Some industrialized approaches in physics-based machine learning

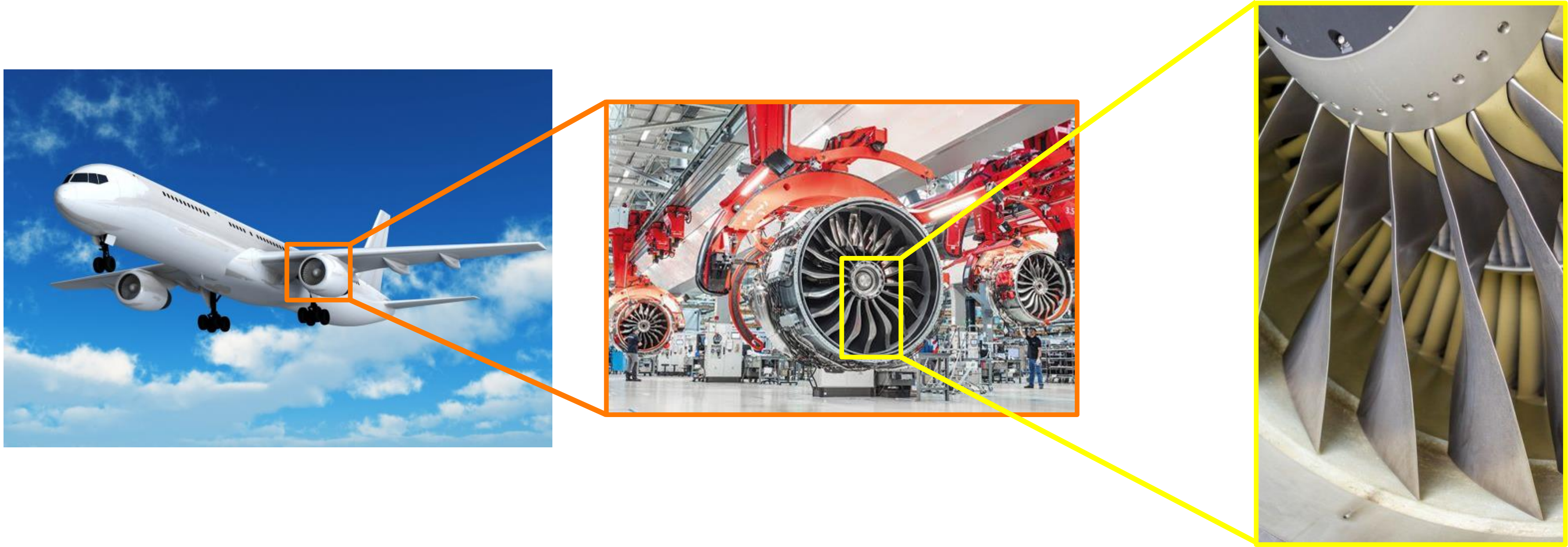
Raphaël CARPINTERO PEREZ

Abbas KABALAN

20/06/2025

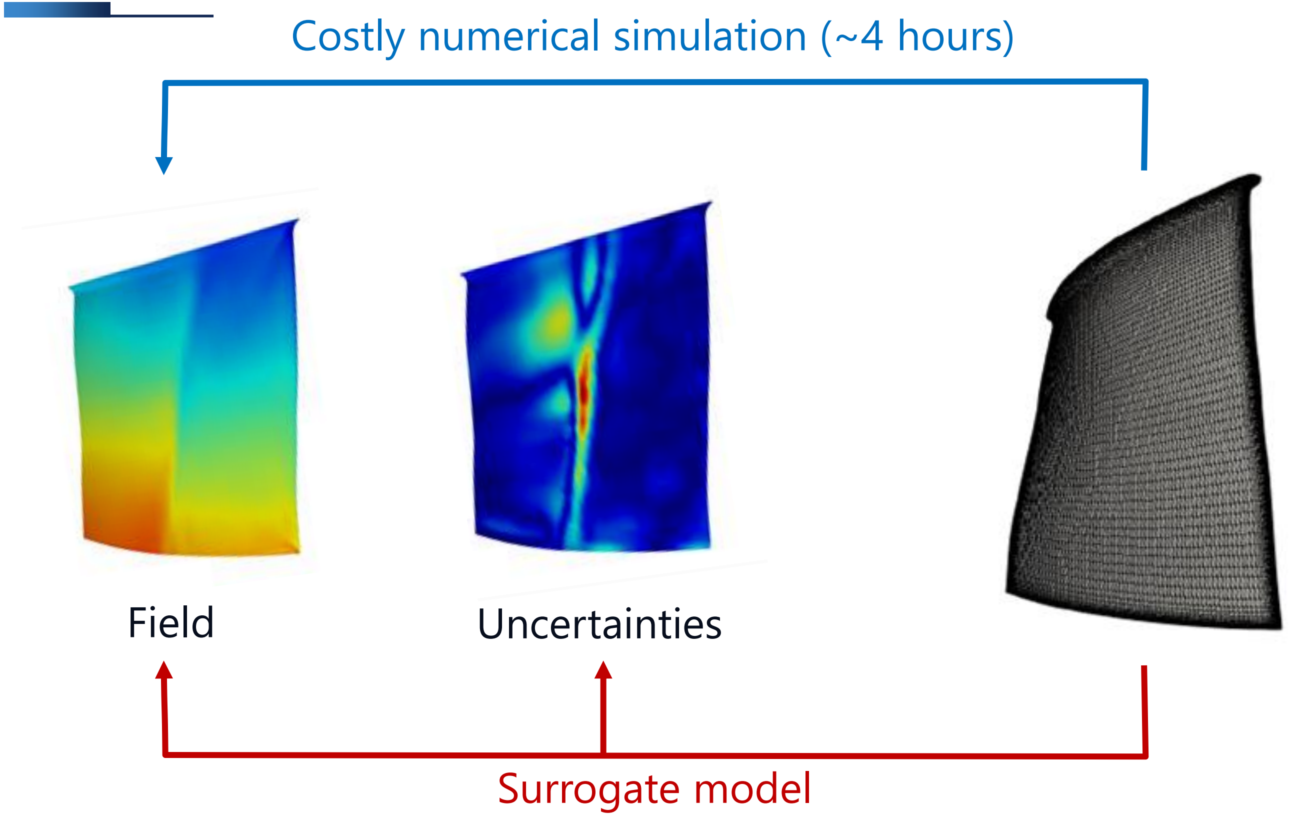
SAFRAN

Objectives



Turbine blades

Objectives



Costly numerical simulation (~4 hours)

Field

Uncertainties

Surrogate model

Turbine blades

Summary

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

- **Raphaël CARPINTERO PEREZ**

- Sébastien DA VEIGA
- Josselin GARNIER
- Brian STABER

- **Morphing techniques for reduced-order modeling under geometrical variability**

- **Abbas KABALAN**

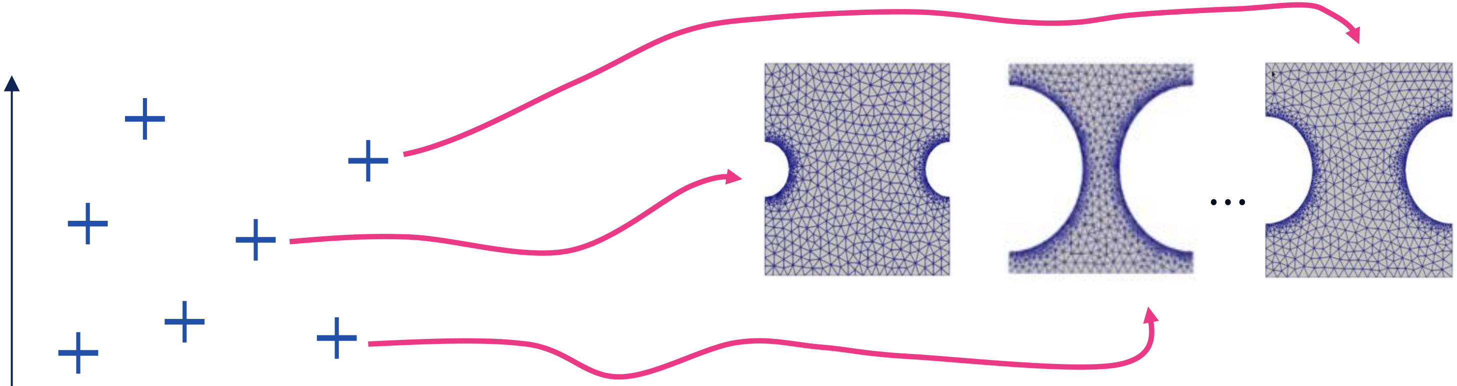
- Virginie EHRLACHER
- Alexandre ERN

- Fabien CASENAVE
- Felipe BORDEU

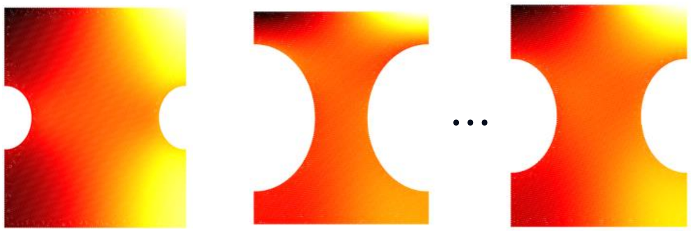
Supervised learning context

Step 1: Generate a design of experiments

Step 2: Create the meshes



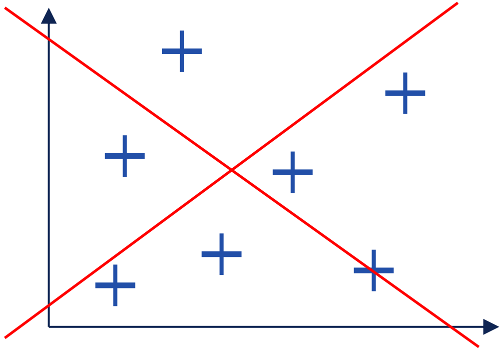
Step 3: Finite-element solver



c characteristic mesh parameters (curvature, cord length, $\cdots$)

b boundary/external conditions

Supervised learning context

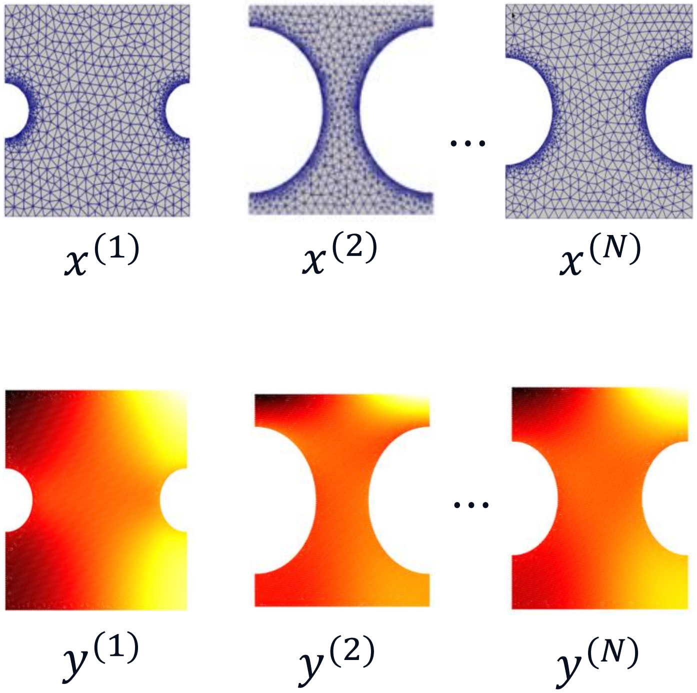


~~c characteristic mesh parameters (curvature, cord length, …)~~

b boundary/external conditions

Meshes

Fields



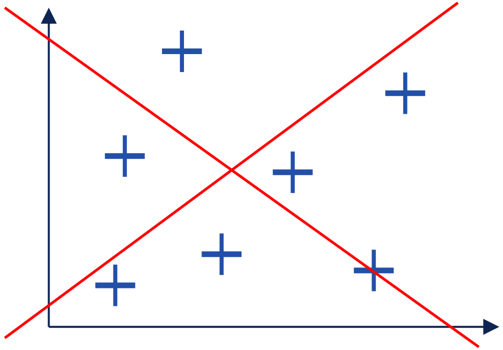
$x^{(1)}$ $x^{(2)}$ … $x^{(N)}$

Inputs

$y^{(1)}$ $y^{(2)}$ … $y^{(N)}$

Outputs

Supervised learning context

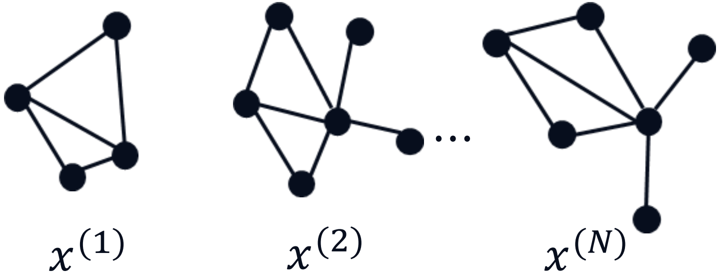


~~c characteristic mesh parameters (curvature, cord length, …)~~

b boundary/external conditions

~~Meshes~~

Graphs



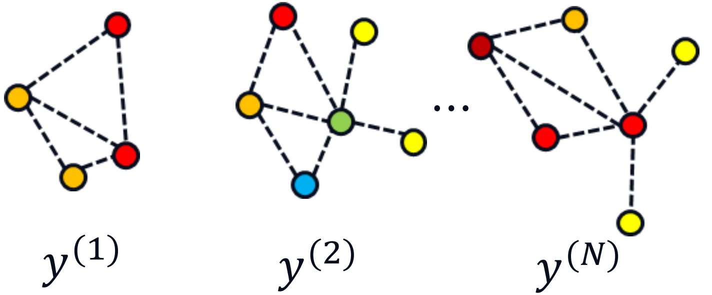
$\in \mathcal{X}$

Inputs

<u>Step 3</u>: Finite-element solver

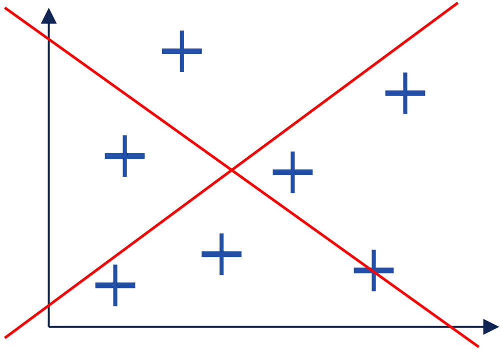
~~Fields~~

Signals



Outputs

Supervised learning context

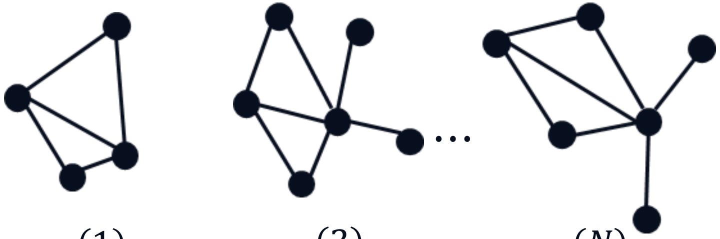


~~c characteristic mesh parameters (curvature, cord length, …)~~

b boundary/external conditions

~~Meshes~~

Graphs



$$x^{(1)} \qquad x^{(2)} \qquad \cdots \qquad x^{(N)} \qquad \in \mathcal{X}$$

Inputs

Scalars

$$\begin{array}{cccc} \mathbb{R} & \mathbb{R} & \cdots & \mathbb{R} \\ \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} \\ y^{(1)} & y^{(2)} & & y^{(N)} \end{array}$$

