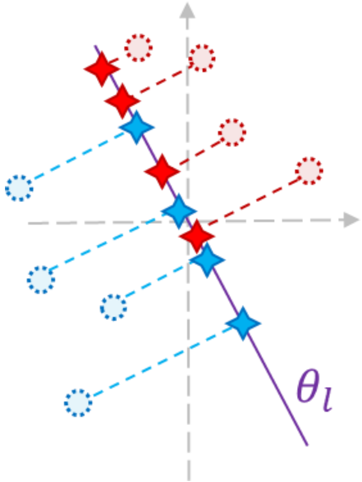
Sliced Wasserstein distance

Sliced Wasserstein

$$\mathcal{SW}_r^r(\mu,\nu) = \int_{\mathbb{S}^{s-1}} \mathcal{W}_r^r(\theta^*_\#\mu, \theta^*_\nu)\mathrm{d}\sigma(\theta)$$

Where:

- $\mathbb{S}^d$: d-dimensional unit sphere,
- σ : uniform distribution on $\mathbb{S}^d$
- $\theta^*_\#\mu$: push-forward measure of $\mu \in \mathcal{P}_r(\mathbb{R}^s)$ by $\theta^*\begin{pmatrix}\mathbb{R}^s \to \mathbb{R} \\ x \mapsto \langle\theta, x\rangle\end{pmatrix}$
- $\mathcal{W}_r^r$: 1-dimensional Wasserstein



Sliced Wasserstein distance

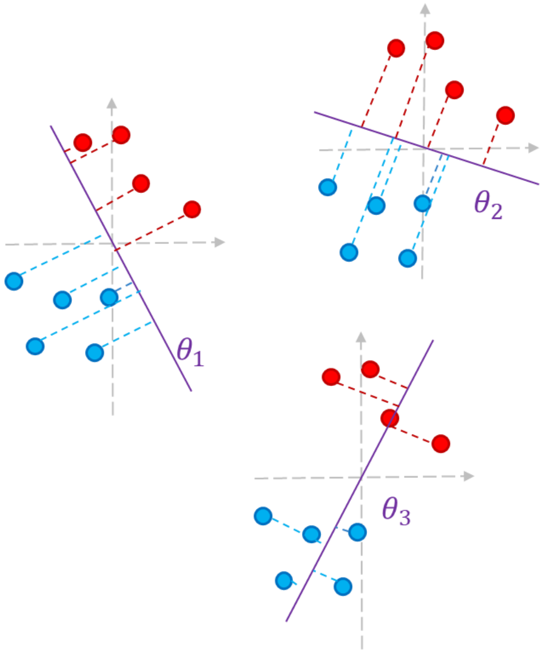
Estimated Sliced Wasserstein

$$\widehat{\mathcal{SW}_r^r}(\mu,\nu) = \frac{1}{P}\sum_{p=1}^{P} \widehat{\mathcal{W}_r^r}\left(\left(\theta_p^*\right)_\#\mu, \left(\theta_p^*\right)_\#\nu\right)$$

$$\widehat{\mathcal{W}_r^r}(\mu,\nu) = \frac{1}{Q}\sum_{i=1}^{Q} \left|x_{(i)} - y_{(i)}\right|^r$$

Where:

- Q: number of quantiles
- P: number of projections



SW distance: properties

Hilbertian pseudo distance

Let $\mathcal{X}$ be a space equipped with a pseudo-distance d. d is Hilbertian if there exists a Hilbert space $\mathcal{F}$ and a feature map $\phi: \mathcal{X} \rightarrow \mathcal{F}$ such that $d(x, y) = ||\phi(x) - \phi(y)||_{\mathcal{F}}$ for all $x, y \in \mathcal{X}$

Useful characterizations

[Hein, Bousquet, 2005]

Denoting $\langle x, y \rangle_d^{x_0} = \frac{1}{2}(d(x, x_0)^2 + d(y, x_0)^2 - d(x, y)^2)$, the three following properties are equivalent:

- d is a Hilbertian pseudo-distance
- $k_{poly}(x, y) = (c + \langle x, y \rangle_d^{x_0})^l$ for all $c \geq 0$, $l \in \mathbb{N}$, $x, y \in \mathcal{X}$ is positive definite
- $k_{exp}(x, y) = \exp(-\gamma d^{2\beta}(x, y))$ for all $\gamma \geq 0, \beta \in [0,1]$, $x, y \in \mathcal{X}$ is positive definite

SW substitution kernels

[Meunier et al., 2022]

$\mathcal{SW}_2$ and $\sqrt{\mathcal{SW}_1}$ are Hilbertian ⇒ positive definite substitution kernels ✓

Sliced Wasserstein Weisfeiler Lehman (SWWL)

SWWL kernel

[CP, Da Veiga, Garnier, Staber, 2024]

$\phi_{\mathrm{WL}}: G \mapsto X_G \in \mathbb{R}^{|V_G|\times \mathrm{d}(H+1)}$: continuous WL embeddings after H iterations

(μ_G associated empirical measure)

$$\widehat{\mathcal{SW}_2^2}(\mu,\nu) = \frac{1}{\mathrm{PQ}}\sum_{\mathrm{p}=1}^{\mathrm{P}}\sum_{q=1}^{Q}\left|u_q^{\theta_p} - u'^{\theta}_q\right|^2 = \left\|E_{\phi_{WL}(G)} - E_{\phi_{WL}(G')}\right\|_2^2$$

Precomputed embeddings: $E_{\phi_{WL}(G)}, E_{\phi_{WL}(G')} \in \mathbb{R}^{PQ}$ where $u_q^{\theta_p} = \langle \theta_p, \phi_{WL}(G)\rangle_{(q)}$

$$E_{\phi_{WL}(G)} = [u_1^{\theta_1}, \cdots, u_Q^{\theta_1}, \cdots, u_1^{\theta_P}, \cdots, u_Q^{\theta_P}]$$

$$k_{SWWL}(G,G') = e^{-\lambda \widehat{\mathcal{SW}_2^2}\left(\mu_{\phi_{WL}(G)}, \mu_{\phi_{WL}(G')}\right)}$$

Complexity for the Gram matrix

$$O(\ \boxed{NH\delta n}\ +\ \boxed{NP\,n\,(\log n + H)}\ +\ \boxed{N^2PQ\ )}$$

WL iterations

Projected Quantile Embeddings

RBF kernel

N: number of graphs

n: average number of nodes

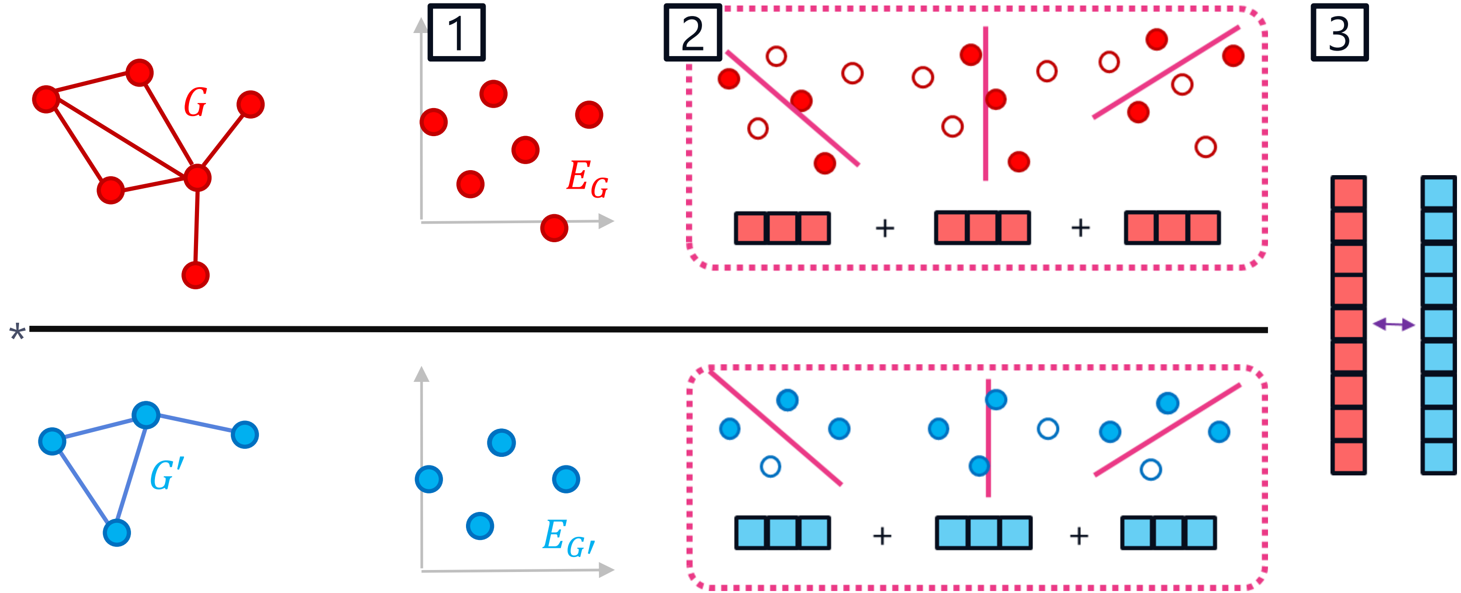
δ average degree

P: number of projections

Q: number of quantiles

H: number of WL iterations

Sliced Wasserstein Weisfeiler Lehman (SWWL)



1 WL iterations

2 Projected Quantile Embeddings

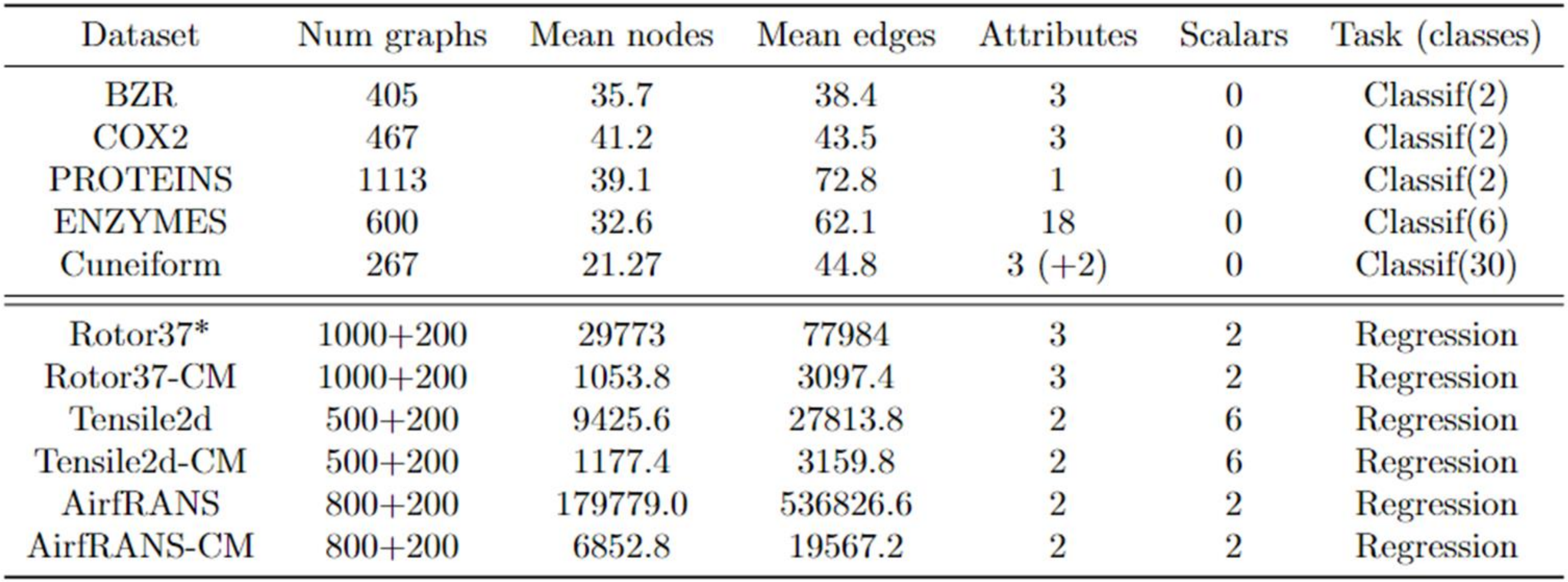
3 RBF kernel

* Steps 1 and 2 can be done separately for each input graph

Gaussian process regression for graph inputs

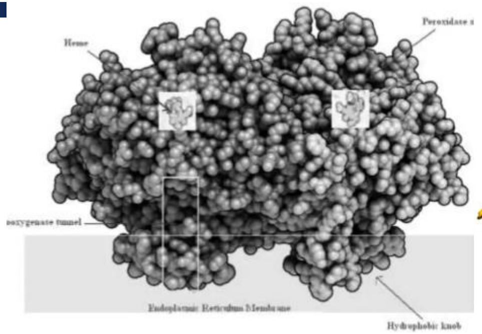
1- Problem statement
2- Classical approaches
3- SWWL kernel
4- Experiments