Outputs

I) Scalar outputs

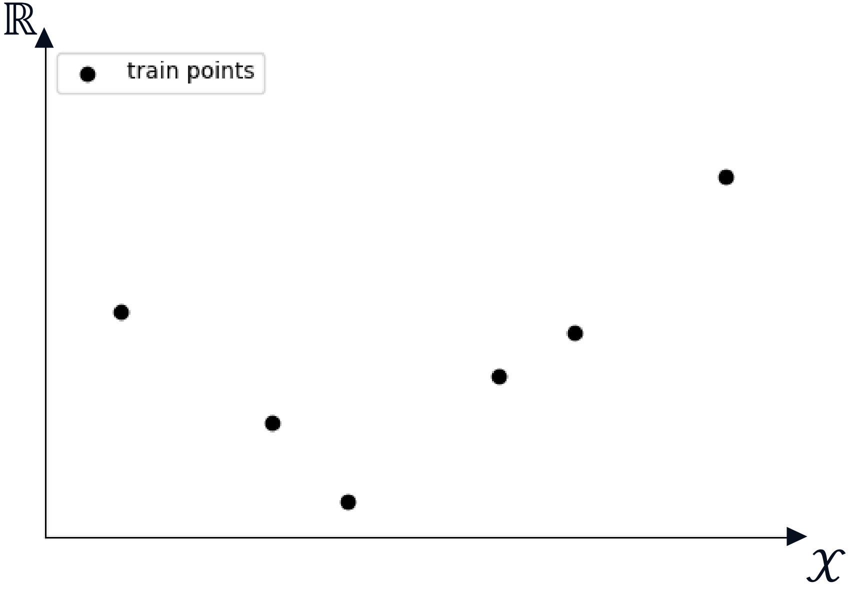
1- Gaussian process regression
2- Graph kernels
3- SWWL graph kernel

II) Signal outputs

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

Regression

<u>Objective:</u> Learn $f: \mathcal{X} \rightarrow \mathbb{R}$ from a set of (noisy) observations $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$

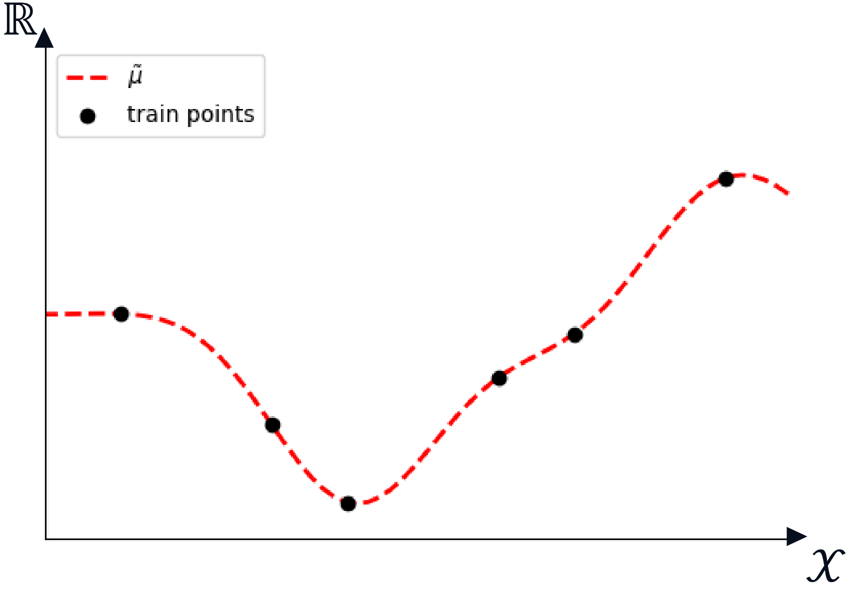


Regression

Objective: Learn $f: \mathcal{X} \rightarrow \mathbb{R}$ from a set of (noisy) observations $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$

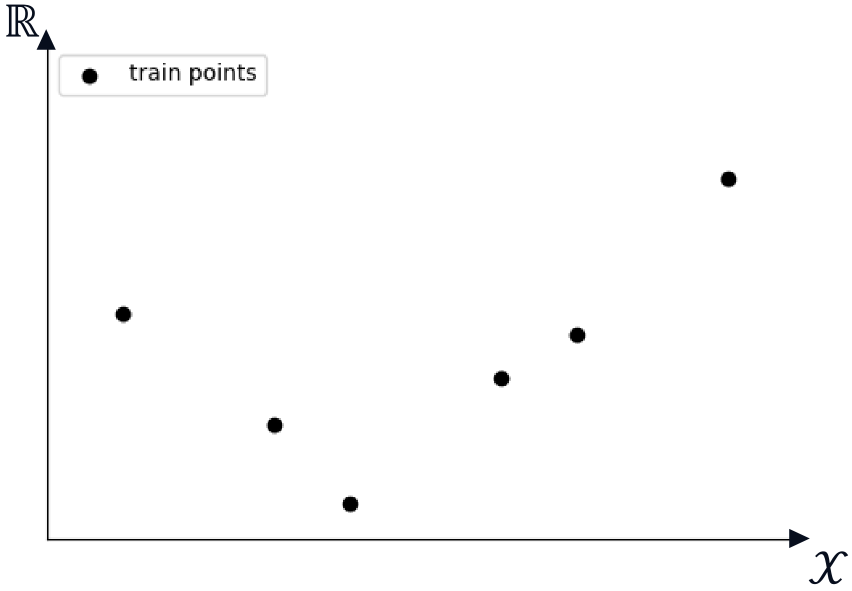
$$f^* \in \underset{f \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - f(x_i)\right)^2 + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2$$

Minimize a penalized loss function (e.g. quadratic) on the train set



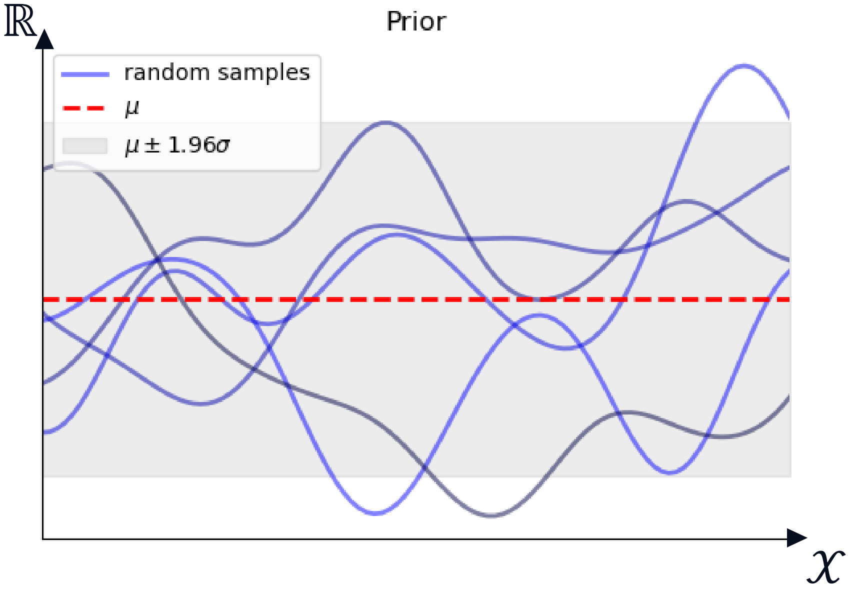
Gaussian process Regression

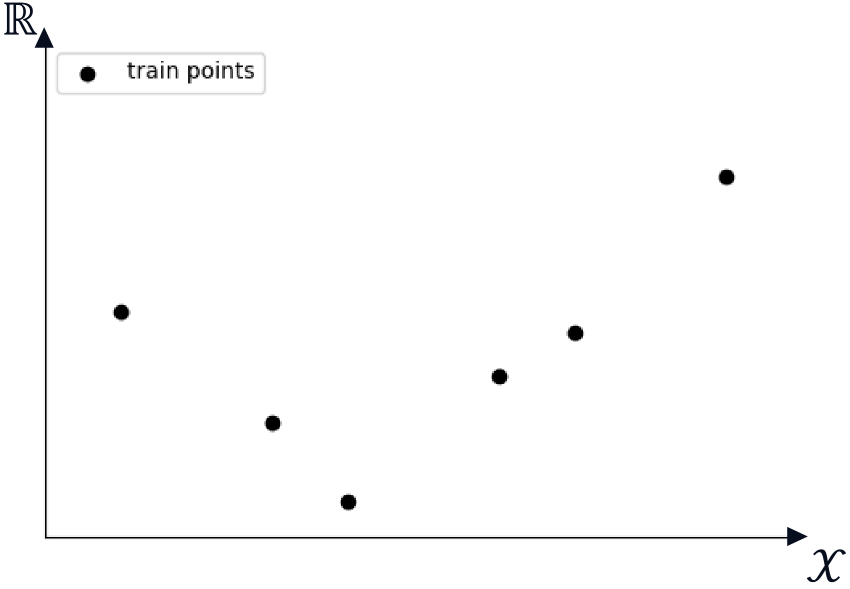
<u>Objective:</u> Learn $f: \mathcal{X} \rightarrow \mathbb{R}$ from a set of (noisy) observations $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$



Gaussian process Regression

Objective: Learn $f: \mathcal{X} \to \mathbb{R}$ from a set of (noisy) observations $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$



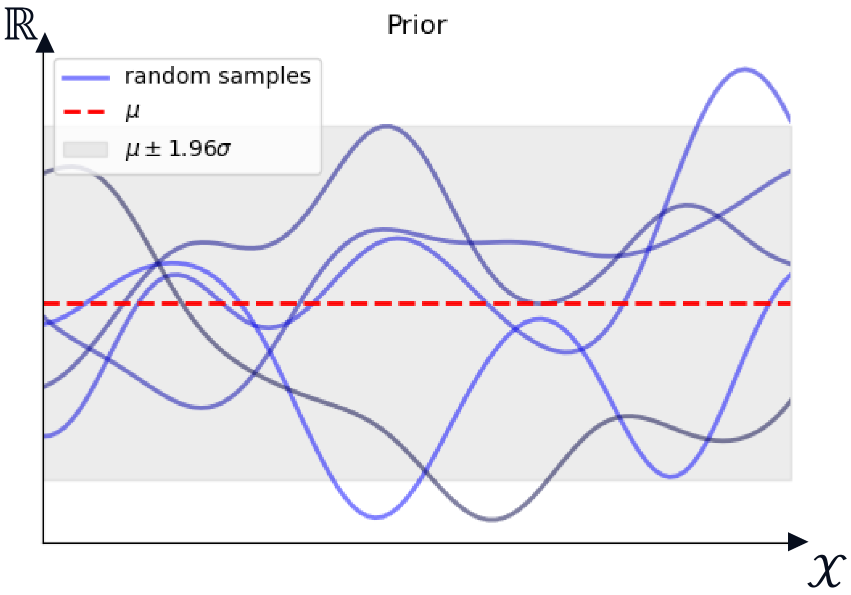


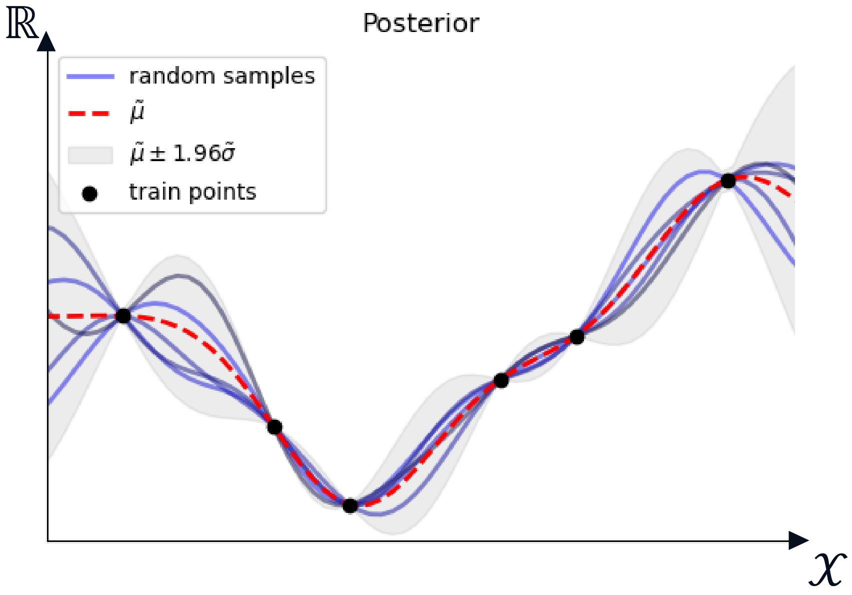
$f \sim \mathcal{GP}(\mu, k)$ where $k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a symmetric **positive definite kernel**

$$\sigma(x) = \sqrt{k(x, x)}$$

Gaussian process Regression

<u>Objective:</u> Learn $f: \mathcal{X} \to \mathbb{R}$ from a set of (noisy) observations $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$





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

$$\sigma(x) = \sqrt{k(x,x)}$$

$$f \mid \mathcal{D} \sim \mathcal{GP}\left(\tilde{\mu}, \tilde{k}\right)$$

Gaussian process Regression

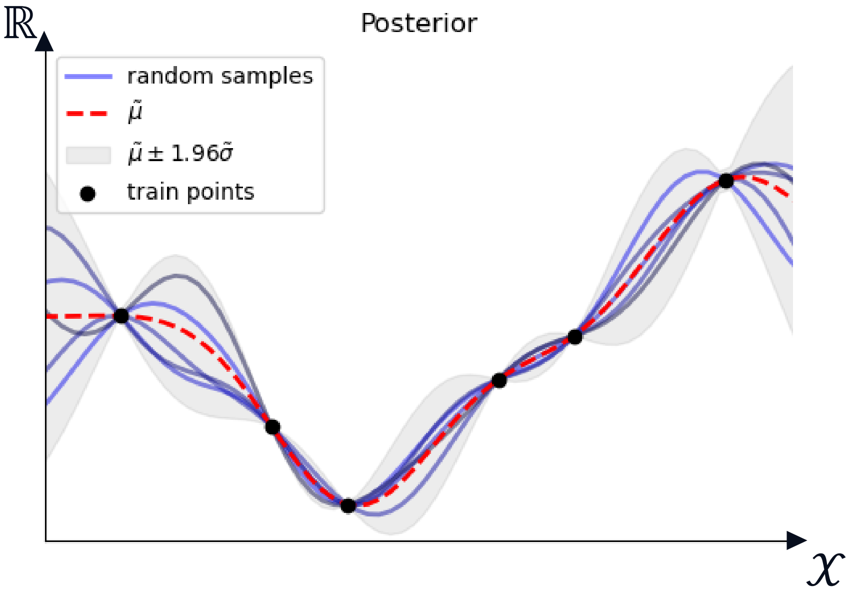
Objective: Learn $f: \mathcal{X} \rightarrow \mathbb{R}$ from a set of (noisy) observations $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$

Choice of the kernel $k: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$?

When $\mathcal{X} = \mathbb{R}^d$:

$$k(x, x') = e^{-\lambda \|x - x'\|^2} \quad \text{(RBF)}$$

$\dots$



$$f \mid \mathcal{D} \sim \mathcal{GP}(\tilde{\mu}, \tilde{k})$$

Gaussian process Regression

Objective: Learn $f: \mathcal{X} \to \mathbb{R}$ from a set of (noisy) observations $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$

Choice of the kernel $k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$?

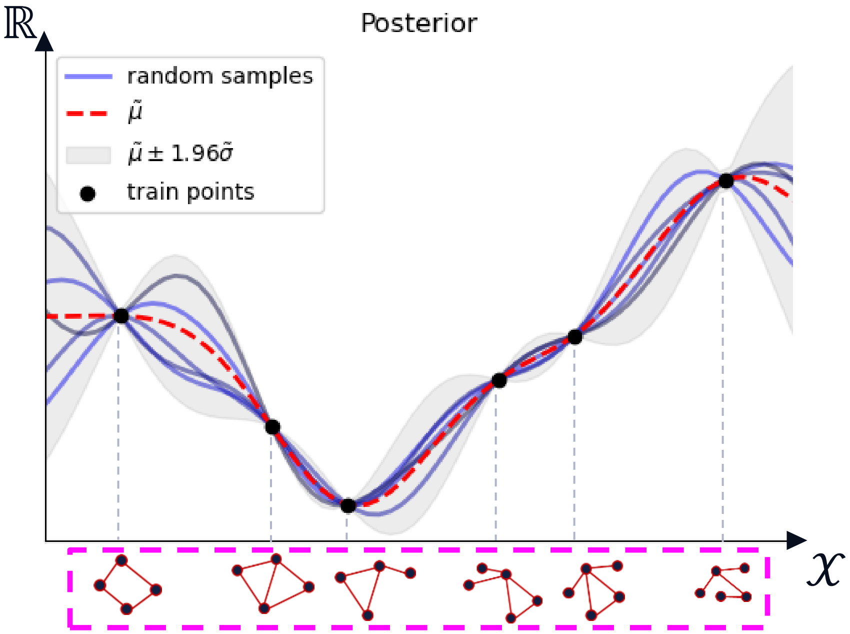
When $\mathcal{X} = \mathbb{R}^d$:

$$k(x, x') = e^{-\lambda \|x - x'\|^2} \quad \text{(RBF)}$$

$\ldots$

When $\mathcal{X} = \mathcal{G}$ is a space of graphs:

k (,) = ?



$$f \mid \mathcal{D} \sim \mathcal{GP}(\tilde{\mu}, \tilde{k})$$

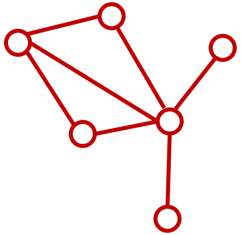
I) Scalar outputs

1- Gaussian process regression
2- Graph kernels
3- SWWL graph kernel

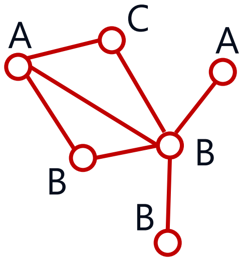
II) Signal outputs

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

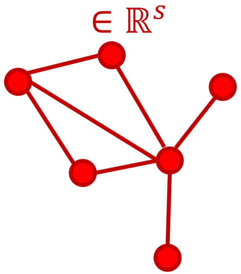
What is a graph ?



Case 1 :
Vertices + Edges

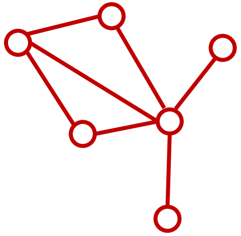


Case 2 :
Vertices + Edges
+ Node labels

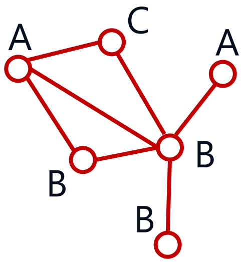


Case 3 :
Vertices + Edges
+ Node attributes

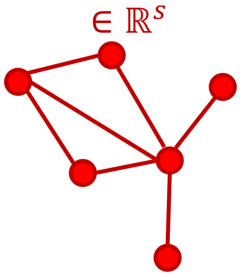
What is a graph ?



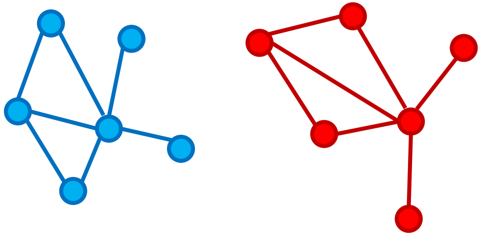
Case 1 :
Vertices + Edges



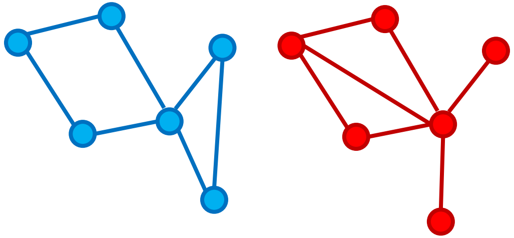
Case 2 :
Vertices + Edges
+ Node labels



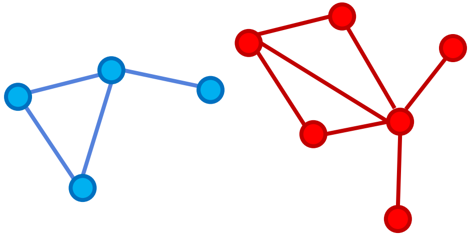
Case 3 :
Vertices + Edges
+ Node attributes



Case 3A: Fixed structure -> signal

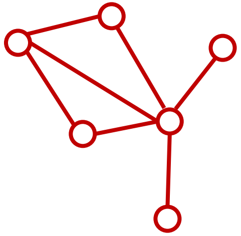


Case 3B: Fixed number of nodes

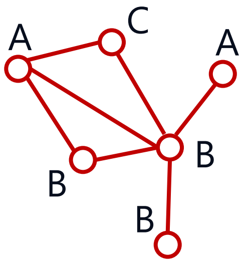


Case 3C: Varying number of
nodes + structure + attributes

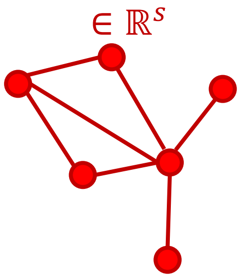
What is a graph ?



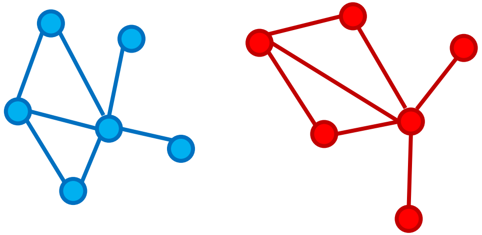
Case 1 :
Vertices + Edges



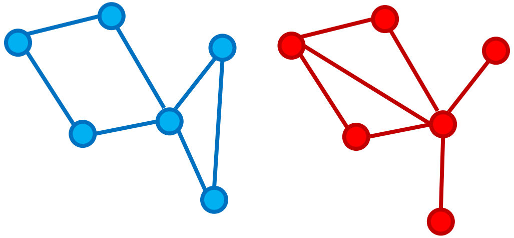
Case 2 :
Vertices + Edges
+ Node labels



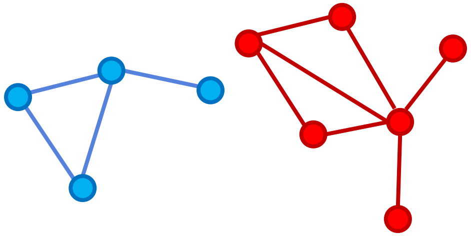
Case 3 :
Vertices + Edges
+ Node attributes



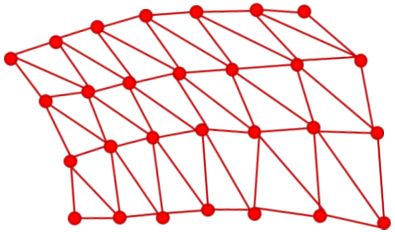
Case 3A: Fixed structure -> signal



Case 3B: Fixed number of nodes



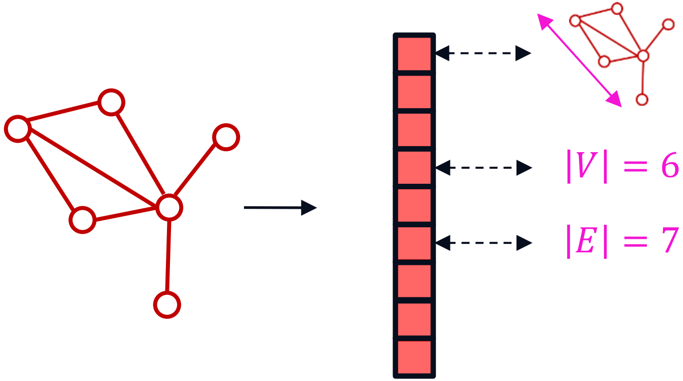
Case 3C: Varying number of
nodes + structure + attributes



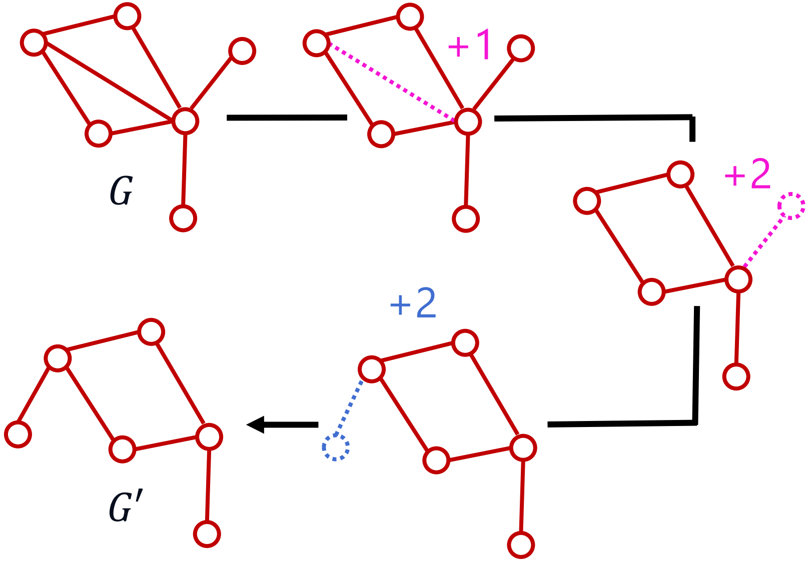
Case 3C+: Varying number of
nodes + structure + attributes
+ large-scale + sparse

Graph kernels (1/3): Early attempts

Invariants / Topological descriptors



Graph edit distance



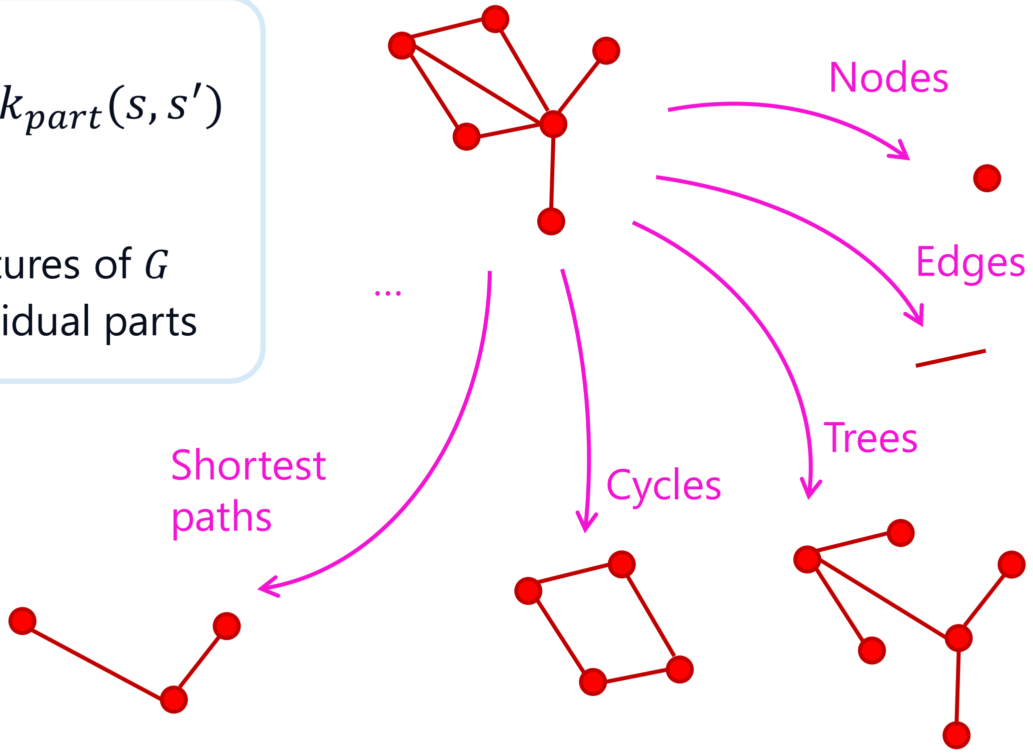
Complete graph invariants: equal for two graphs iif they are isomorphic

→ (Gartner 2003) require exponential runtime

Graph kernels (2/3): $\mathcal{R}$-convolution framework

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

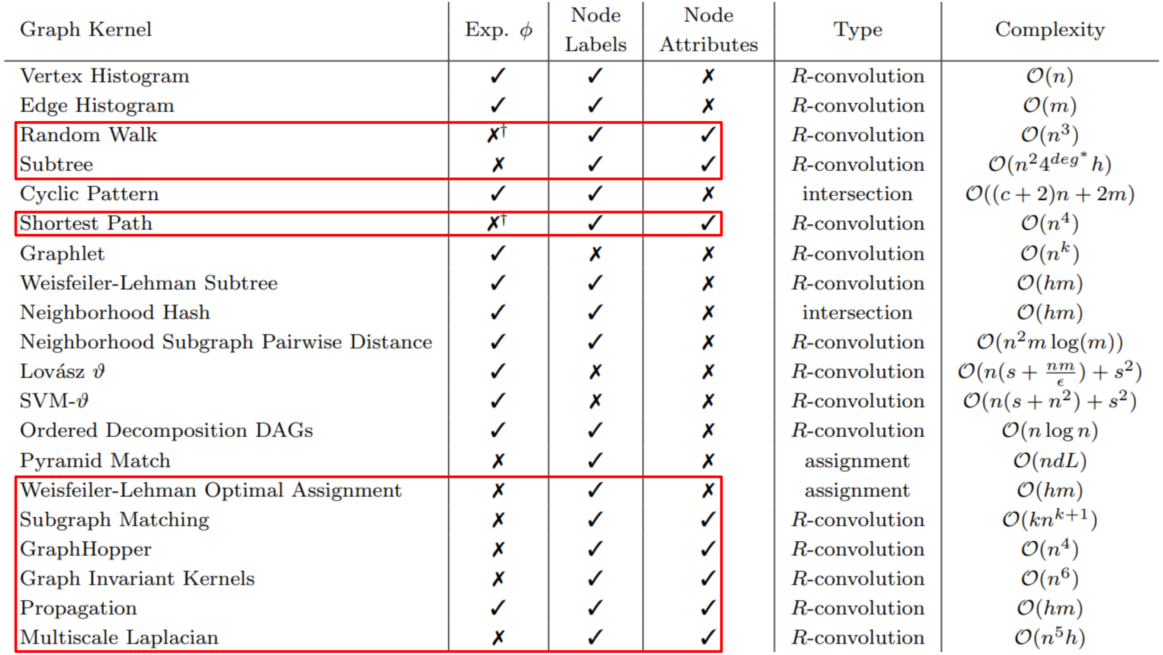
- $S(G)$: set of parts/substructures of G
- k_{part}: kernel between individual parts



Graph kernels (3/3): checklist

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	$\mathcal{O}(n^5 h)$

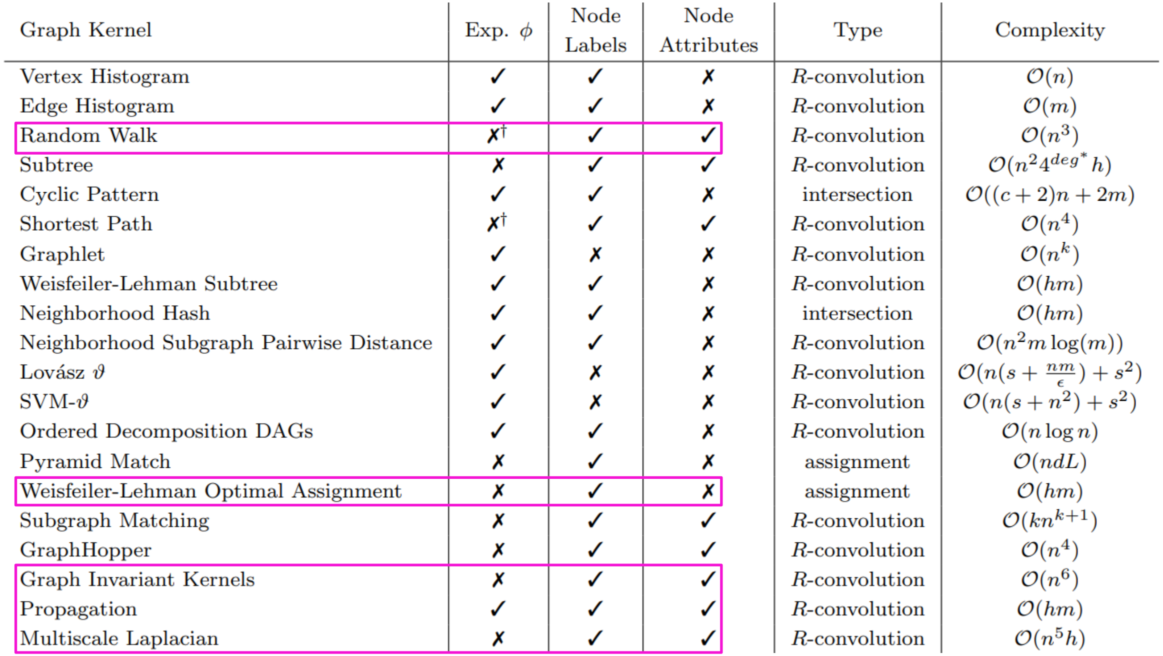
[Nikolentzos et al., 2021]