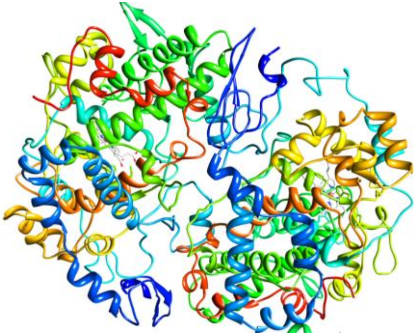
SWWL: experiments on small graphs

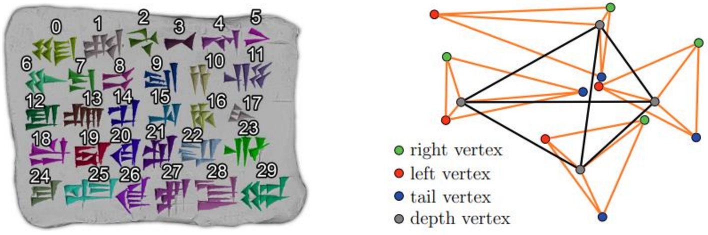


Dataset	Num graphs	Mean nodes	Mean edges	Attributes	Scalars	Task (classes)
BZR	405	35.7	38.4	3	0	Classif(2)
COX2	467	41.2	43.5	3	0	Classif(2)
PROTEINS	1113	39.1	72.8	1	0	Classif(2)
ENZYMES	600	32.6	62.1	18	0	Classif(6)
Cuneiform	267	21.27	44.8	3 (+2)	0	Classif(30)
Rotor37*	1000+200	29773	77984	3	2	Regression
Rotor37-CM	1000+200	1053.8	3097.4	3	2	Regression
Tensile2d	500+200	9425.6	27813.8	2	6	Regression
Tensile2d-CM	500+200	1177.4	3159.8	2	6	Regression
AirfRANS	800+200	179779.0	536826.6	2	2	Regression
AirfRANS-CM	800+200	6852.8	19567.2	2	2	Regression

SWWL: experiments on small graphs



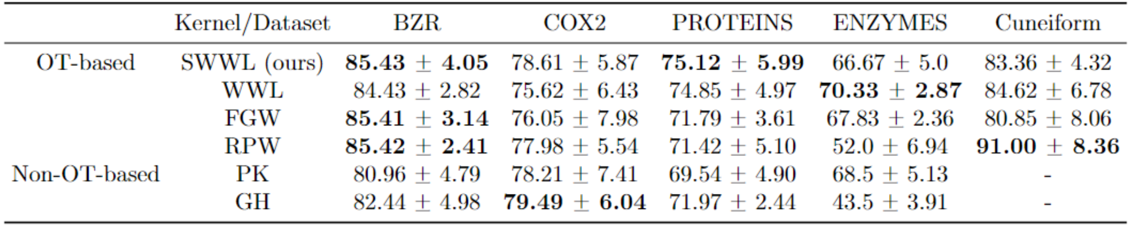




(a) Cuneiform tablet (b) Graph representation

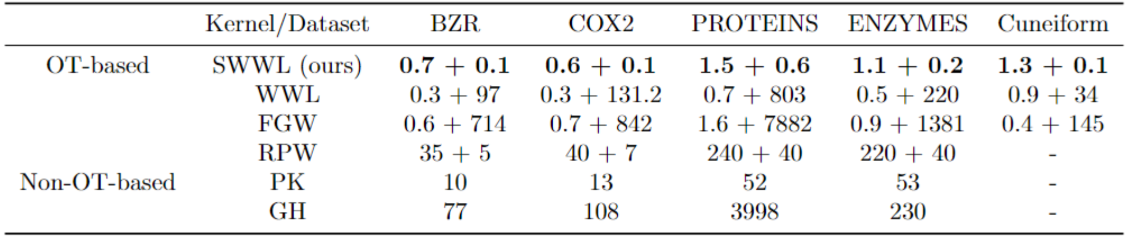
[Kriege et al., 2019]

RMSE (5 exp)



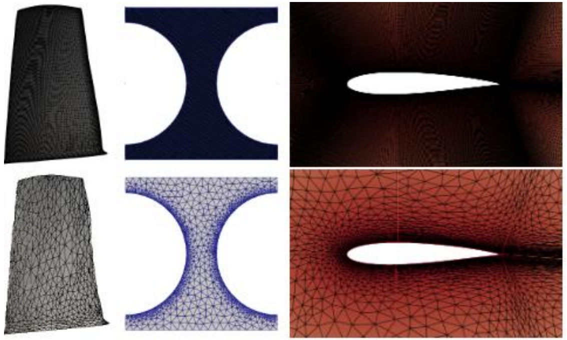
	Kernel/Dataset	BZR	COX2	PROTEINS	ENZYMES	Cuneiform
OT-based	SWWL (ours)	**85.43 ± 4.05**	78.61 ± 5.87	**75.12 ± 5.99**	66.67 ± 5.0	83.36 ± 4.32
	WWL	84.43 ± 2.82	75.62 ± 6.43	74.85 ± 4.97	**70.33 ± 2.87**	84.62 ± 6.78
	FGW	**85.41 ± 3.14**	76.05 ± 7.98	71.79 ± 3.61	67.83 ± 2.36	80.85 ± 8.06
	RPW	**85.42 ± 2.41**	77.98 ± 5.54	71.42 ± 5.10	52.0 ± 6.94	**91.00 ± 8.36**
Non-OT-based	PK	80.96 ± 4.79	78.21 ± 7.41	69.54 ± 4.90	68.5 ± 5.13	-
	GH	82.44 ± 4.98	**79.49 ± 6.04**	71.97 ± 2.44	43.5 ± 3.91	-

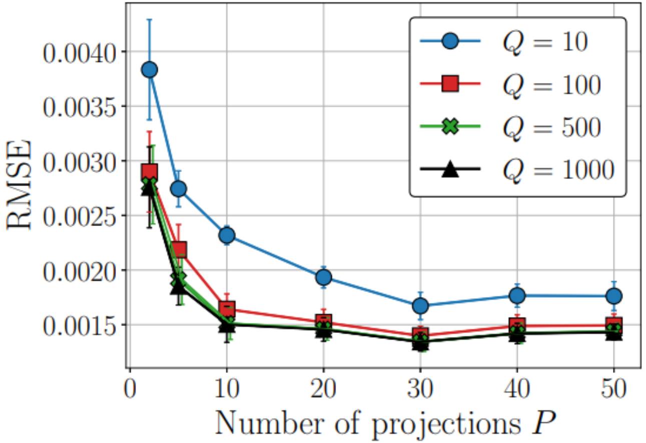
Time to build the Gram matrices



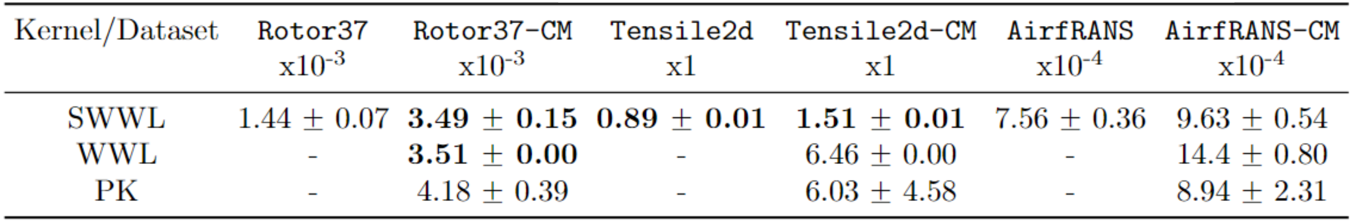
	Kernel/Dataset	BZR	COX2	PROTEINS	ENZYMES	Cuneiform
OT-based	SWWL (ours)	**0.7 + 0.1**	**0.6 + 0.1**	**1.5 + 0.6**	**1.1 + 0.2**	**1.3 + 0.1**
	WWL	0.3 + 97	0.3 + 131.2	0.7 + 803	0.5 + 220	0.9 + 34
	FGW	0.6 + 714	0.7 + 842	1.6 + 7882	0.9 + 1381	0.4 + 145
	RPW	35 + 5	40 + 7	240 + 40	220 + 40	-
Non-OT-based	PK	10	13	52	53	-
	GH	77	108	3998	230	-

SWWL: experiments on meshes



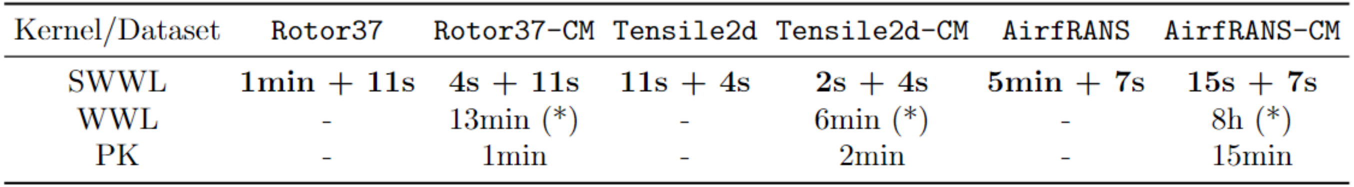


RMSE (5 exp)



Kernel/Dataset	Rotor37 x10^{-3}	Rotor37-CM x10^{-3}	Tensile2d x1	Tensile2d-CM x1	AirfRANS x10^{-4}	AirfRANS-CM x10^{-4}
SWWL	1.44 ± 0.07	**3.49 ± 0.15**	**0.89 ± 0.01**	**1.51 ± 0.01**	7.56 ± 0.36	9.63 ± 0.54
WWL	-	**3.51 ± 0.00**	-	6.46 ± 0.00	-	14.4 ± 0.80
PK	-	4.18 ± 0.39	-	6.03 ± 4.58	-	8.94 ± 2.31