Graph kernels (3/3): checklist

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]

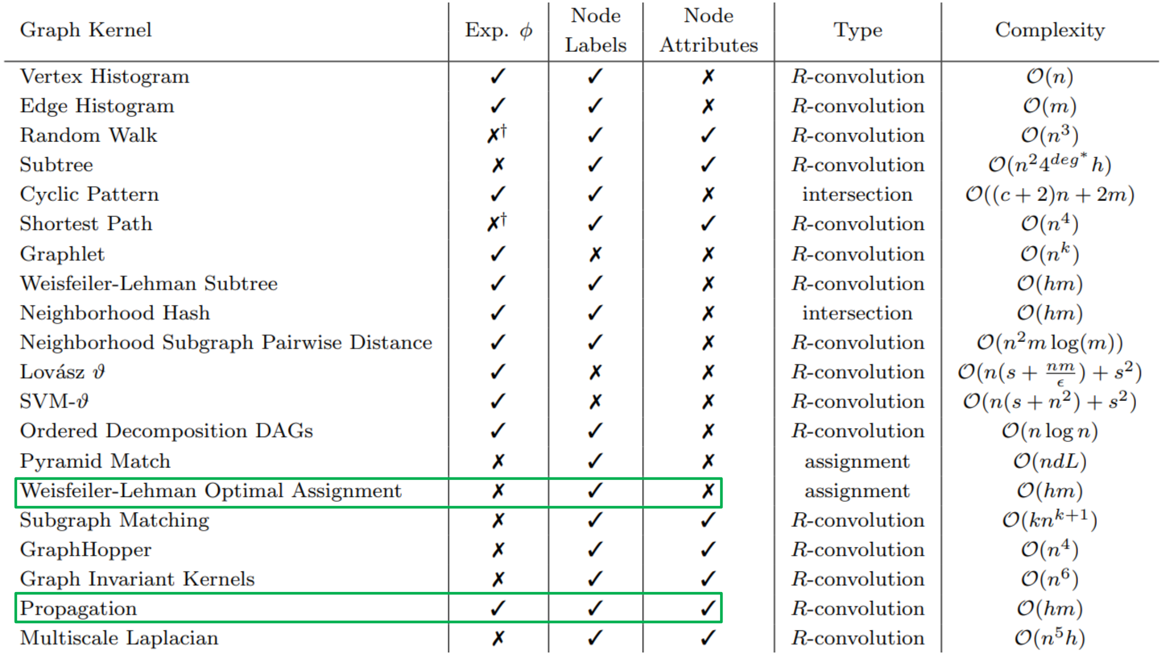
Graph kernels (3/3): checklist

Checklist:

✓ continuous node attributes

✓ no relying heavily on the graph structure

✓ tractable



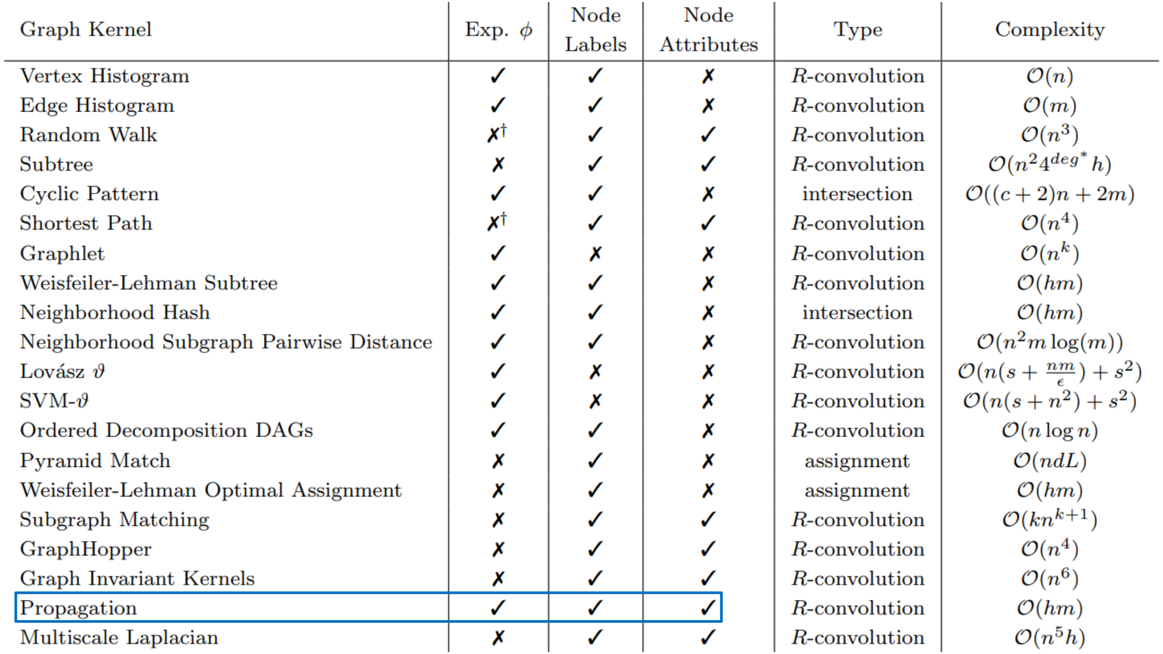
Graph Kernel	Exp. ϕ	Node Labels	Node Attributes	Type	Complexity
Vertex Histogram	✓	✓	✗	R-convolution	$\mathcal{O}(n)$
Edge Histogram	✓	✓	✗	R-convolution	$\mathcal{O}(m)$
Random Walk	✗†	✓	✓	R-convolution	$\mathcal{O}(n^3)$
Subtree	✗	✓	✓	R-convolution	$\mathcal{O}(n^2 4^{deg^*} h)$
Cyclic Pattern	✓	✓	✗	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 (3/3): checklist

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]

I) Scalar outputs

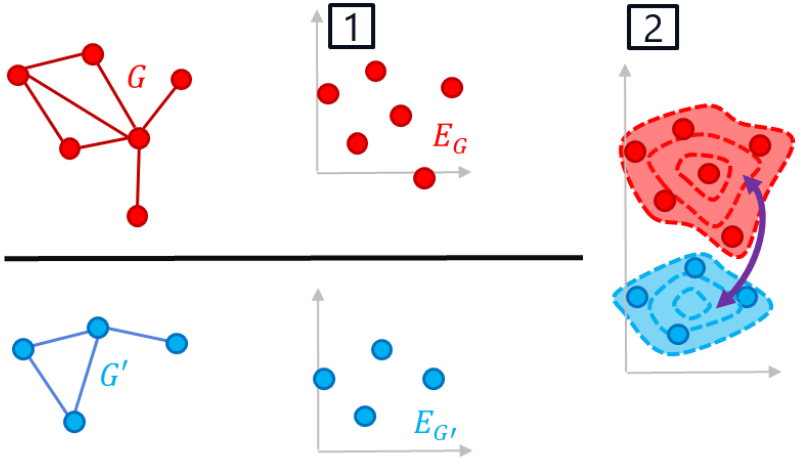
1- Gaussian process regression
2- Graph kernels
3- SWWL graph kernel

II) Signal outputs

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

Sliced Wasserstein Weisfeiler-Lehman graph kernels

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



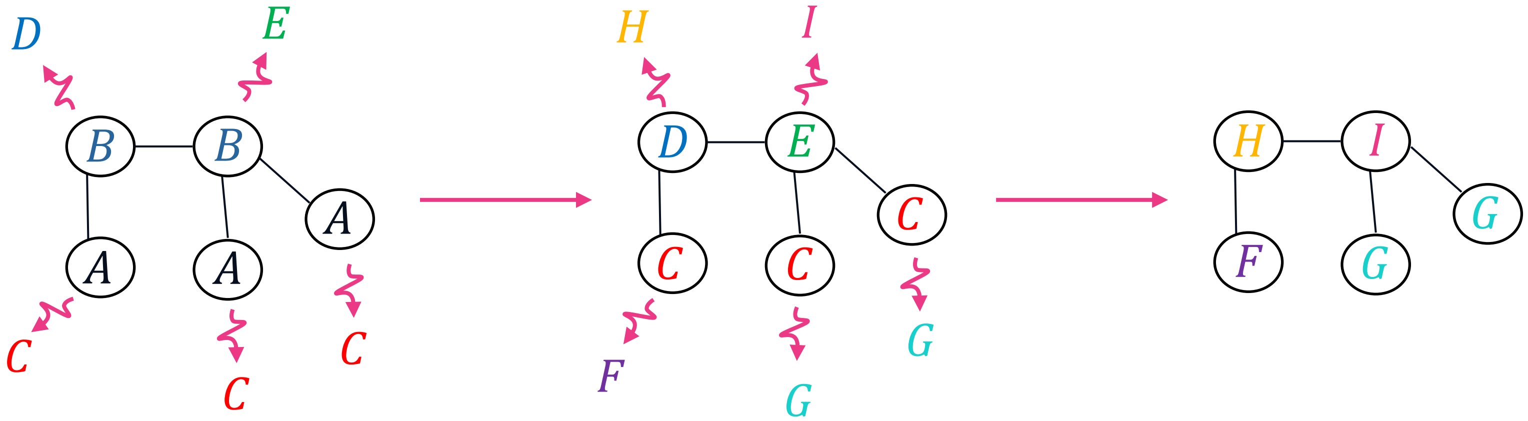
$\boxed{1}$ Embeddings of the graphs

$\boxed{2}$ Build a positive definite kernel between empirical measures

Weisfeiler-Lehman embeddings

Example from [Kriege et al., 2020]

- WL relabeling (categorical case)



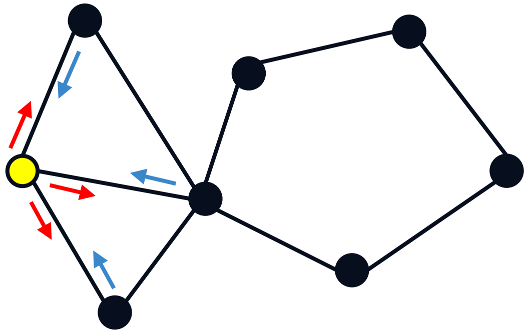
$$l^{(i+1)}(v) = Hash\left(l^i(v), \{l^i(u)\ , u \in \mathcal{N}(v)\}\right)$$

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

Continuous Weisfeiler-Lehman embeddings

[Togninalli et al., 2019]

- WL relabeling (continuous case)

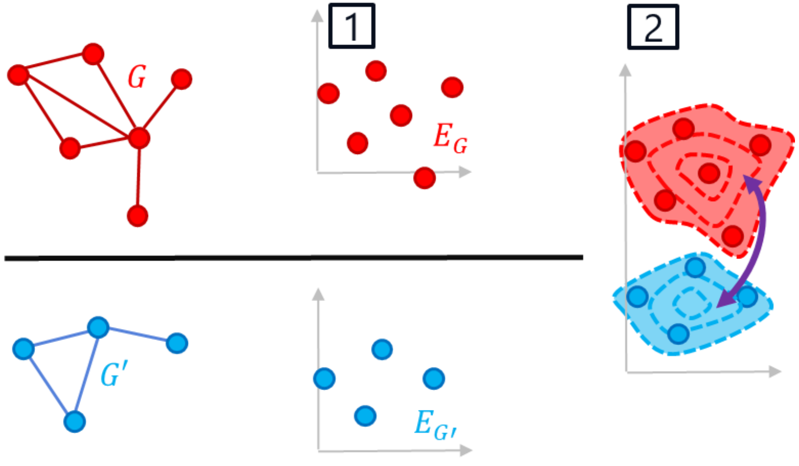


$$a^{(i+1)}(v) = \frac{1}{2}\left(a^{(i)}(v) + \frac{1}{\deg(v)} \sum_{u \in \mathcal{N}(v)} w(v,u)\, a^{(i)}(u)\right)$$

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

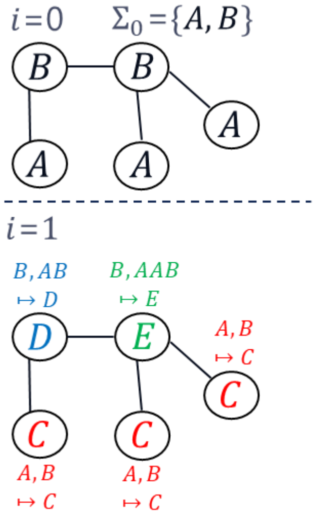
Sliced Wasserstein Weisfeiler-Lehman graph kernels

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



$\boxed{1}$ Embeddings of the graphs

$\boxed{2}$ Build a positive definite kernel between empirical measures



Continuous WL embeddings

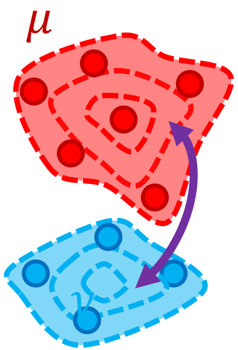
Wasserstein distance

Wasserstein distance

$$\mathcal{W}^2(\mu,\nu) = \inf_{\gamma\in\Pi(\mu,\nu)} \int_{\mathbb{R}^s\times\mathbb{R}^s} \|x-y\|^2 d\gamma(x,y),$$

Where:

- $s \in [1,+\infty)$,
- $\mathcal{P}_2(\mathbb{R}^s)$: probability measures on $\mathbb{R}^s$ with finite moments of order 2,
- $\Pi(\mu,\nu) = \{\pi \in \mathcal{P}_2(\mathbb{R}^s \times \mathbb{R}^s):\ (Proj_1)_{\#\pi} = \mu,\ (Proj_2)_{\#\pi} = \nu\}$



✘ $\mathcal{O}(\boldsymbol{n}^3\mathbf{log}(\boldsymbol{n}))$

✘ Substitution kernels are not positive definite in dimension $s \geq 2$

Sliced Wasserstein distance

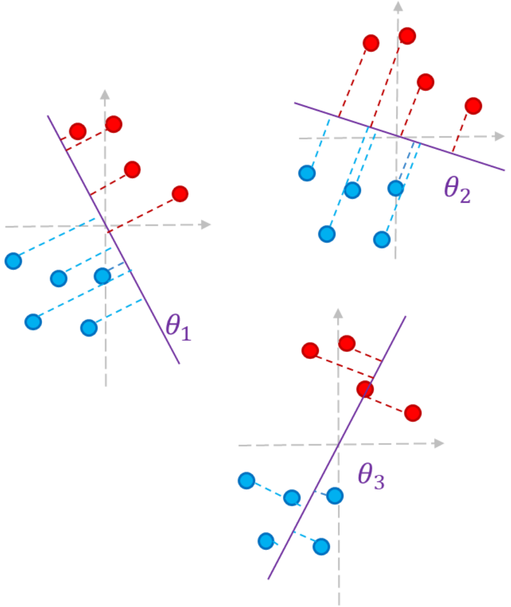
Sliced Wasserstein distance

[Bonneel et al. 2015]

$$\mathcal{SW}^2(\mu,\nu) = \int_{\mathbb{S}^{s-1}} \mathcal{W}^2(\theta^*_\#\mu, \theta^*_\#\nu)\mathrm{d}\sigma(\theta)$$

Where:

- $\mathbb{S}^{s-1}$: $(s-1)$-dimensional unit sphere,
- σ : uniform distribution on $\mathbb{S}^{s-1}$
- $\theta^*_\#\mu$: push-forward measure of $\mu \in \mathcal{P}_2(\mathbb{R}^s)$ by $\theta^* \begin{pmatrix} \mathbb{R}^s \to \mathbb{R} \\ x \mapsto \langle\theta, x\rangle \end{pmatrix}$



✓ Complexity: scales as $\boldsymbol{n}\,\mathbf{log}(\boldsymbol{n})$

✓ Positive definite substitution kernels

Sliced Wasserstein distance

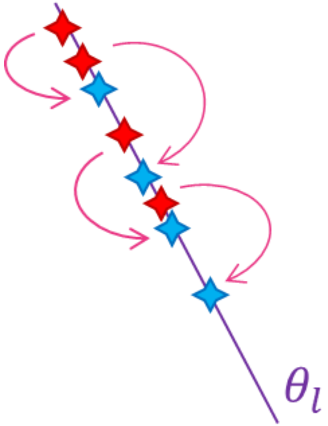
Sliced Wasserstein distance

[Bonneel et al. 2015]

$$\mathcal{W}^2(\theta_{\#}^{*}\mu, \theta_{\#}^{*}\nu) = \int_0^1 \underbrace{|F^{-1}(\mu) - F^{-1}(\nu)|^2}_{\text{Quantile function}} \, \mathrm{d}t$$

Where:

- $\mathbb{S}^{s-1}$: $(s-1)$-dimensional unit sphere,
- σ : uniform distribution on $\mathbb{S}^{s-1}$
- $\theta_{\#}^{*}\mu$: push-forward measure of $\mu \in \mathcal{P}_2(\mathbb{R}^s)$ by $\theta^* \begin{pmatrix} \mathbb{R}^s \to \mathbb{R} \\ x \mapsto \langle \theta, x \rangle \end{pmatrix}$



✓ Complexity: scales as $\boldsymbol{n}\,\mathbf{log}(\boldsymbol{n})$

✓ Positive definite substitution kernels

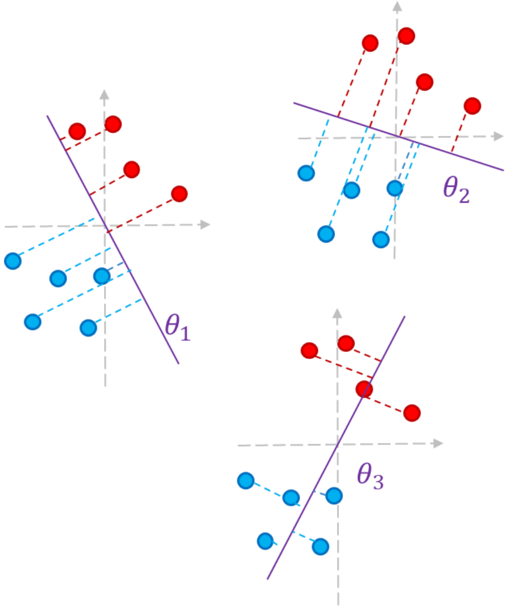
Estimation of the Sliced Wasserstein distance (1/2)

1) Monte Carlo samples for the projections

P projections $\theta_1, \cdots, \theta_P$

$$\mathcal{SW}^2(\mu,\nu) = \int_{\mathbb{S}^{s-1}} \mathcal{W}^2(\theta^*_\# \mu, \theta^*_\# \nu) \mathrm{d}\sigma(\theta)$$

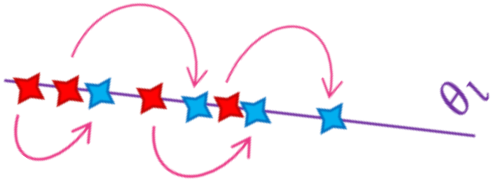
$$\mathcal{SW}^2(\mu,\nu) \simeq \frac{1}{P} \sum_{p=1}^{P} \mathcal{W}^2 \left(\left(\theta_p^*\right)_\# \mu, \left(\theta_p^*\right)_\# \nu \right)$$



Estimation of the Sliced Wasserstein distance (2/2)

2) Fixed quantiles

A) If μ and ν have the same support size

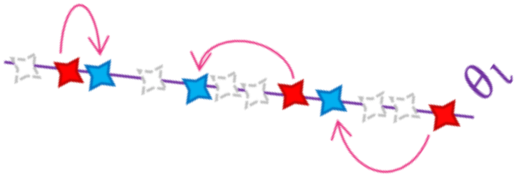


$$\mu = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i} \qquad \nu = \frac{1}{n}\sum_{i=1}^{n} \delta_{y_i}$$

$$\mathcal{W}^2(\mu,\nu) = \frac{1}{n}\sum_{i=1}^{n} \left|x_{(i)} - y_{(i)}\ \right|^2$$

B) If μ and ν have different support sizes

Q quantile levels common to all inputs



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

$$W^2(\mu,\nu) \simeq \frac{1}{Q}\sum_{i=1}^{Q} \left|x_{(i)} - y_{(i)}\right|^r$$

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

Sliced Wasserstein Weisfeiler Lehman (SWWL)

SWWL kernel

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

$\mu_G := \frac{1}{|V|}\sum_{i=1}^{n}(E_G)_i$: continuous WL embedding of G

$$k_{SWWL}(G, G') = e^{-\lambda \widehat{\mathcal{SW}^2}(\mu_G, \mu_{G'})}$$

Distance substitution kernel

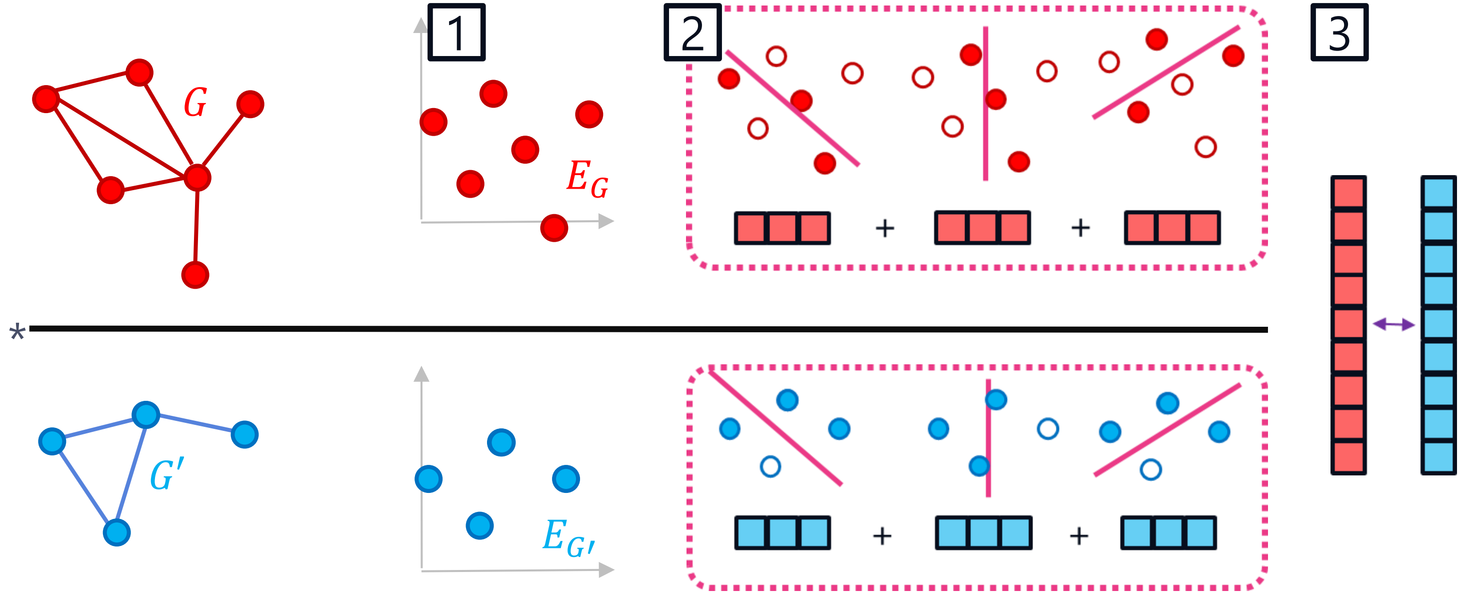
$$\widehat{\mathcal{SW}_2^2}(\mu_G, \mu_{G'}) = \frac{1}{\mathrm{PQ}}\sum_{\mathrm{p}=1}^{\mathrm{P}}\sum_{q=1}^{Q}\left|u_q^{\theta_p} - u'^{\theta}_q\right|^2 = \|U_G - U_{G'}\|_2^2$$

where $u_q^{\theta_p} = \langle \theta_p, E_G \rangle_{(q)}$

$$U_G = [u_1^{\theta_1}, \cdots, u_Q^{\theta_1}, \cdots, u_1^{\theta_P}, \cdots, u_Q^{\theta_P}]$$

Embeddings in $\mathbb{R}^{PQ}$

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

Sliced Wasserstein Weisfeiler Lehman (SWWL)

SWWL kernel

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

$$\boxed{\mu_G := \frac{1}{|V|}\sum_{i=1}^{n}(E_G)_i \quad : \text{continuous WL embedding of } G}$$

$$k_{SWWL}(G,G') = \boxed{e^{-\lambda\widehat{\mathcal{SW}^2}(\mu_G,\mu_{G'})}}$$

$$\widehat{\mathcal{SW}_2^2}(\mu_G,\mu_{G'}) = \frac{1}{\mathrm{PQ}}\sum_{\mathrm{p}=1}^{\mathrm{P}}\sum_{q=1}^{Q}\left|u_q^{\theta_p} - u'^{\theta}_q\right|^2 = \|U_G - U_{G'}\|_2^2$$

where $u_q^{\theta_p} = \langle\theta_p, E_G\rangle_{(q)}$

$U_G = [u_1^{\theta_1},\cdots,u_Q^{\theta_1},\cdots,u_1^{\theta_P},\cdots,u_Q^{\theta_P}]$

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

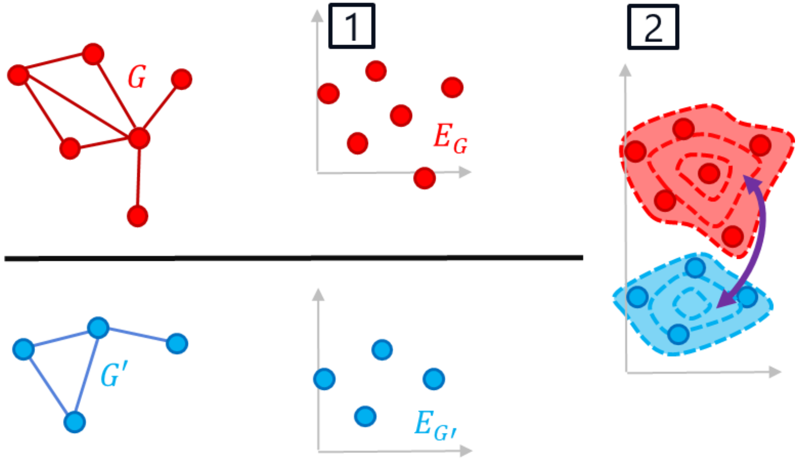
P: number of projections

Q: number of quantiles

H: number of WL iterations

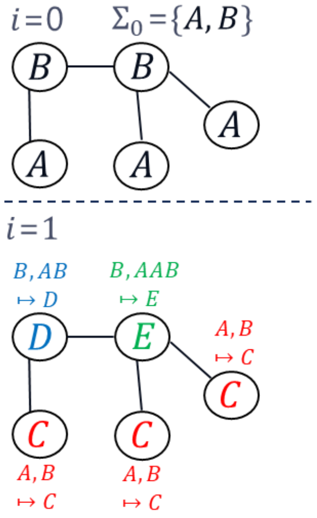
Sliced Wasserstein Weisfeiler-Lehman graph kernels

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



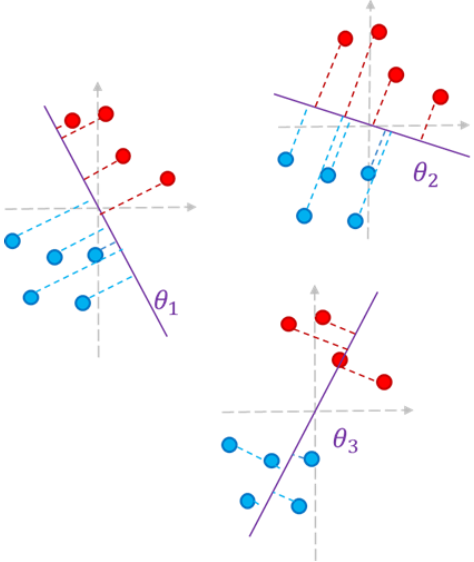
- ✓ Complexity: scales as $\boldsymbol{n}\,\mathbf{log}(\boldsymbol{n})$
- ✓ Positive definite substitution kernels

$\boxed{1}$ Embeddings of the graphs



Continuous WL embeddings

$\boxed{2}$ Build a positive definite kernel between empirical measures



Sliced Wasserstein

I) Scalar outputs

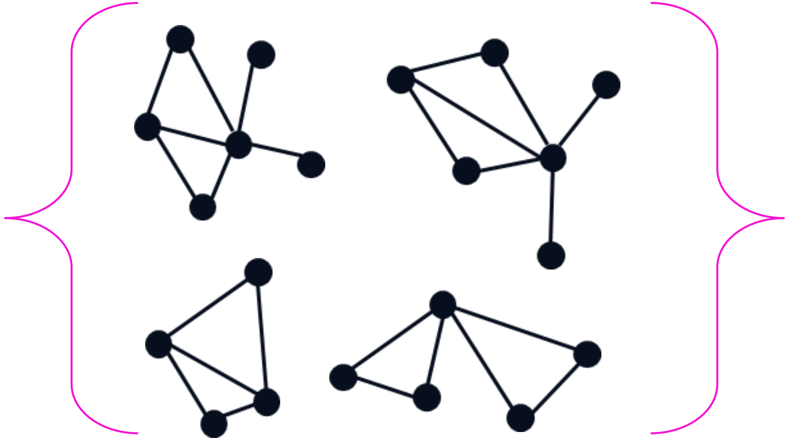
1- Gaussian process regression
2- Graph kernels
3- SWWL graph kernel

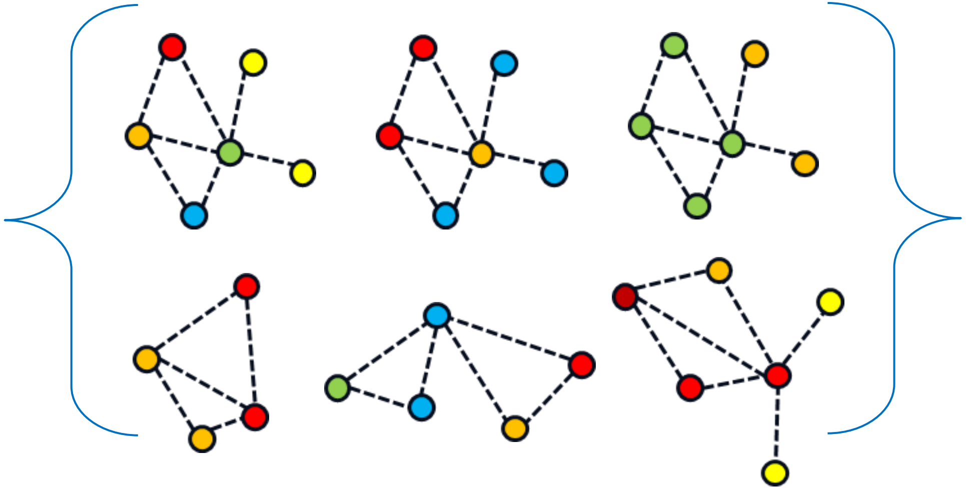
II) Signal outputs

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

Learning output fields/signals

Learn $f : \mathcal{X} \to \mathcal{Y}$ from a train dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1,\cdots,N}$