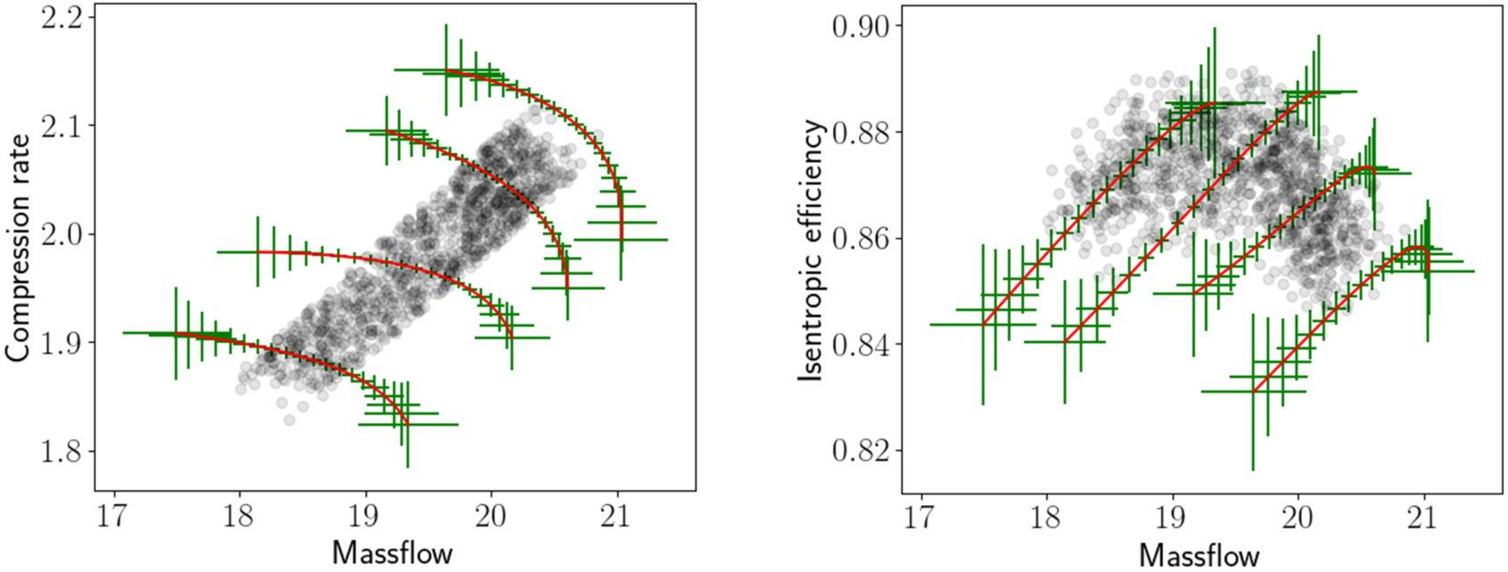
Time to build the Gram matrix



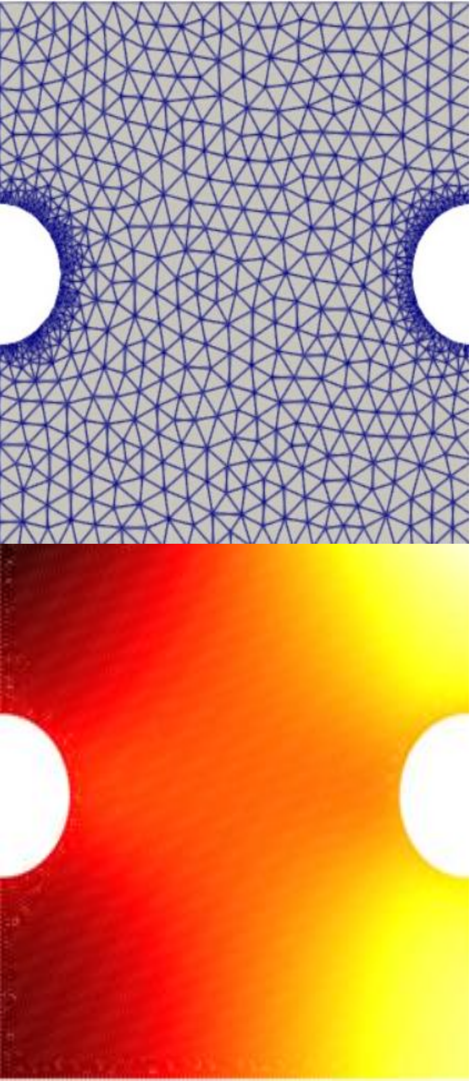
Kernel/Dataset	Rotor37	Rotor37-CM	Tensile2d	Tensile2d-CM	AirfRANS	AirfRANS-CM
SWWL	**1min + 11s**	**4s + 11s**	**11s + 4s**	**2s + 4s**	**5min + 7s**	**15s + 7s**
WWL	-	13min (*)	-	6min (*)	-	8h (*)
PK	-	1min	-	2min	-	15min

(*) in parallel, using 100 jobs

Engineering curves



Predicted compression rate/isentropic efficiency with respect to the massflow for a test mesh, 4 input rotations and 20 input pressure (going beyond the range of train/test datasets) with 95% confidence intervals
Rotor37, P=50, Q=100, H=6



Prediction of output fields

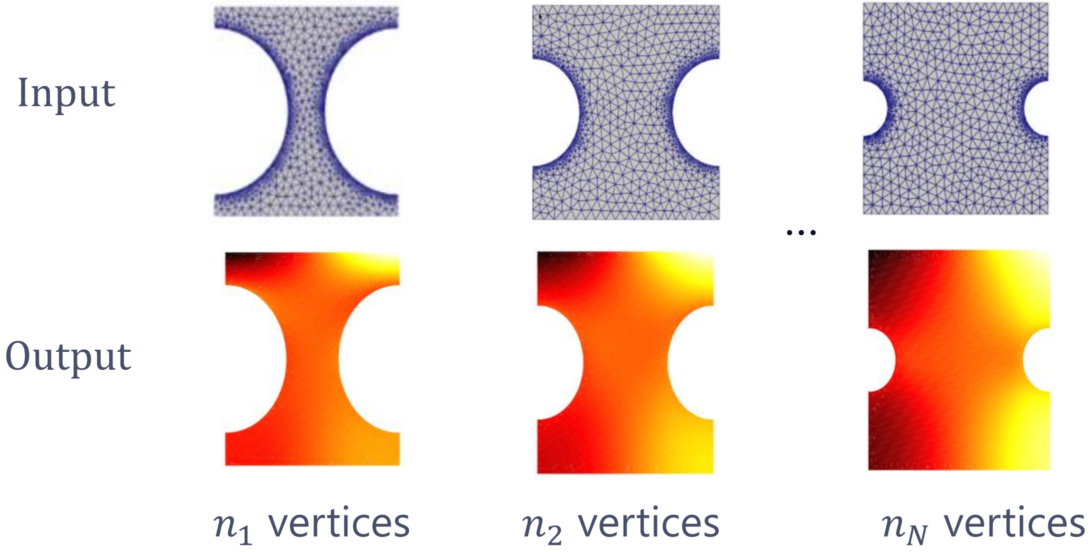
1- Problem statement

2- Related approaches

3- TOS-GP

4- Experiments

Learning output fields/signals



$$\mathcal{Y} = \bigcup_{X=(V,E,w,F)\in\mathcal{X}} \{Y : V \to \mathbb{R}\}$$

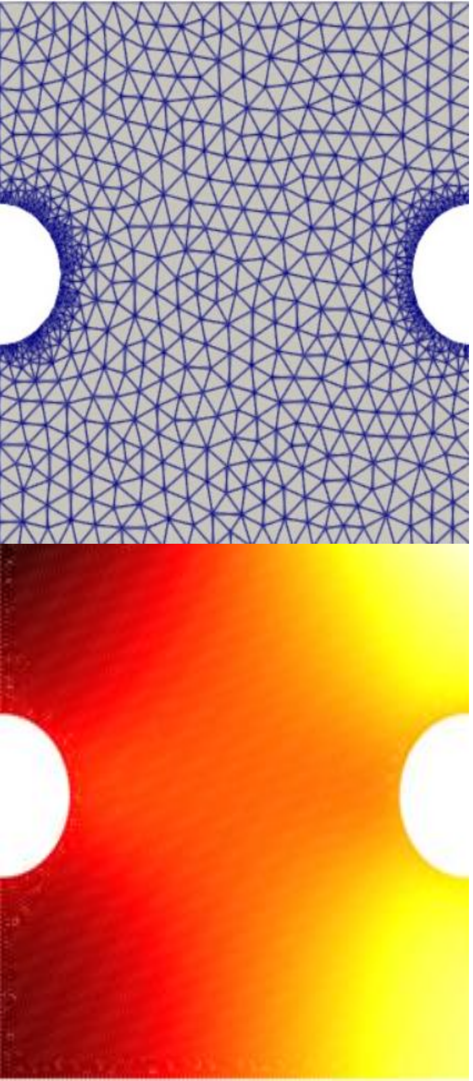
Train data: $\{(x^{(i)}, y^{(i)})\}_{i=1,\cdots,N}$

$$x^{(i)} = \left(V^{(i)}, E^{(i)}, w^{(i)}, F^{(i)}\right) \in \mathcal{X}$$
$$y^{(i)} \in \mathcal{Y}$$

1- Inputs can have **different sizes**, so do the outputs
2- **No natural ordering** of the outputs dimensions
3- The output dimension can be very **large**

By abuse of notations:

$$y^{(i)} = (y_1^{(i)}, \cdots, y_{|V^{(i)}|}^{(i)})$$



Prediction of output fields

1- Problem statement

2- Related approaches

3- TOS-GP

4- Experiments

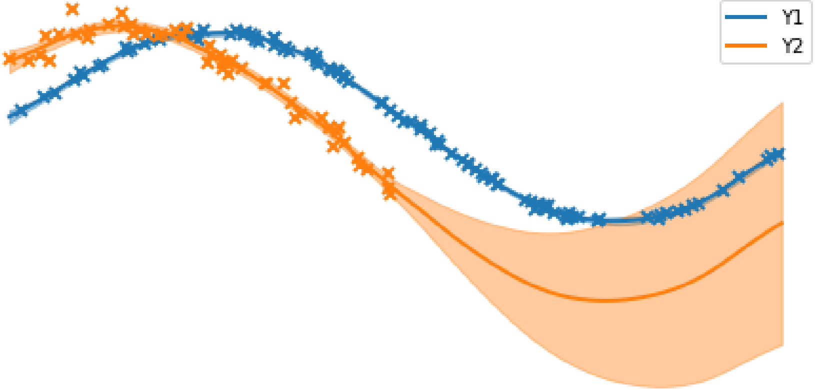
Related approaches: Multi-Output GP

MOGP [Goovaerts, 1997]

$f : \mathcal{X} \to \mathbb{R}^D \qquad f(x) = (f_1(x), \cdots, f_D(x)\ )$

Vector-Valued Kernel: $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{D\times D}$

$$cov\big(f_j(x), f_l(x')\big) = k_{j,l}(x, x')$$



Intrinsic Coregionalization Model (ICM):

$$k_{l,j}(x, x') = B_{j,l}\ k_{scal}(x, x')$$
$$K = B \otimes K_{scal}$$

Where $B \in \mathbb{R}^{D\times D}, K_{scal} \in \mathbb{R}^{N\times N}, K \in \mathbb{R}^{(ND)\times(ND)}$

Issues for us ✗

- Outputs are not vectors
- $D_i = n_i$: very large outputs (even if they can be put in vectorial form)

Related approaches: operator-valued GP

Operator/Function valued Gaussian Processes

[Kadri, 2016]

$f: \mathcal{X} \to \mathcal{Y} = L^2(\Omega_Y)$ where Ω_Y is a compact set

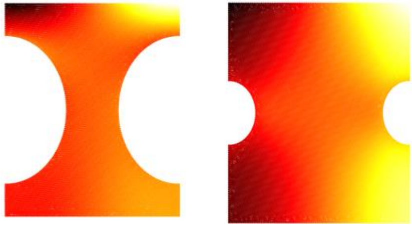
Operator valued kernel: $k: \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$