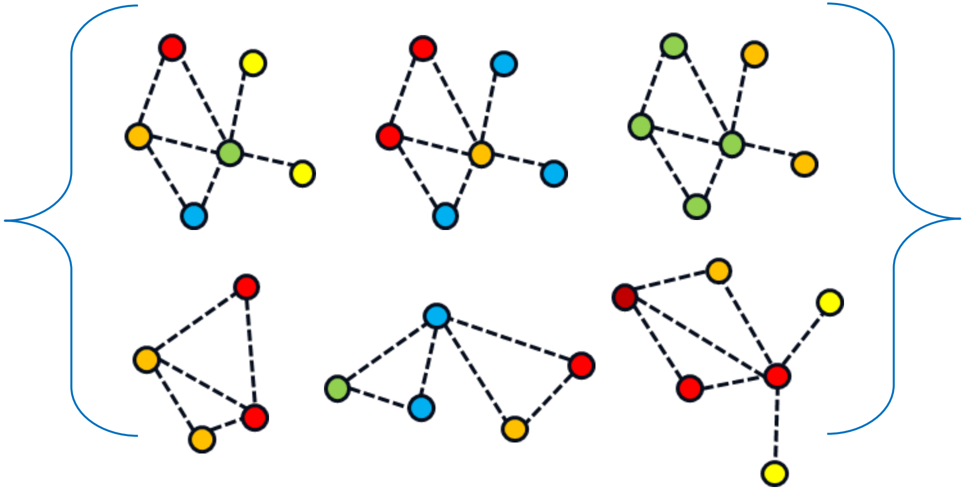


$$\mathcal{Y} = \bigcup_{x=(V,E,w,F)\in\mathcal{X}} \{y: V \to \mathbb{R}\}$$

Learning output fields/signals

Learn $f : \mathcal{X} \rightarrow \mathcal{Y}$ from a train dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1,\cdots,N}$

- Inputs can have different sizes, so do the outputs
- No natural ordering of the output scalar elements
- The number of output elements can be very large



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

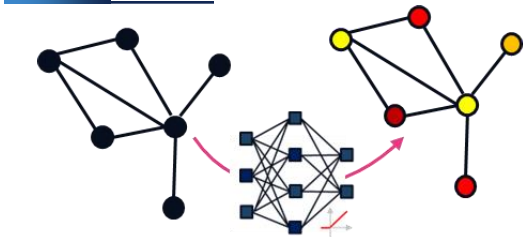
I) Scalar outputs

1- Gaussian process regression
2- Graph kernels
3- SWWL graph kernel

II) Signal outputs

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

Related approaches



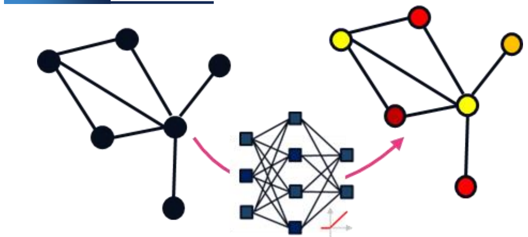
Graph Neural Networks

✓ Signal prediction [Pfaff, 2020]

✗ No uncertainties

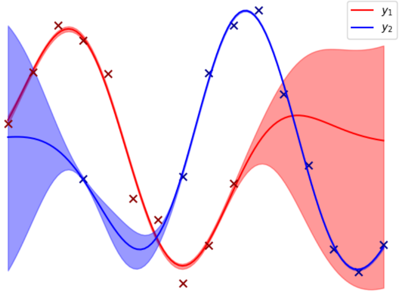
✗ Training time

Related approaches



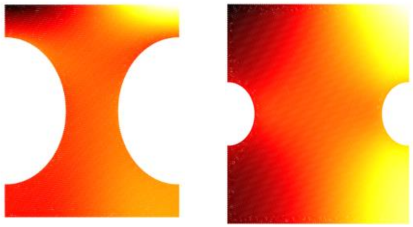
Graph Neural Networks

- ✓ Signal prediction [Pfaff, 2020]
- ✗ No uncertainties
- ✗ Training time

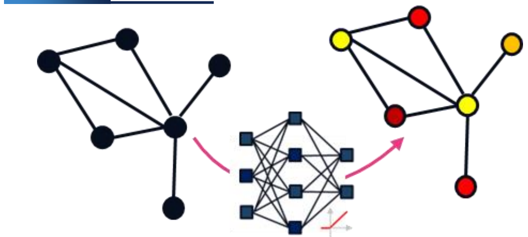


Multi/Functional Output GPs

- ✗ No ordering of the output elements
- ✗ Varying domains [Goovaerts, 1997]

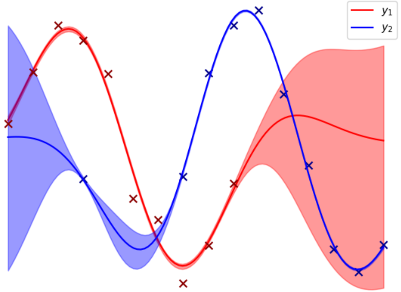


Related approaches



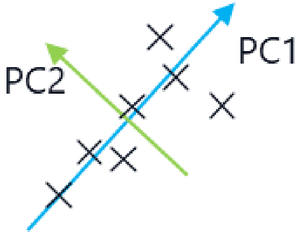
Graph Neural Networks

- ✓ Signal prediction [Pfaff, 2020]
- ✗ No uncertainties
- ✗ Training time



Multi/Functional Output GPs

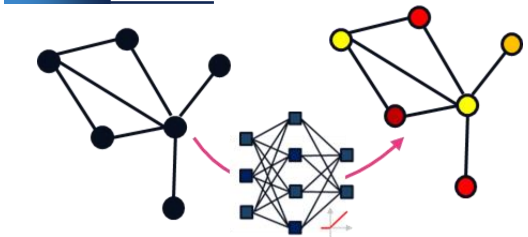
- ✗ No ordering of the output elements
- ✗ Varying domains [Goovaerts, 1997]



Dimension reduction

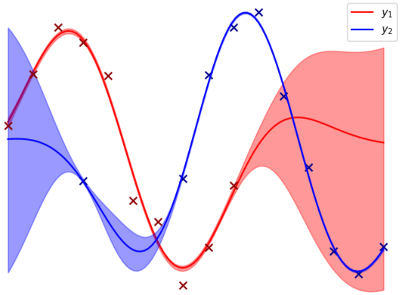
- ✗ No ordering of the output elements [Kontolati, 2022]

Related approaches



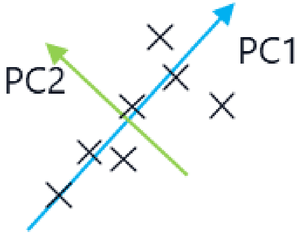
Graph Neural Networks

- ✓ Signal prediction [Pfaff, 2020]
- ✗ No uncertainties
- ✗ Training time



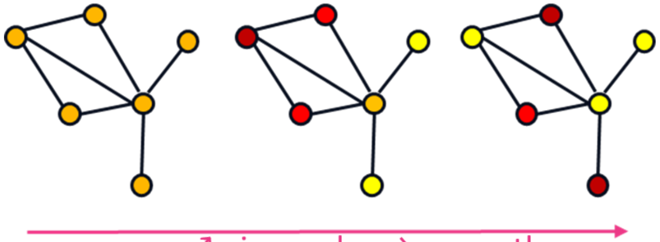
Multi/Functional Output GPs

- ✗ No ordering of the output elements
- ✗ Varying domains [Goovaerts, 1997]



Dimension reduction

- ✗ No ordering of the output elements [Kontolati, 2022]

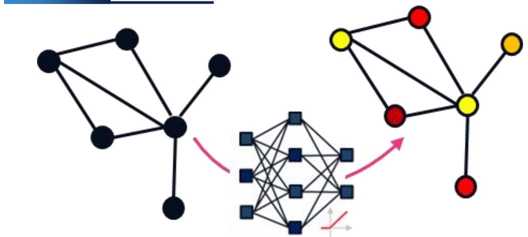


↗ eigenvalue, ↘ smoothness

Graph signal processing [Ortega, 2018]

- ✗ Incomparable eigendecompositions

Related approaches

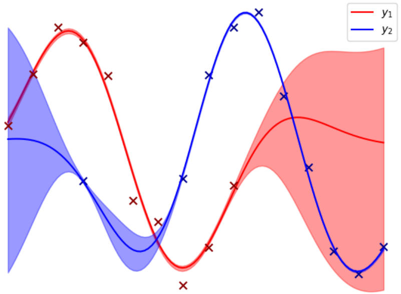


Graph Neural Networks

✓ Signal prediction [Pfaff, 2020]

✘ No uncertainties

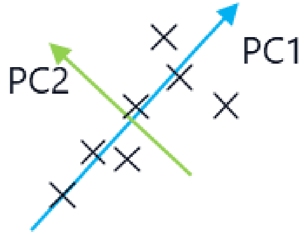
✘ Training time



Multi/Functional Output GPs

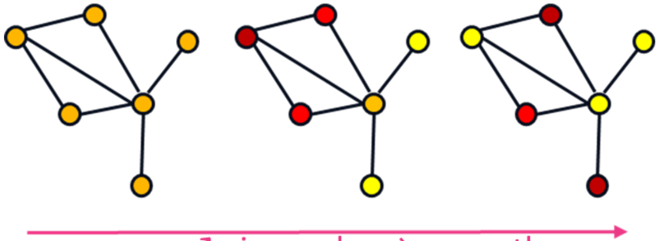
✘ No ordering of the output elements

✘ Varying domains [Goovaerts, 1997]



Dimension reduction

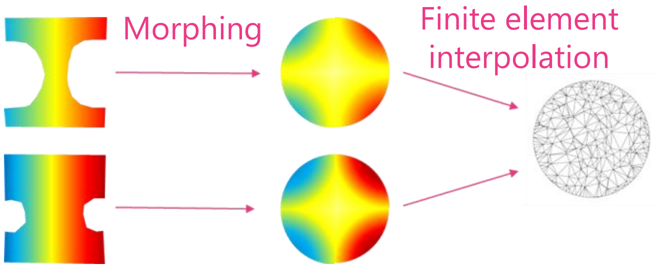
✘ No ordering of the output elements [Kontolati, 2022]



↗ eigenvalue, ↘ smoothness

Graph signal processing [Ortega, 2018]

✘ Incomparable eigendecompositions



Morphing

Finite element interpolation

Mesh Morphing Gaussian Processes

✓ Prediction + uncertainties [Casenave, 2024]

✘ Specific to meshes + same topology

I) Scalar outputs

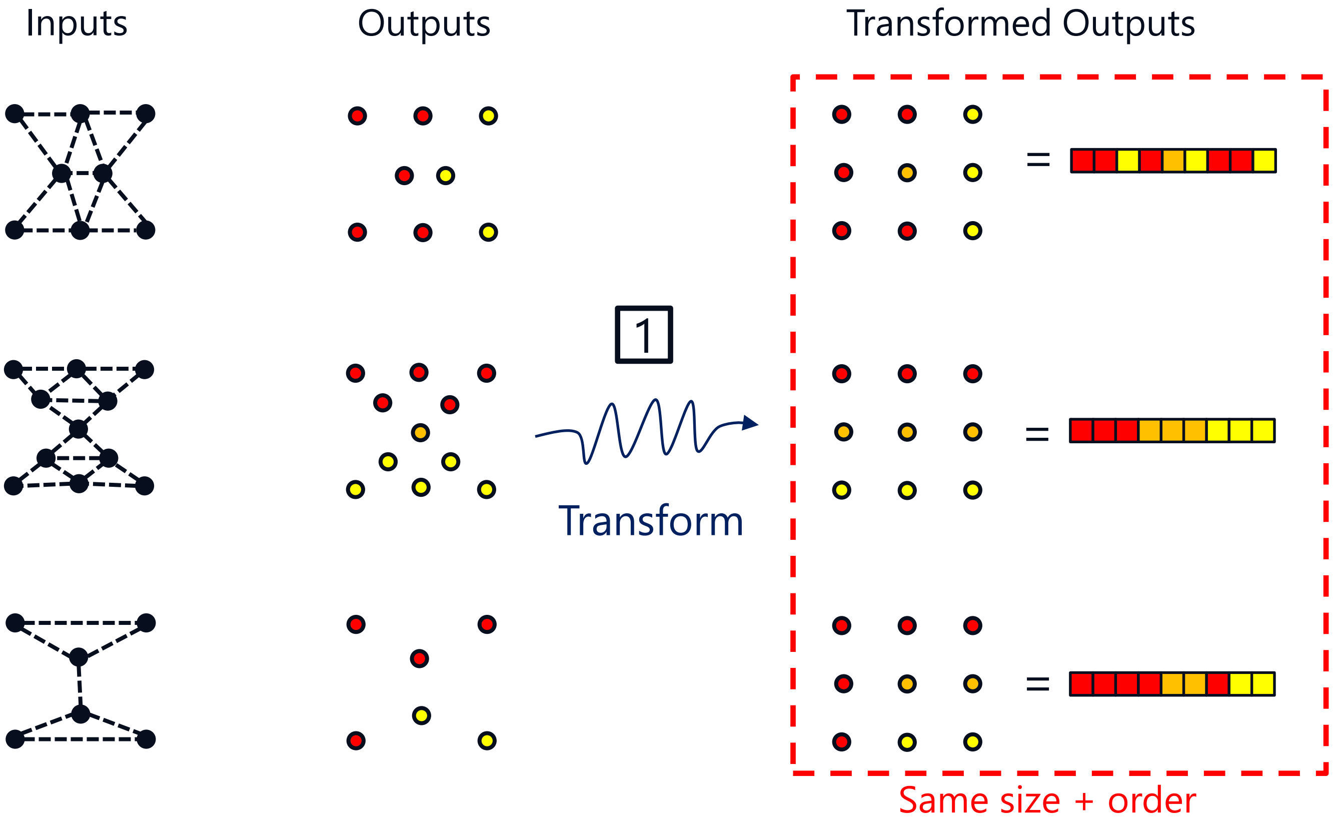
1- Gaussian process regression
2- Graph kernels
3- SWWL graph kernel

II) Signal outputs

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

SAFRAN

Express signals/fields in the same space?