Block operator kernel matrix: $K \in \mathcal{L}(\mathcal{Y}^N)$

$$K = \begin{bmatrix} K_{1,1} & \cdots & K_{1N} \\ \vdots & \ddots & \vdots \\ K_{N1} & \cdots & K_{NN} \end{bmatrix}$$

Issues for us ✘

- Function domain Ω_Y would not be fixed
- In practice, OVGP rely on a discretization to grid points common to all samples



Related approaches: structured output prediction

Figures from
[Brouard, 2016]

$$\begin{tikzcd}
 & \mathcal{F}_y \arrow[d, dashed, shift left, "f"] \\
\mathcal{X} \arrow[ur, "h"] \arrow[r] & \mathcal{Y} \arrow[u, shift left, "\varphi_y"]
\end{tikzcd}$$

Output Kernel Regression

$$\begin{tikzcd}
\mathcal{F}_x \arrow[r, "g"] & \mathcal{F}_y \arrow[d, dashed, shift left, "f"] \\
\mathcal{X} \arrow[u, "\varphi_x"] \arrow[ur, "h"] \arrow[r] & \mathcal{Y} \arrow[u, shift left, "\varphi_y"]
\end{tikzcd}$$

Kernel Dependency Estimation
[Weston, 2003]

$$\begin{tikzcd}
\mathcal{B}(\mathcal{F}_y, \mathcal{H}) \arrow[r, "g"] & \mathcal{F}_y \arrow[d, dashed, shift left, "f"] \\
\mathcal{X} \arrow[u, "\phi_x"] \arrow[ur, "h"] \arrow[r] & \mathcal{Y} \arrow[u, shift left, "\varphi_y"]
\end{tikzcd}$$

Input-Output kernel regression
[Brouard, 2016]

Issues for us ✗

- Need to define a kernel in the output space
- Solving a pre-image problem

Related approaches: Graph Signal Processing

Graph Fourier Transform [Schuman et al., 2013]

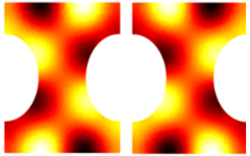
$L \in \mathbb{R}^{n\times n}$: Laplacian matrix of $G = (V, E)$.
Eigenvalues: $0 = \lambda_1 \le \cdots \le \lambda_n$. Eigenvectors: $U_1, \cdots, U_n$
Signal: $y: V \to \mathbb{R}$ (by abuse of notation, $y = (y_1, \cdots, y_n)$)

l-th GFT coefficient: $\tilde{y}_l = \langle y, U_l \rangle,\ 1 \le l \le Q \le n$

Inverse (truncated) GFT: $$y_i = \sum_{l=1}^{Q} \tilde{y}_l \times (U_l)_i$$

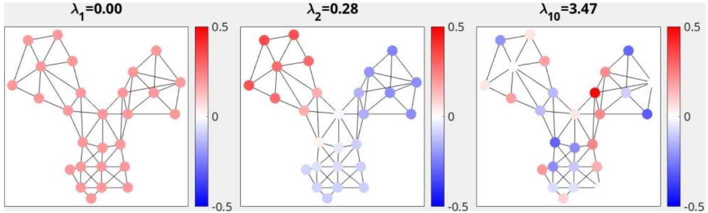
Issues for us ✘

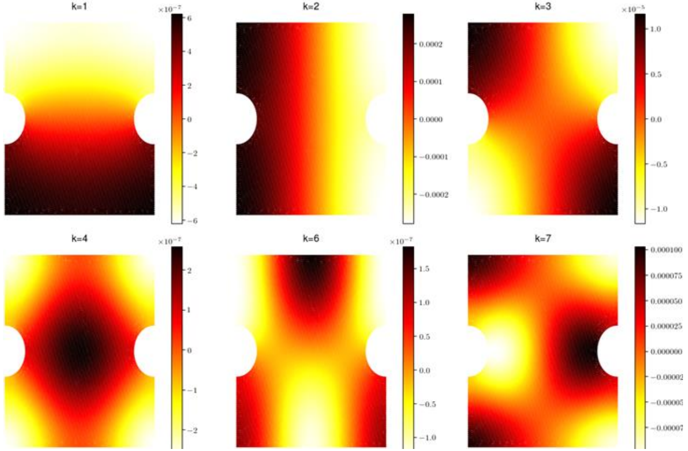
- Signs/choice of basis of eigenvectors?
- Numerical unstabilities for small eigenvalues



→ reversed eigenvector

Figure from [Ortega, 2018]





Related approaches: Mesh Morphing Gaussian Process

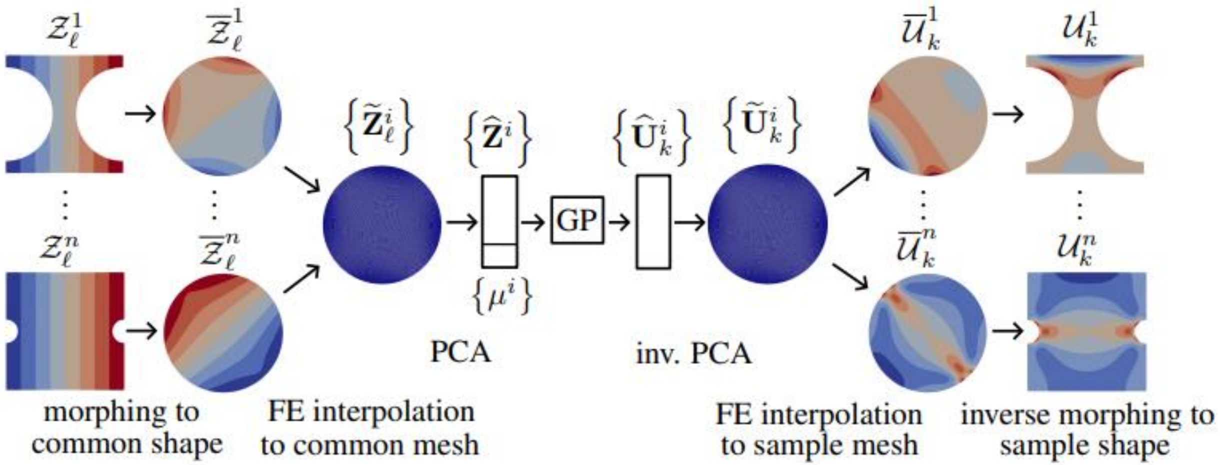

$\mathcal{Z}^1_\ell$
$\overline{\mathcal{Z}}^1_\ell$
$\mathcal{Z}^n_\ell$
$\overline{\mathcal{Z}}^n_\ell$
$\{\widetilde{\mathbf{Z}}^i_\ell\}$
$\{\widehat{\mathbf{Z}}^i\}$
$\{\mu^i\}$
GP
$\{\widehat{\mathbf{U}}^i_k\}$
$\{\widetilde{\mathbf{U}}^i_k\}$
$\overline{\mathcal{U}}^1_k$
$\mathcal{U}^1_k$
$\overline{\mathcal{U}}^n_k$
$\mathcal{U}^n_k$
PCA
inv. PCA
morphing to common shape
FE interpolation to common mesh
FE interpolation to sample mesh
inverse morphing to sample shape

Figure from

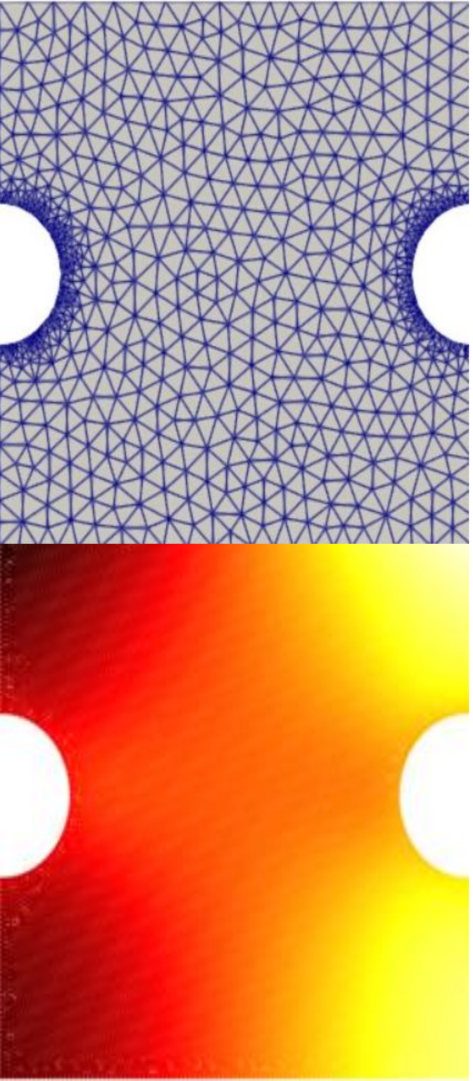
[Casenave, 2024]

Issues for us ✗

- Specific to mesh data
- Morphing → same topology
- Transformation of both inputs and outputs

Benefits ✓

- Uncertainty quantification
- Very good results for output field prediction

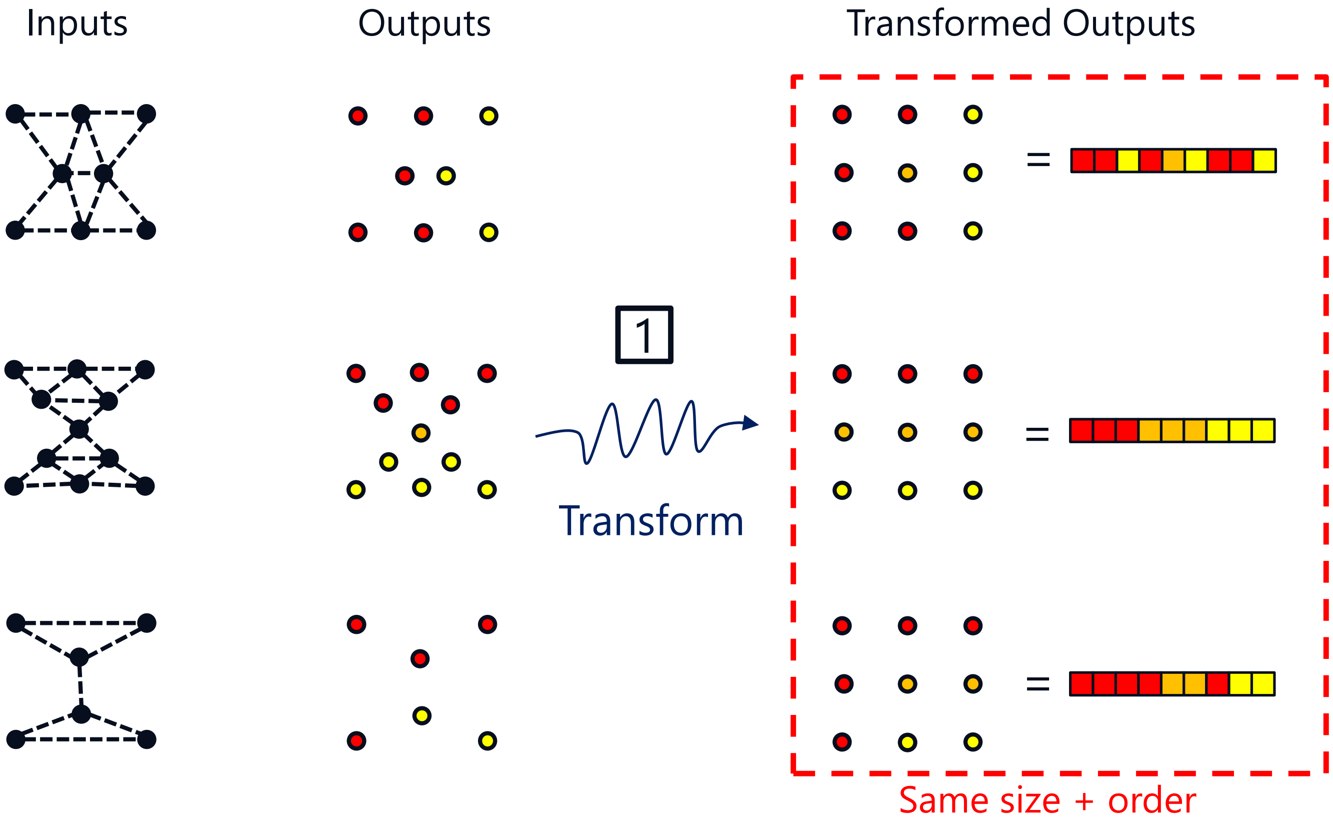


Prediction of output fields

1- Problem statement
2- Related approaches
3- TOS-GP
4- Experiments

SAFRAN

Express signals/fields in the same space?



Regularized Wasserstein distance

Regularized Wasserstein distance (discrete case)

$$\mathcal{W}_\lambda\ (\mu,\nu) = L_\lambda(\mu,\nu,P_\lambda) \qquad P_\lambda = \underset{P\in U(n,n')}{argmin}\ L_\lambda(\mu,\nu,P)$$

$$L_\lambda(\mu,\nu,P) = \langle C^{\mu,\nu}, P\ \rangle - \lambda H(P) \qquad ,\lambda > 0$$

Entropic regularization

Where:

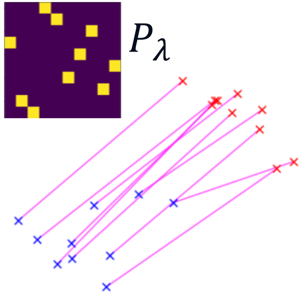
- $\mu = \frac{1}{n}\sum_{i=1}^{n}\delta_{x_i} \quad \nu = \frac{1}{n'}\sum_{i=1}^{n'}\delta_{y_i}$
- $U(n,n') = \left\{P \in \mathbb{R}_+^{n\times n'} : P1_{n'} = \frac{1}{n}1_n, P1_n = \frac{1}{n'}1_{n'}\right\}$
- $C^{\mu,\nu} = \left[\left\|x_i - y_j\right\|^r\right]_{i=1\ldots n,\ j=1\ldots n'}$

Why regularizing?

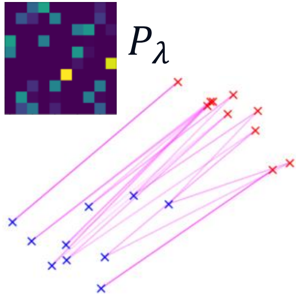
1- Computation

Sinkhorn: $O(n^2\log(n))$

2- Smoothing of transport plans



$\lambda = 0$



$\lambda > 0$

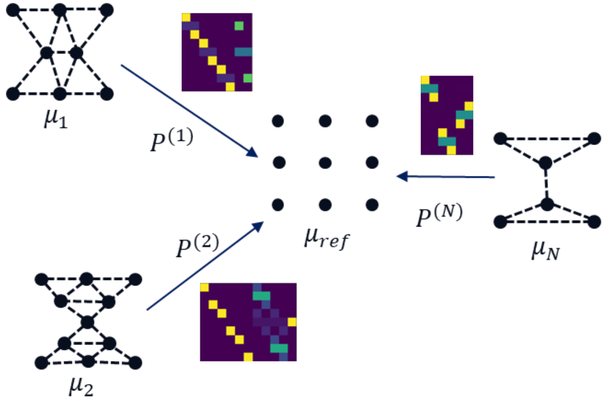
Transferring fields with transport plans

Part 1: getting transport plans (**input** space)

μ_{ref}: reference measure of size n_{ref}

$$\mu_i = \frac{1}{n_i}\sum_{j=1}^{n_i} \delta_{[\phi_{WL}(G^{(i)})]_j} \quad \text{: WL embeddings of input graph } i$$

$$P_\lambda^{(i)} = \underset{P\in U(n_i,n_{ref})}{argmin} \; L_\lambda\left(\mu_i,\mu_{ref},P\right) \in \mathbb{R}^{n_i\times n_{ref}}$$



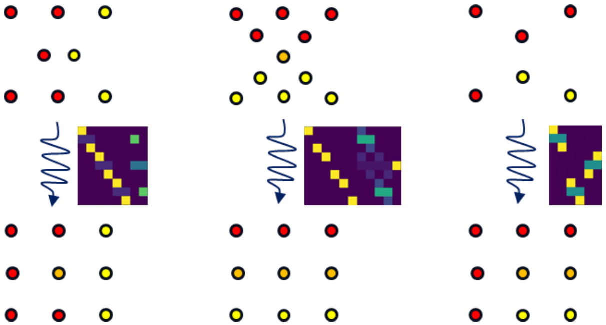
Part 2: transferring **output** signals

$$T^{(i)} = \left(n_{ref}P_\lambda^{(i)}\right)^T y^{(i)} \in \mathbb{R}^{n_{ref}}$$

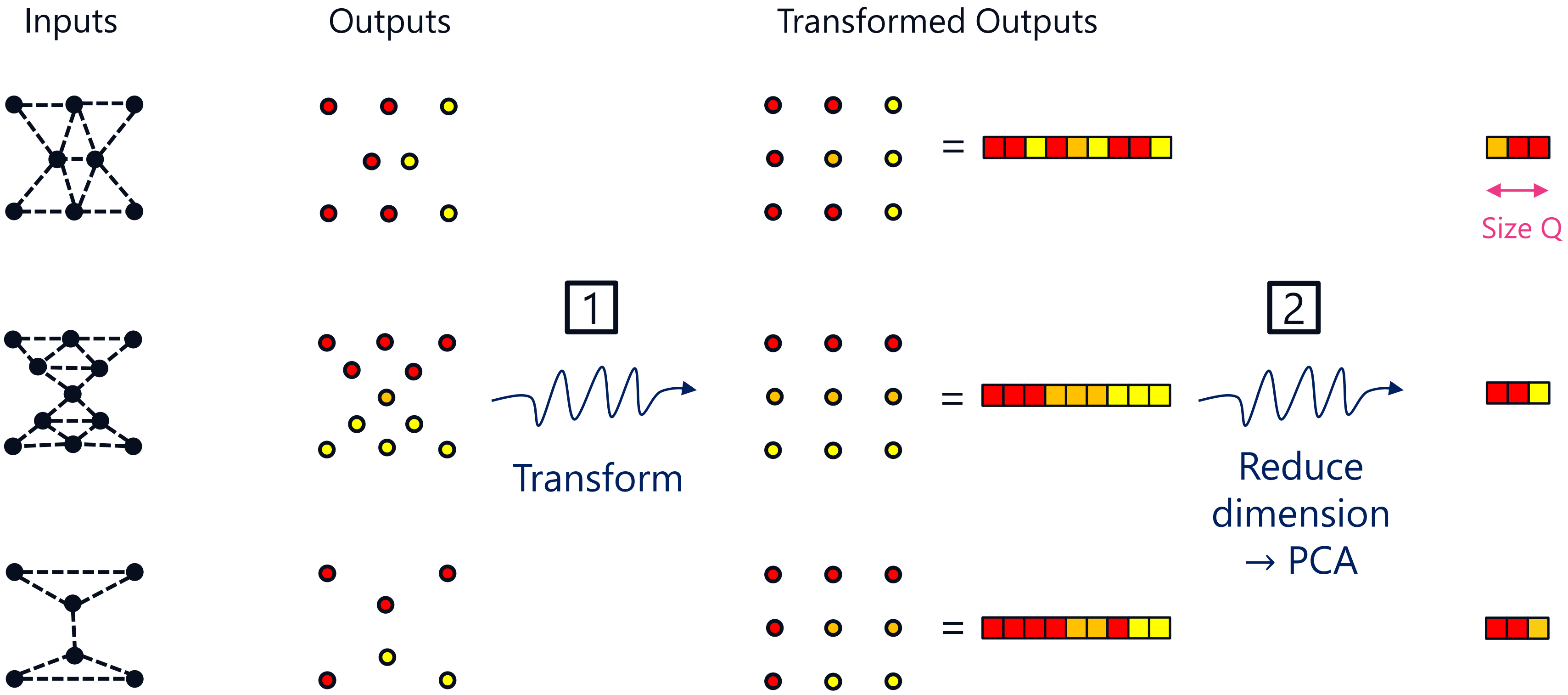
Transferred field

$$\tilde{y}^{(i)} = \left(n_i P_\lambda^{(i)}\right) T^{(i)} \in \mathbb{R}^{n_i}$$

Reconstructed field



Express signals/fields in the same space?



Dimension reduction (in practice)

Principal component analysis

[Kontolati 2022]

$\boldsymbol{T} = (T^{(1)}, \cdots, T^{(N)}) \in \mathbb{R}^{N \times n_{ref}}$

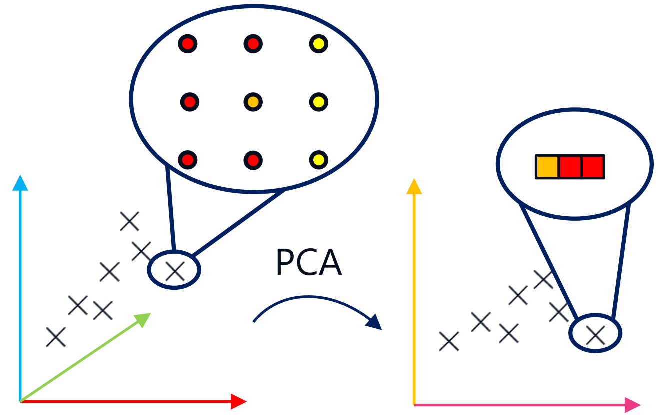
$\overline{\boldsymbol{T}} = \boldsymbol{T}$ centered

$$\frac{1}{N}\overline{\boldsymbol{T}}^T\overline{\boldsymbol{T}} = EDiag(\lambda_1, \cdots, \lambda_Q)\mathrm{E}^{\mathrm{T}}$$

$\lambda_1 \leq \cdots, \leq \lambda_Q$: eigenvalues

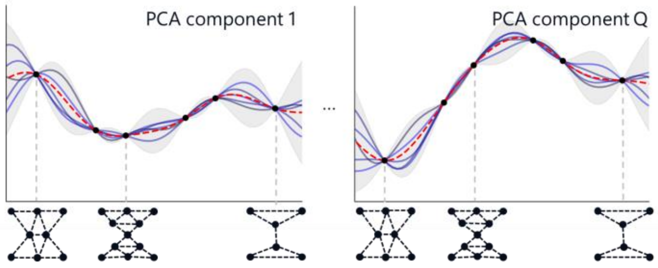
$\mathrm{E} \in \mathbb{R}^{n_{ref} \times Q}$: eigenvectors