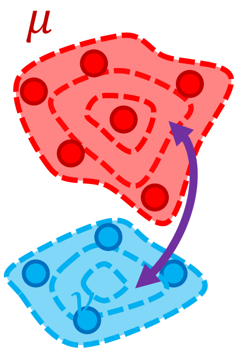
Wasserstein distance

Wasserstein distance

$$\mathcal{W}^2(\mu,\nu) = \inf_{\gamma\in\Pi(\mu,\nu)} \int_{\mathbb{R}^s\times\mathbb{R}^s} \|x-y\|^2 d\gamma(x,y),$$

Where:

- $s \in [1,+\infty)$,
- $\mathcal{P}_2(\mathbb{R}^s)$: probability measures on $\mathbb{R}^s$ with finite moments of order 2,
- $\Pi(\mu,\nu) = \{\pi \in \mathcal{P}_2(\mathbb{R}^s \times \mathbb{R}^s) \colon \ (Proj_1)_{\#\pi} = \mu, \ (Proj_2)_{\#\pi} = \nu\}$



Wasserstein distance

Wasserstein distance (discrete case)

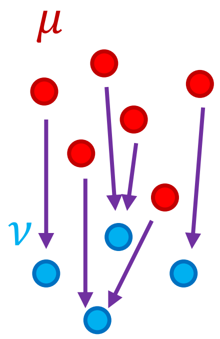
$$\mathcal{W}^2(\mu,\nu) = \min_{P\in U(n,n')} \langle C^{\mu,\nu}, P\rangle$$

Transport plan

Cost matrix

Where:

- $\mu = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i} \qquad \nu = \frac{1}{n'}\sum_{i=1}^{n'} \delta_{z_i}$
- $U(n,n') = \left\{P \in \mathbb{R}_+^{n\times n'} : P\mathbb{1}_{n'} = \frac{1}{n}\mathbb{1}_n, P\mathbb{1}_n = \frac{1}{n'}\mathbb{1}_{n'}\right\}$
- $C^{\mu,\nu} = \left[\left\|x_i - z_j\right\|^2\right]_{i=1\ldots n,\ j=1\ldots n\prime}$



Wasserstein distance

Regularized Wasserstein distance

[Peyré & Cuturi, 2019]

$$\mathcal{W}_\lambda^2(\mu,\nu) = \min_{P\in U(n,n')} \langle C^{\mu,\nu}, P\rangle - \lambda H(P), \quad \lambda > 0$$

Entropic regularization

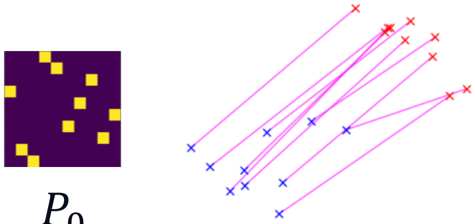
$$L_\lambda(\mu,\nu,P) = \langle C^{\mu,\nu}, P \rangle - \lambda H(P)$$

$$\boxed{P_\lambda = \underset{P\in U(n,n')}{argmin} \; L_\lambda(\mu,\nu,P)}$$

Smoothed transport plan

1- Without regularization

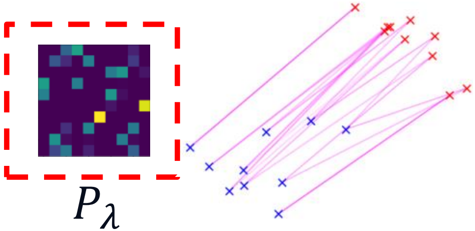
$\lambda = 0$



P_0

2- With regularization

$\lambda > 0$



P_λ

✓ Smoothing of the transport plans

✓ Sinkhorn: $O(n^2 \log(n))$

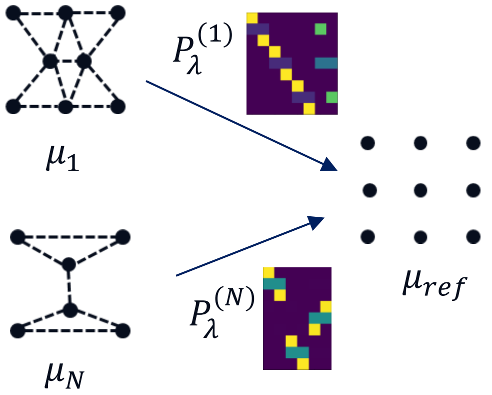
Transferring fields with transport plans

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

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

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

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



Transferring fields with transport plans

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

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

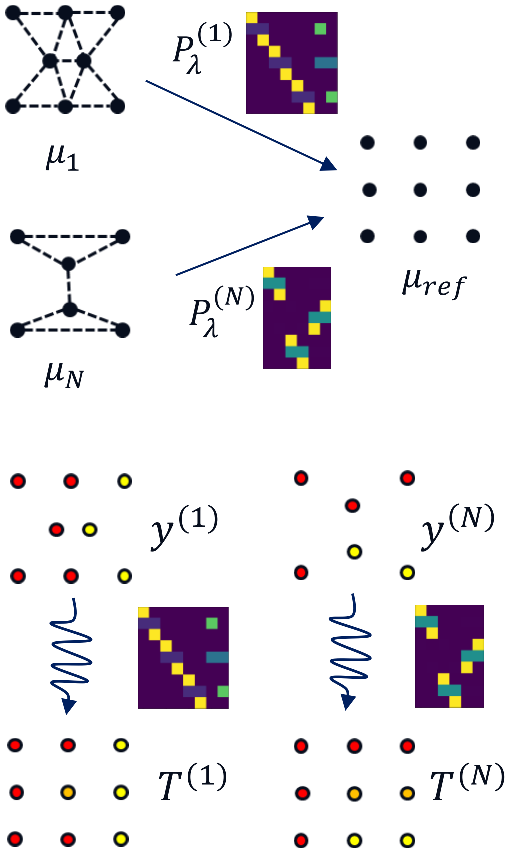
$\mu_i = \frac{1}{n_i}\sum_{j=1}^{n_i} \delta_{[\phi_{WL}(G^{(i)})]_j}$: WL embeddings of input graph i

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

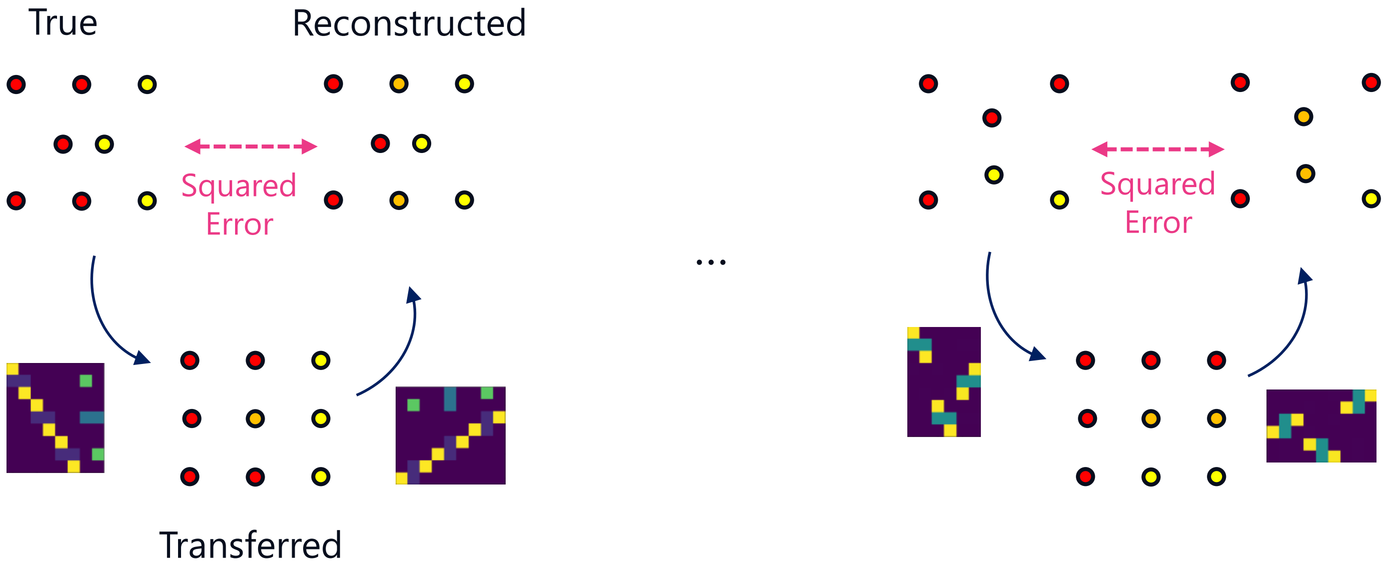
Part 2: transferring **output** signals

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

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



How to choose the regularization parameter ?

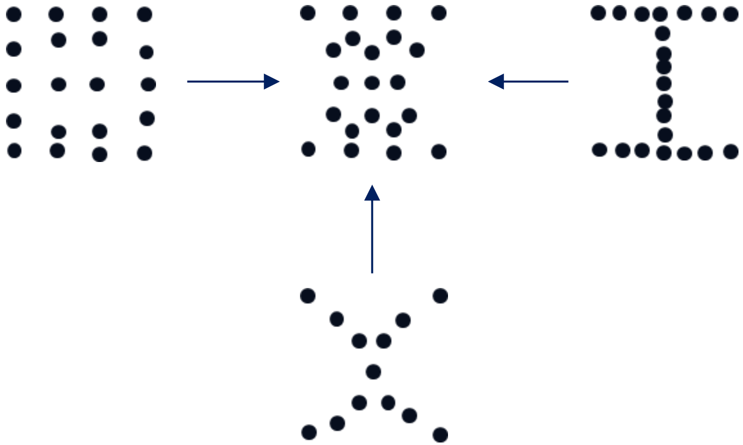


Choose $\lambda > 0$ that minimizes the error (RRMSE) between

- the **train** output fields and
- the **train** reconstructed fields

How to choose a reference measure ?

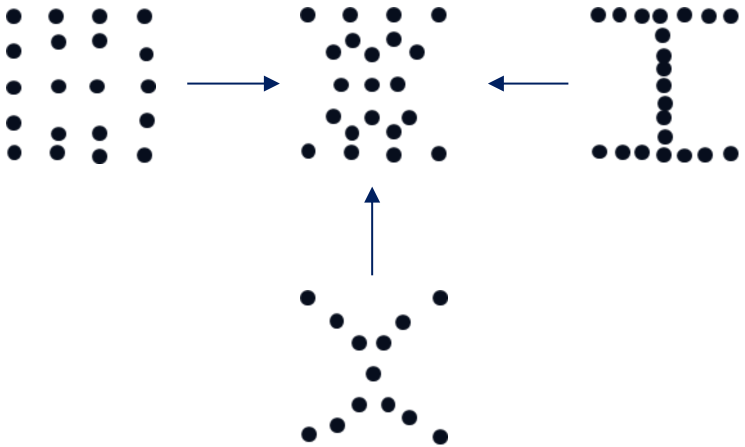
1) Optimal transport barycenter:



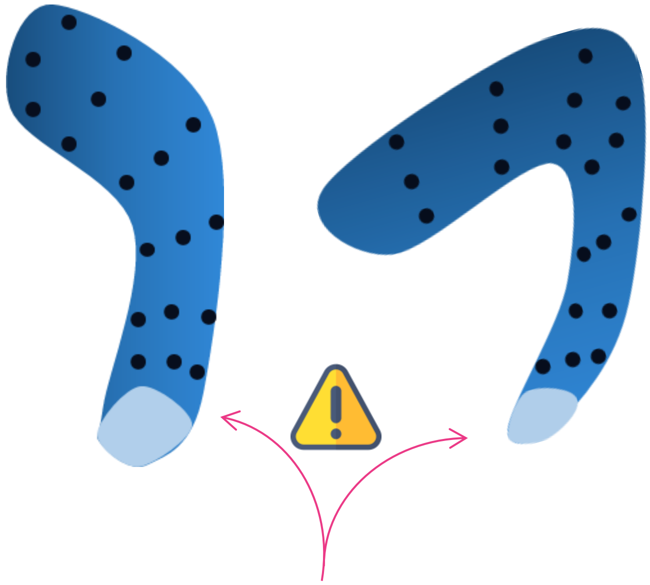
Barycenter of all train measures

How to choose a reference measure ?

1) Optimal transport barycenter:



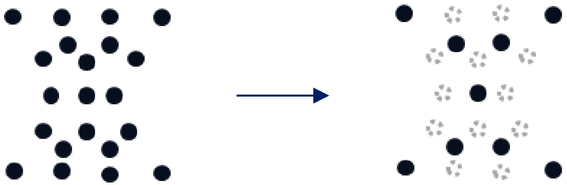
Barycenter of all train measures



Discretizations of manifolds

How to choose a reference measure ?

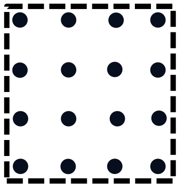
2) Subsample from a train measure:



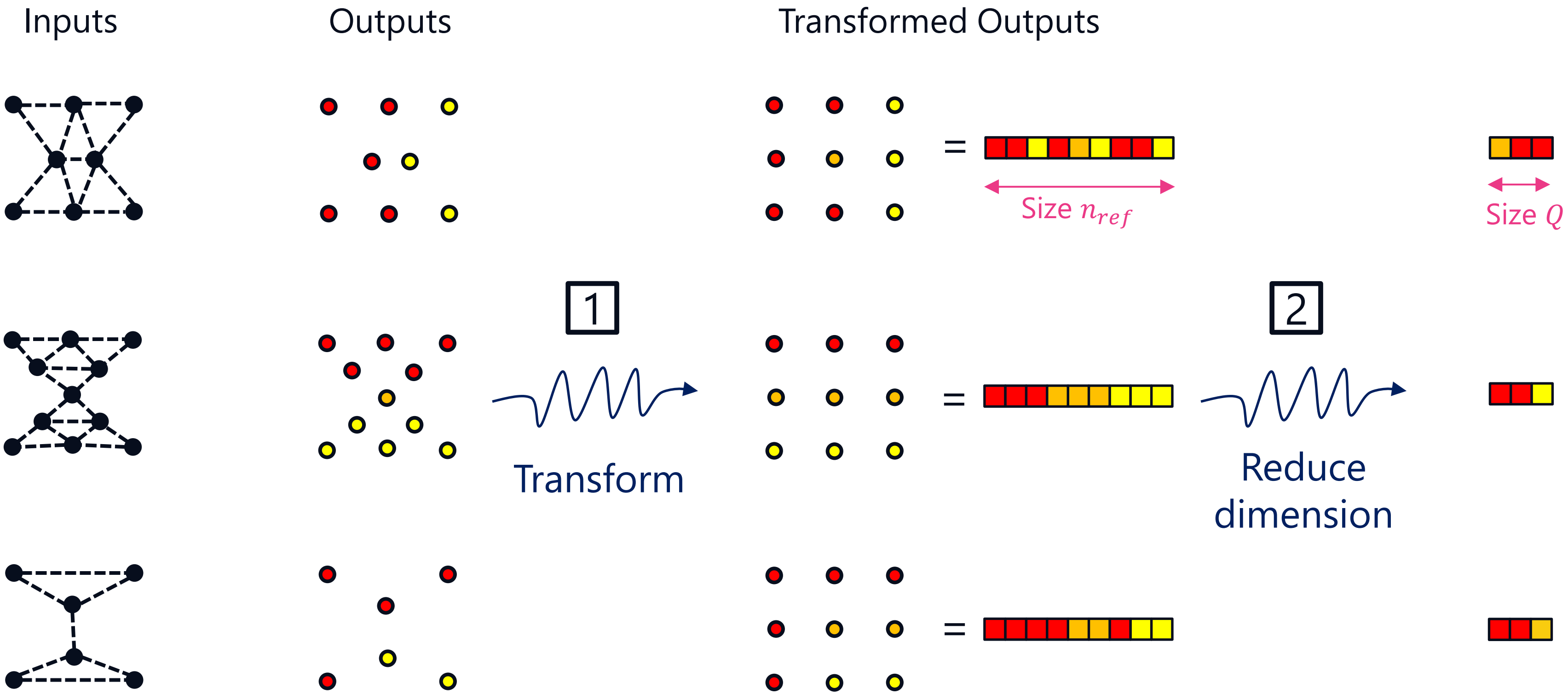
One train measure

subsampled

3) Uniform grid on a reference shape:



Express signals/fields in the same space?