Q first PCA coefficients: $C = \boldsymbol{T}E \in \mathbb{R}^{N \times Q}$



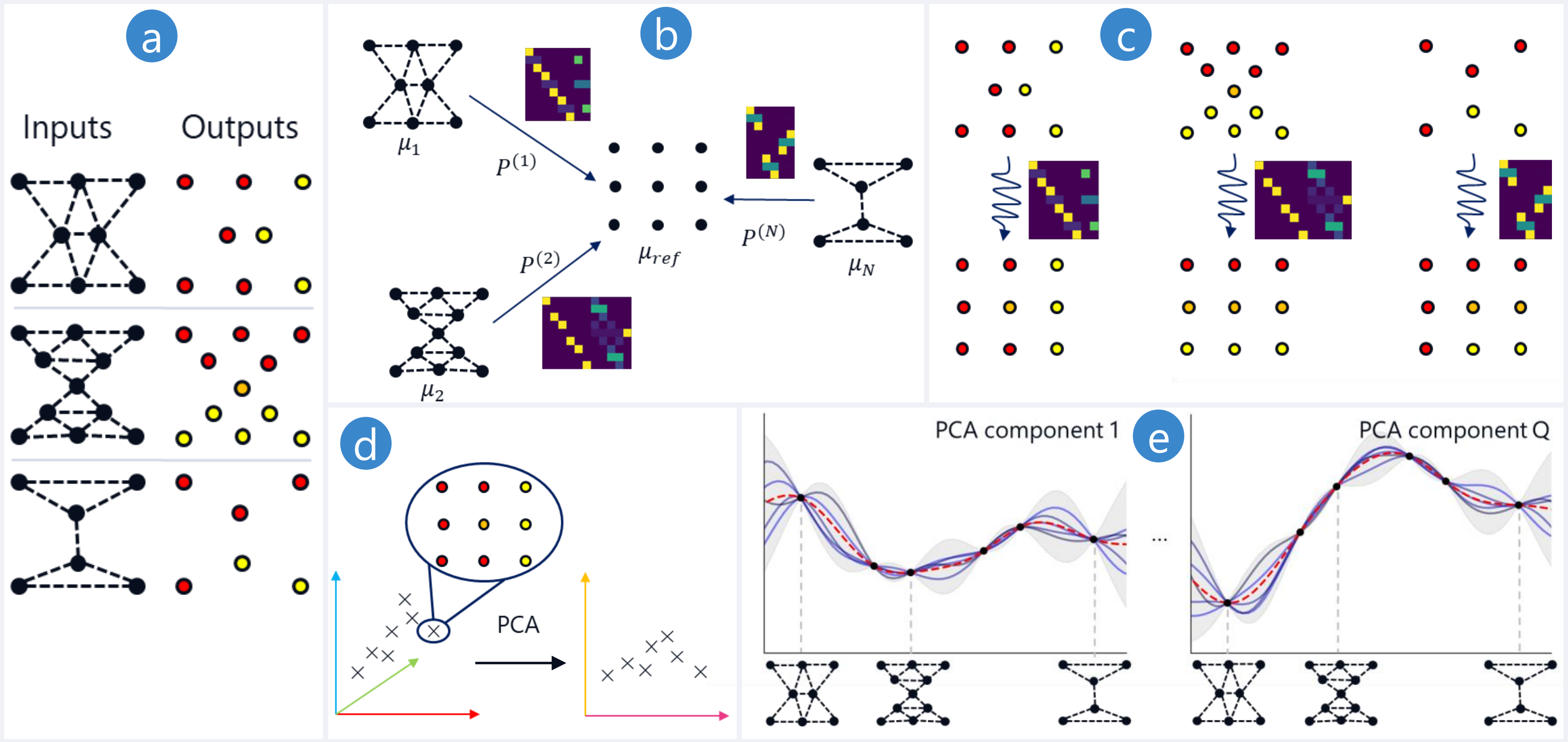
Why PCA?

Linear model ⇒ analytical formulas for UQ



TOS-GP: Transported Output Signal Gaussian Processes

[CP, Da Veiga, Garnier, Staber, 2024+]



TOS-GP: Transported Output Signal Gaussian Processes

Train

1- Compute all regularized transport plans to the reference +transfer fields
2- PCA
3- Independent SWWL GPs

Test

1- Predict PCA coefficients for new test outputs
2- Inverse PCA → predicted transferred fields
3- Compute all regularized transport plans to the reference +transfer back fields

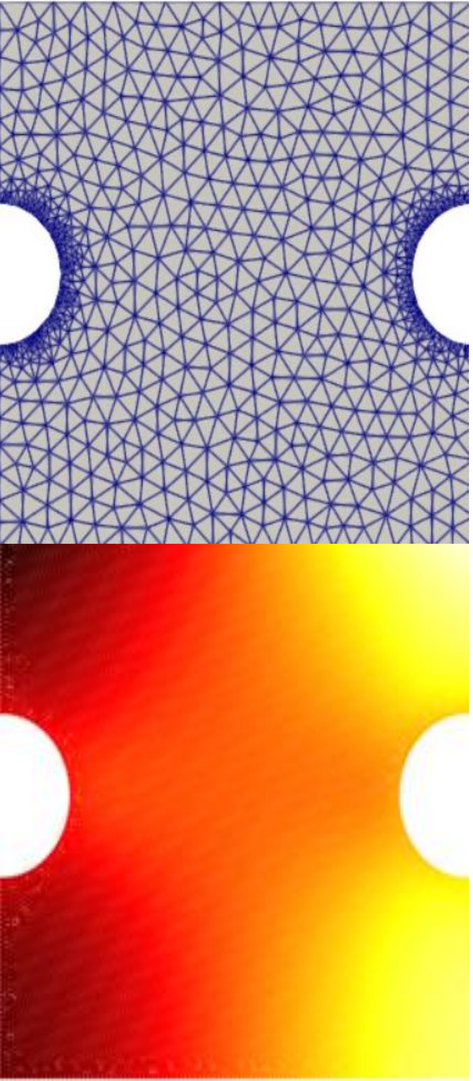
Uncertainty propagation

Hyperparameters

- Reference measure
- Regularization parameter
- Number of WL iterations

Remarks

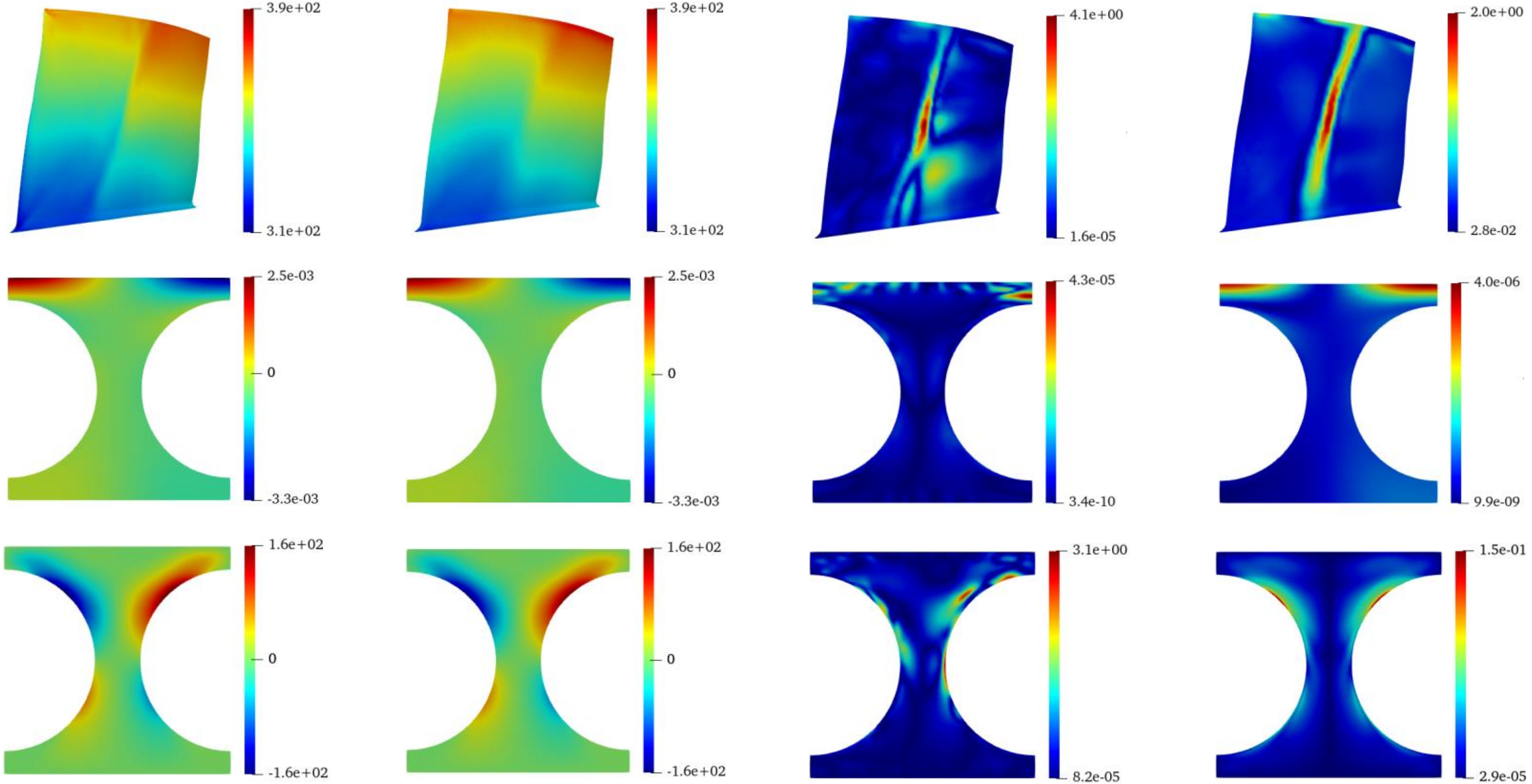
- Agnostic to the choice of the regressor
- No assumption on the data (mesh, topology, …)
- Analytical UQ formulas



Prediction of output fields

1- Problem statement
2- Related approaches
3- TOS-GP
4- Experiments

TOS-GP: predictions and uncertainties



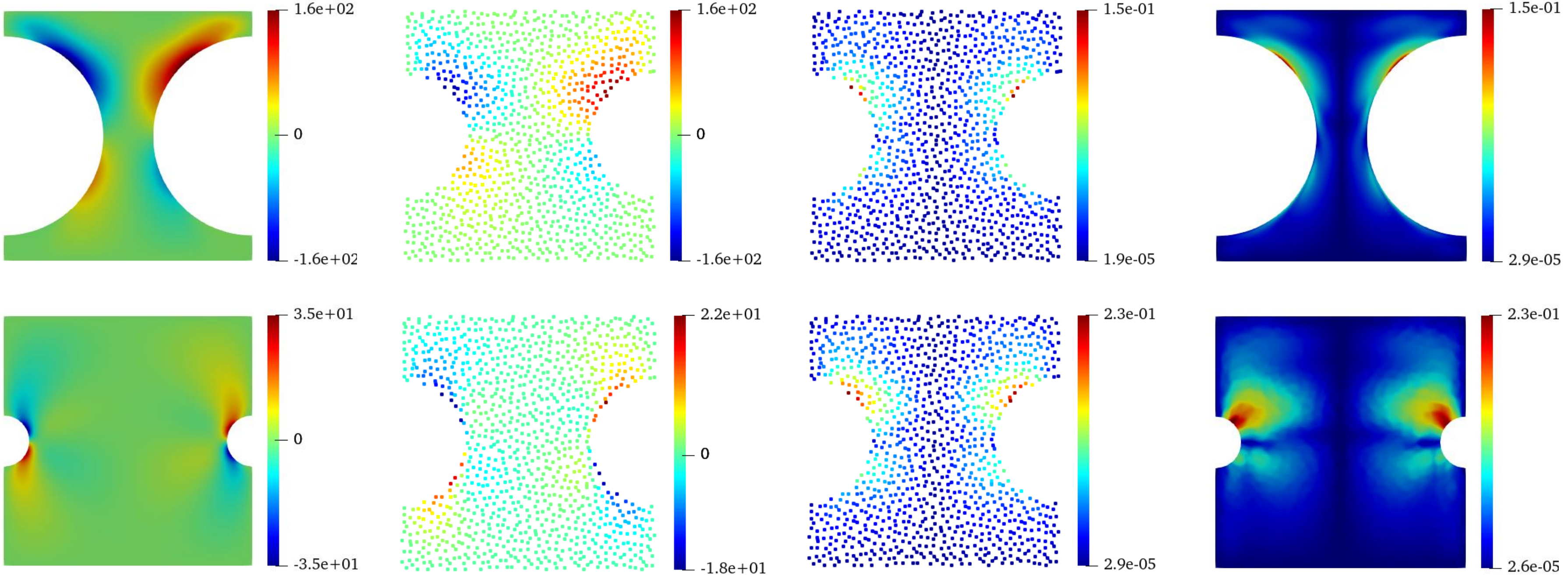
Ground truth

Prediction

Absolute error

Posterior std

TOS-GP: uncertainty propagation (field σ_{12})



Ground truth

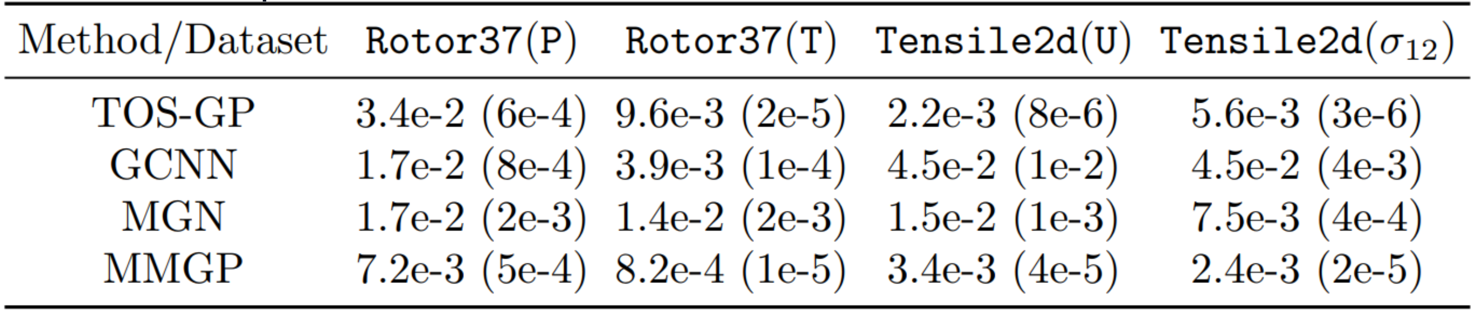
Prediction (transferred space)

Posterior std (transferred space)

Posterior std

TOS-GP: regression scores

RRMSE (10 exp)

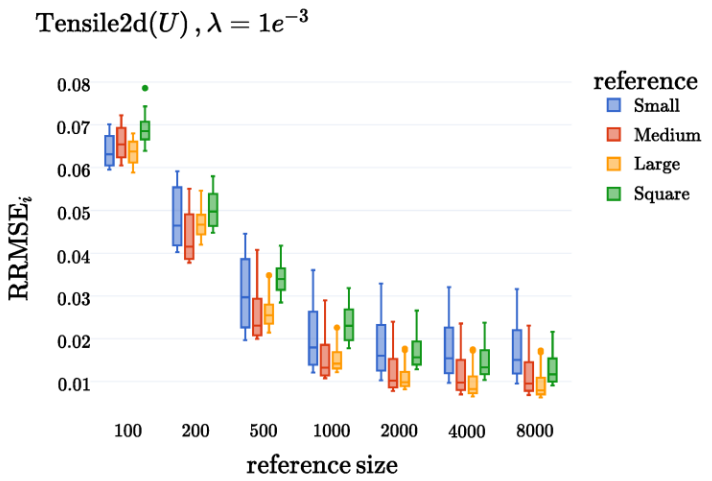


Method/Dataset	Rotor37(P)	Rotor37(T)	Tensile2d(U)	Tensile2d(σ_{12})
TOS-GP	3.4e-2 (6e-4)	9.6e-3 (2e-5)	2.2e-3 (8e-6)	5.6e-3 (3e-6)
GCNN	1.7e-2 (8e-4)	3.9e-3 (1e-4)	4.5e-2 (1e-2)	4.5e-2 (4e-3)
MGN	1.7e-2 (2e-3)	1.4e-2 (2e-3)	1.5e-2 (1e-3)	7.5e-3 (4e-4)
MMGP	7.2e-3 (5e-4)	8.2e-4 (1e-5)	3.4e-3 (4e-5)	2.4e-3 (2e-5)

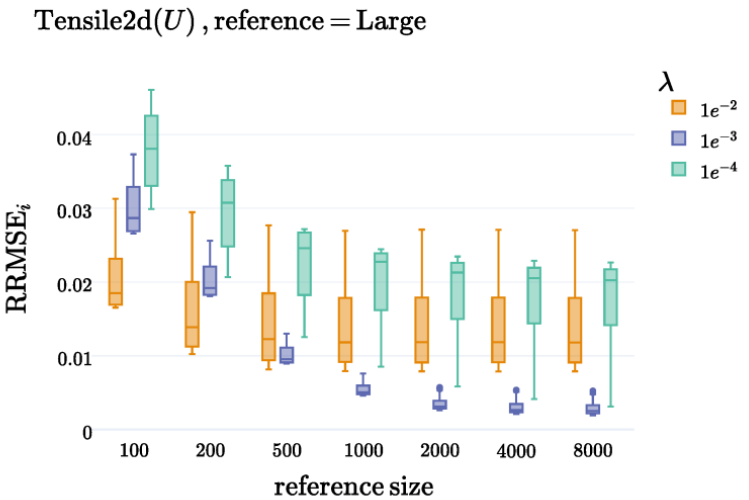
$$RRMSE^2\left(\{y^{(i)}\}_{i=1,\cdots,N_*},\{\hat{y}^{(i)}\}_{i=1,\cdots,N_*}\right)=\frac{1}{N_*}\sum_{i=1}^{N_*}RRMSE_i^2\left(y^{(i)},\hat{y}^{(i)}\right)$$

$$RRMSE_i^2\left(y^{(i)},\hat{y}^{(i)}\right)=\frac{\left\|y^{(i)}-\hat{y}^{(i)}\right\|_2^2}{n_{*i}\left\|y^{(i)}\right\|_\infty^2}$$

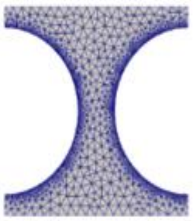
TOS-GP: regression scores



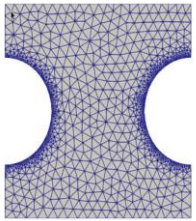
$\text{Tensile2d}(U)\,,\lambda = 1e^{-3}$



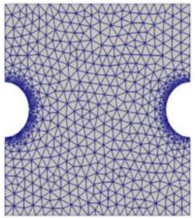
$\text{Tensile2d}(U)\,,\text{reference} = \text{Large}$



Small



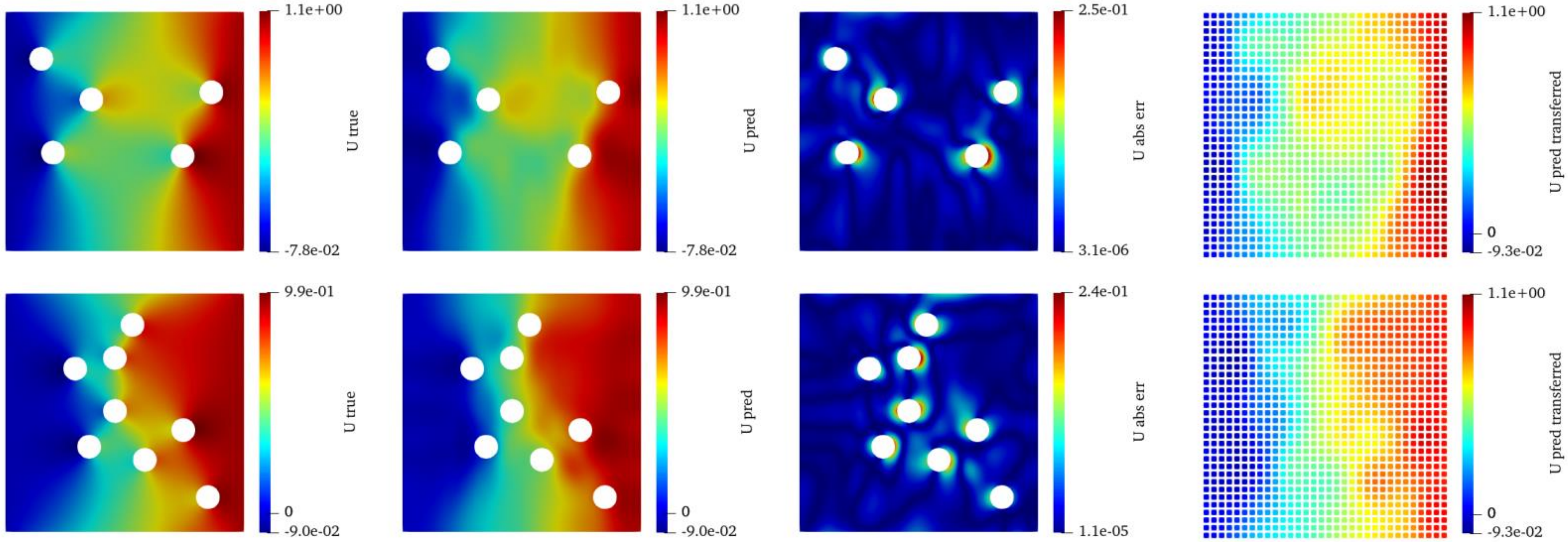
Medium



Large

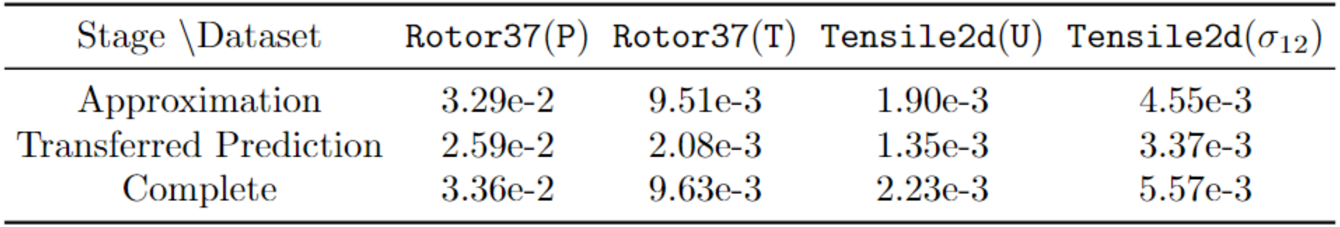
MMD to obtain subsampled empirical distributions

Use case: varying topologies



TOS-GP: limitations

Approximation vs prediction error



Stage \Dataset	Rotor37(P)	Rotor37(T)	Tensile2d(U)	Tensile2d(σ_{12})
Approximation	3.29e-2	9.51e-3	1.90e-3	4.55e-3
Transferred Prediction	2.59e-2	2.08e-3	1.35e-3	3.37e-3
Complete	3.36e-2	9.63e-3	2.23e-3	5.57e-3

Discontinuous signals

Large signal variations
-> more sensitive to the regularization of transport plans

Computation times

* Depends on the size of the input, the reference, and the regularization

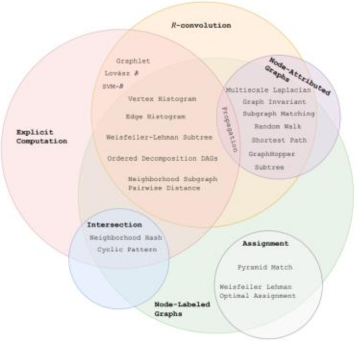
1 transport plan: Tensile2d: ~10 seconds* Rotor37: ~50 seconds*

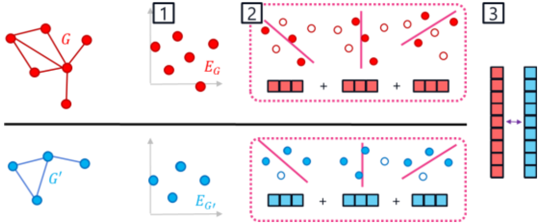
→ Embarrassingly parallel. But preprocessing required for new test inputs

Conclusion

Inputs = Graphs, Outputs = Scalars

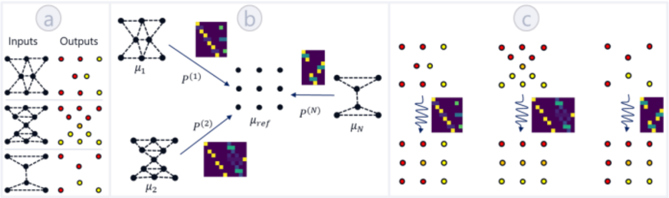
- Lots of approaches, but many
  - Are not tractable
  - Do not handle continuous attributes
  - Do not guarantee positive definiteness
  - Are too dependent on the graph structure
- SWWL graph kernel
  - Positive definite
  - Can consider very large graphs
  - Competitive results for mesh-based Gaussian process regression

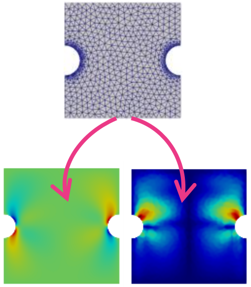




Inputs = Graphs, Outputs = Signals

- Classical techniques impossible to use directly
  - MOGP, OVGP, GSP, dimension reduction, …
- TOS-GP
  - Extension of GPs to predict signal outputs
  - Optimal transport + Dimension reduction





Acknowledgments

This work was supported by the French National Research Agency (ANR) through the SAMOURAI project under grant ANR20-CE46-0013.

References

- **Graph kernels, Gaussian processes**
  - Nikolentzos, G., Siglidis, G., & Vazirgiannis, M. (2021). Graph kernels: A survey.
  - Kriege, N. M., Johansson, F. D., & Morris, C. (2020). A survey on graph kernels.
  - Feragen, A., Kasenburg, N., Petersen, J., de Bruijne, M., & Borgwardt, K. (2013). Scalable kernels for graphs with continuous attributes.
  - Williams, C. K., & Rasmussen, C. E. (2006). Gaussian processes for machine learning.
  - Hein, M., & Bousquet, O. (2005). Hilbertian metrics and positive definite kernels on probability measures.
  - Kriege, N. M., Fey, M., Fisseler, D., Mutzel, P., & Weichert, F. (2018, August). Recognizing cuneiform signs using graph based methods.
- **Optimal transport**
  - Peyré, G., & Cuturi, M. (2019). Computational optimal transport: With applications to data science.
  - Togninalli, M., Ghisu, E., Llinares-López, F., Rieck, B., & Borgwardt, K. (2019). Wasserstein weisfeiler-lehman graph kernels.
  - Meunier, D., Pontil, M., & Ciliberto, C. (2022, June). Distribution Regression with Sliced Wasserstein Kernels.

References

- **Multi-output approaches**
  - Goovaerts, P. (1997). *Geostatistics for natural resources evaluation*.
  - Kadri, H., Duflos, E., Preux, P., Canu, S., Rakotomamonjy, A., & Audiffren, J. (2016). Operator-valued kernels for learning from functional response data.
  - Kadri, H., Ghavamzadeh, M., & Preux, P. (2013, February). A generalized kernel approach to structured output learning.
  - Weston, J., Chapelle, O., Vapnik, V., Elisseeff, A., & Schölkopf, B. (2003). Kernel dependency estimation.
  - Brouard, C., Szafranski, M., & d'Alché-Buc, F. (2016). Input output kernel regression: Supervised and semi-supervised structured output prediction with operator-valued kernels.
  - Kontolati, K., Loukrezis, D., Giovanis, D. G., Vandanapu, L., & Shields, M. D. (2022). A survey of unsupervised learning methods for high-dimensional uncertainty quantification in black-box-type problems.
  - Casenave, F., Staber, B., & Roynard, X. (2024). Mmgp: a mesh morphing gaussian process-based machine learning method for regression of physical problems under nonparametrized geometrical variability
  - Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., & Vandergheynst, P. (2013). The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains.

MMD subsampling

Definition 2 (Maximum Mean Discrepancy). *Let x and y be random variables defined on a topological space $\mathcal{Z}$, with respective Borel probability measures p and q. Let $k : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ be a kernel function and let $\mathcal{H}(k)$ be the associated reproducing kernel Hilbert space. The maximum mean discrepancy between p and q is defined as*

$$\mathrm{MMD}_k(p,q) = \sup_{\|f\|_{\mathcal{H}(k)} \leqslant 1} \left| \mathbb{E}_{x\sim p}[f(x)] - \mathbb{E}_{y\sim q}[f(y)] \right| .$$

The MMD admits the following closed-form expression:

$$\mathrm{MMD}_k(p,q)^2 = \mathbb{E}_{x\sim p, x'\sim p}[k(x,x')] + \mathbb{E}_{y\sim q, y'\sim q}[k(y,y')] - 2\mathbb{E}_{x\sim p, y\sim q}[k(x,y)] , \qquad (12)$$

$$\mu_{i+1}(j) = \frac{1}{i+1} \sum_{\ell \in \mathcal{P}_i} \delta_{\mathbf{F}_\ell} + \frac{1}{i+1} \delta_{\mathbf{F}_j} , \quad j = 1, \ldots, n$$

Algorithm 3 MMD subsampling

```
Input: Empirical measure $\mu$, kernel $k$, subsample size $m$
Output: Subsampled measure $\mu'$
$\pi_1 \leftarrow \mathrm{argmin}_{j=1\cdots n_0} \mathrm{MMD}_k^2(\mu, \delta_{\mathbf{F}_j})$
$\mathcal{P}_1 = \{\pi_1\}$
for $i = 1, \cdots, m-1$ do
    $\pi_{i+1} \leftarrow \mathrm{argmin}_{j=1\cdots n_0} \mathrm{MMD}_k^2(\mu, \mu_{i+1}(j))$
    $\mathcal{P}_{i+1} \leftarrow \mathcal{P}_i \cup \{\pi_{i+1}\}$
end for
$\mu' \leftarrow \frac{1}{m} \sum_{j \in \mathcal{P}_m} \delta_{\mathbf{F}_j}$
```

TOS-GP: more experimental details

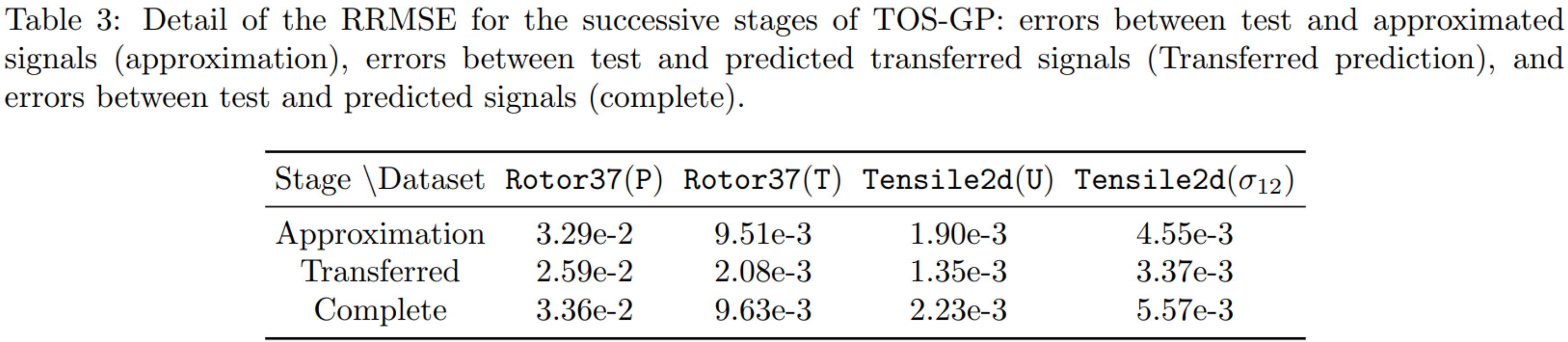


Table 3: Detail of the RRMSE for the successive stages of TOS-GP: errors between test and approximated signals (approximation), errors between test and predicted transferred signals (Transferred prediction), and errors between test and predicted signals (complete).

Stage \Dataset	Rotor37(P)	Rotor37(T)	Tensile2d(U)	Tensile2d(σ_{12})
Approximation	3.29e-2	9.51e-3	1.90e-3	4.55e-3
Transferred	2.59e-2	2.08e-3	1.35e-3	3.37e-3
Complete	3.36e-2	9.63e-3	2.23e-3	5.57e-3

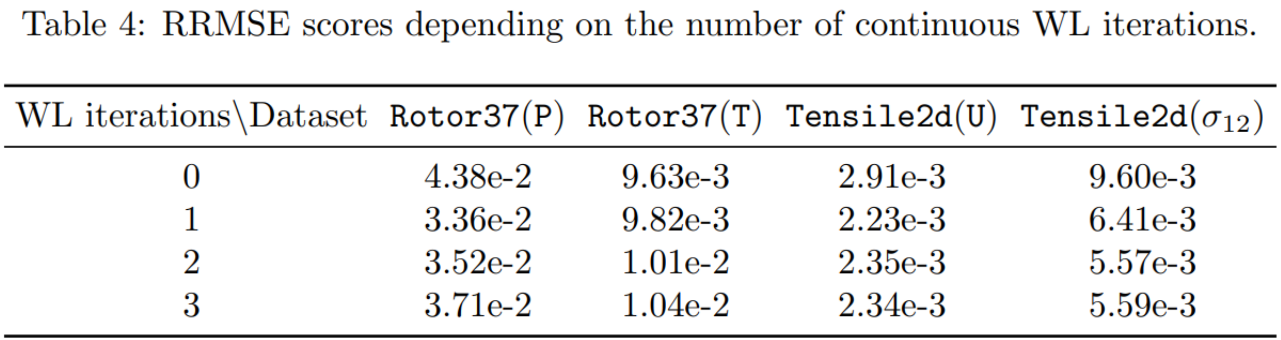


Table 4: RRMSE scores depending on the number of continuous WL iterations.

WL iterations\Dataset	Rotor37(P)	Rotor37(T)	Tensile2d(U)	Tensile2d(σ_{12})
0	4.38e-2	9.63e-3	2.91e-3	9.60e-3
1	3.36e-2	9.82e-3	2.23e-3	6.41e-3
2	3.52e-2	1.01e-2	2.35e-3	5.57e-3
3	3.71e-2	1.04e-2	2.34e-3	5.59e-3

POWERED BY TRUST