Dimension reduction (in practice)

Principal component analysis

[Kontolati 2022]

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

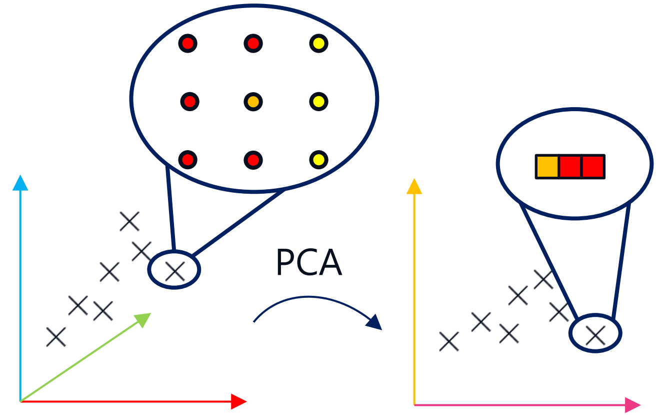
$\overline{\boldsymbol{T}} = \boldsymbol{T}$ centered

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

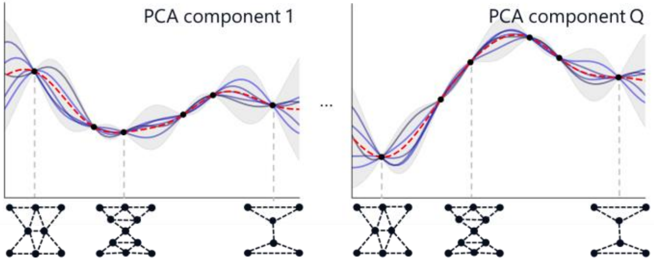
$\lambda_1 \leq \cdots \leq \lambda_Q$: eigenvalues

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

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



Learn Q independent GPs using SWWL graph kernels for the inputs

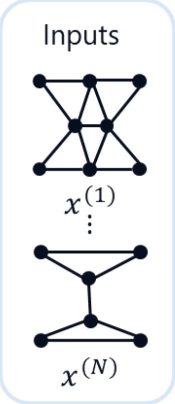


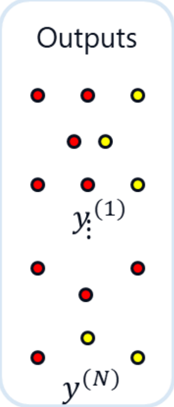
TOS-GP: Transported Output Signal Gaussian Processes

[CP, Da Veiga, Garnier, Staber, 2025]

Train →

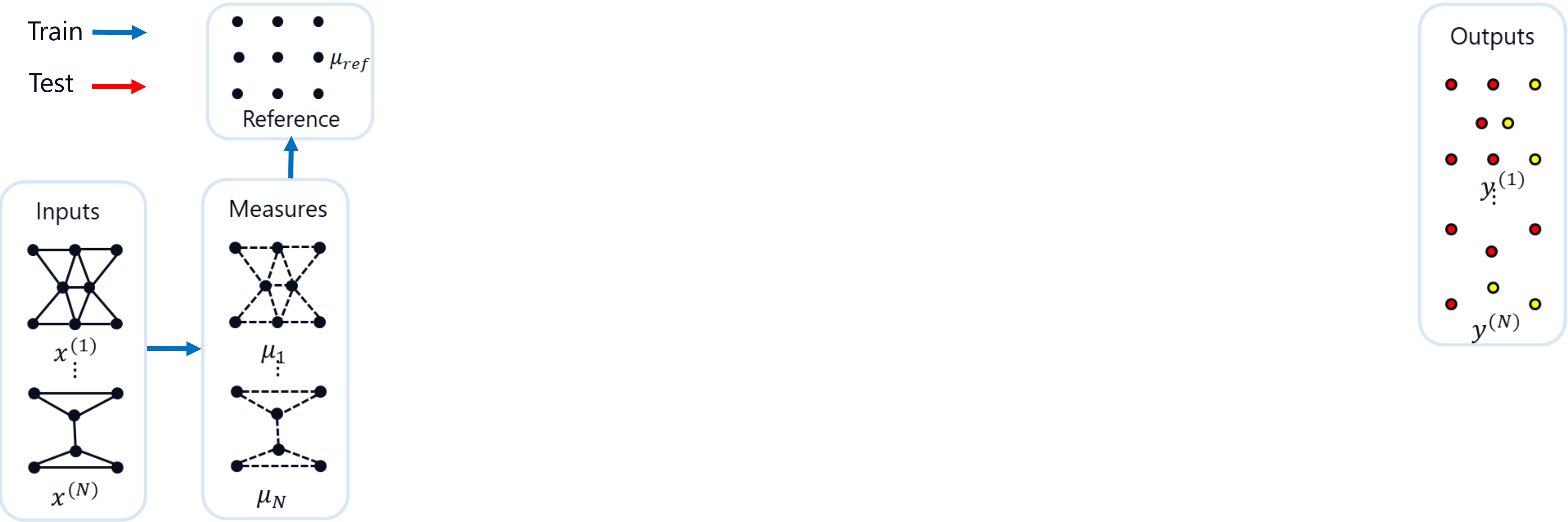
Test →





TOS-GP: Transported Output Signal Gaussian Processes

[CP, Da Veiga, Garnier, Staber, 2025]



Train

Test

μ_{ref}

Reference

Inputs

$x^{(1)}$

$x^{(N)}$

Measures

μ_1

μ_N

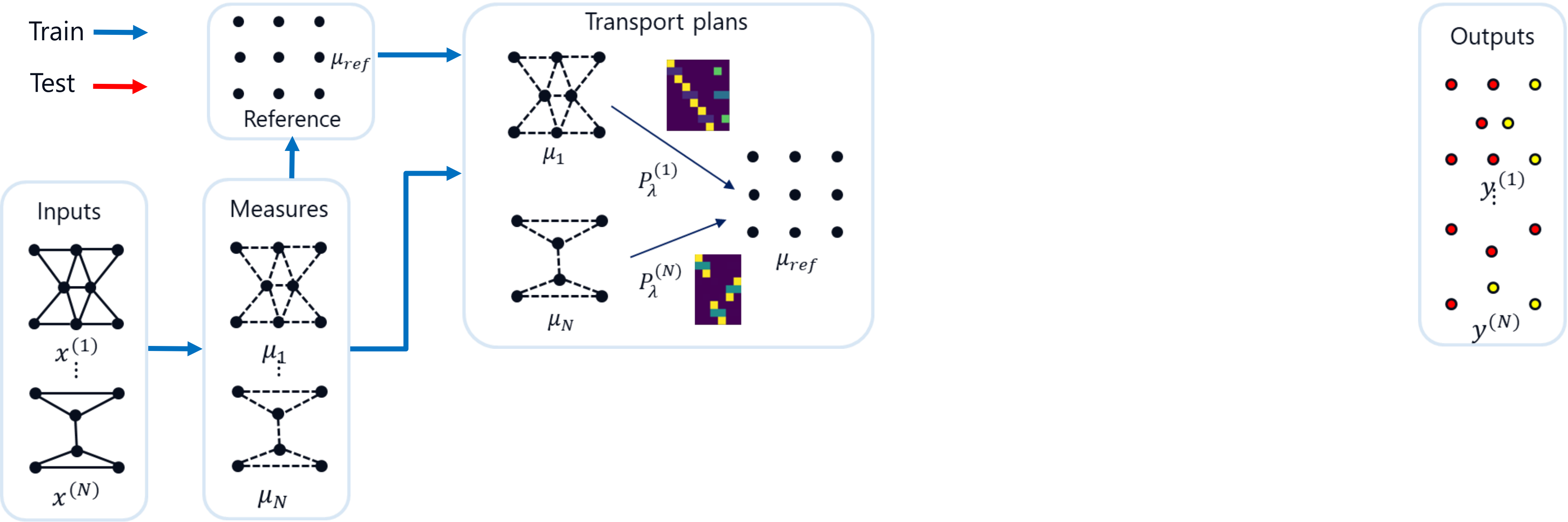
Outputs

$y^{(1)}$

$y^{(N)}$

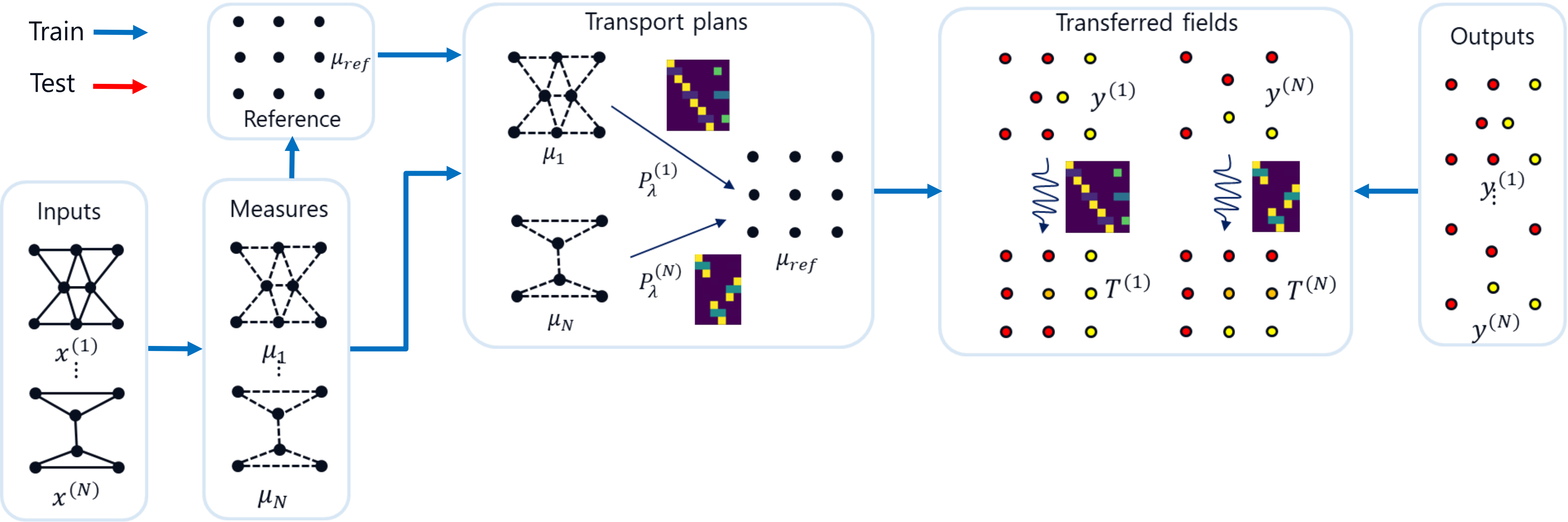
TOS-GP: Transported Output Signal Gaussian Processes

[CP, Da Veiga, Garnier, Staber, 2025]



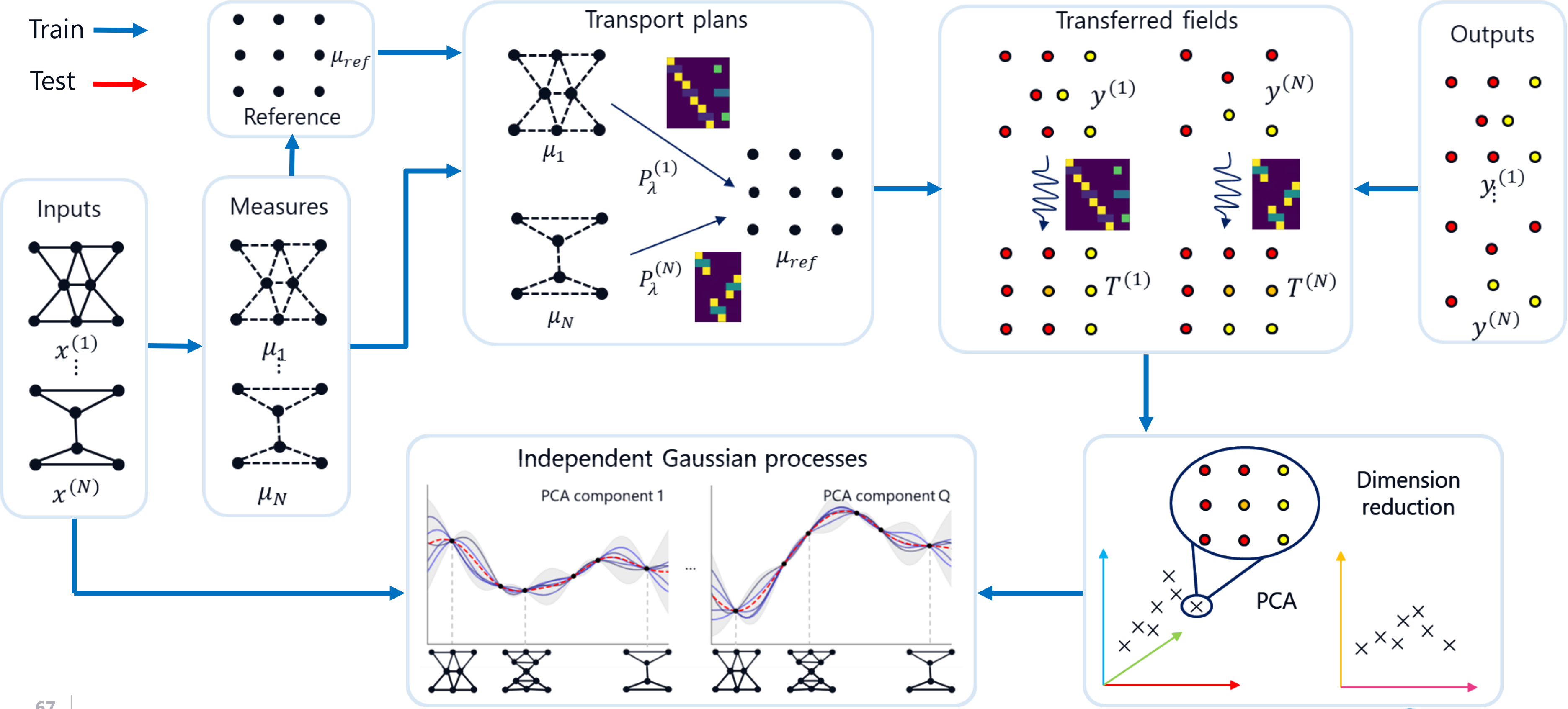
TOS-GP: Transported Output Signal Gaussian Processes

[CP, Da Veiga, Garnier, Staber, 2025]



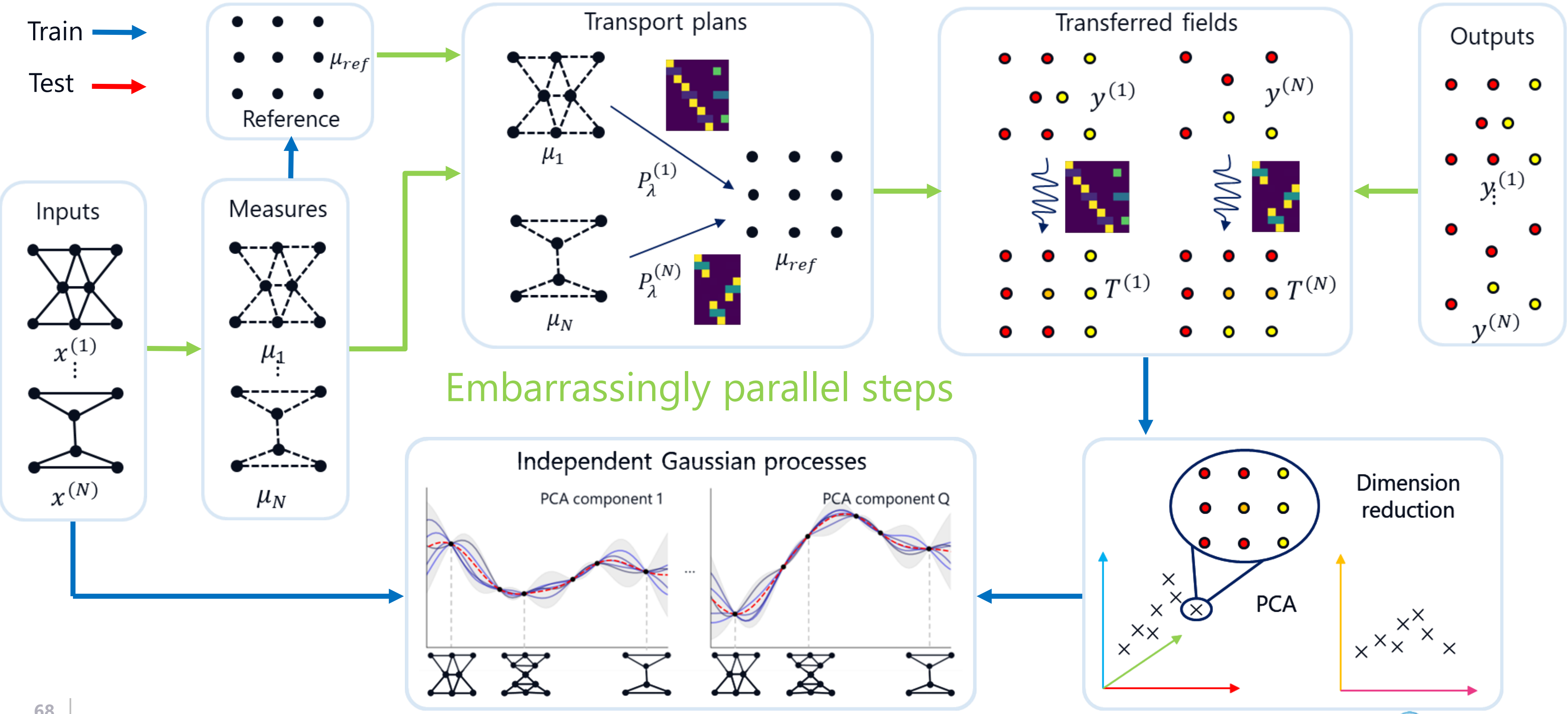
TOS-GP: Transported Output Signal Gaussian Processes

[CP, Da Veiga, Garnier, Staber, 2025]



TOS-GP: Transported Output Signal Gaussian Processes

[CP, Da Veiga, Garnier, Staber, 2025]



Train

Test

Reference

Inputs

Measures

Transport plans

Transferred fields

Outputs

Embarrassingly parallel steps

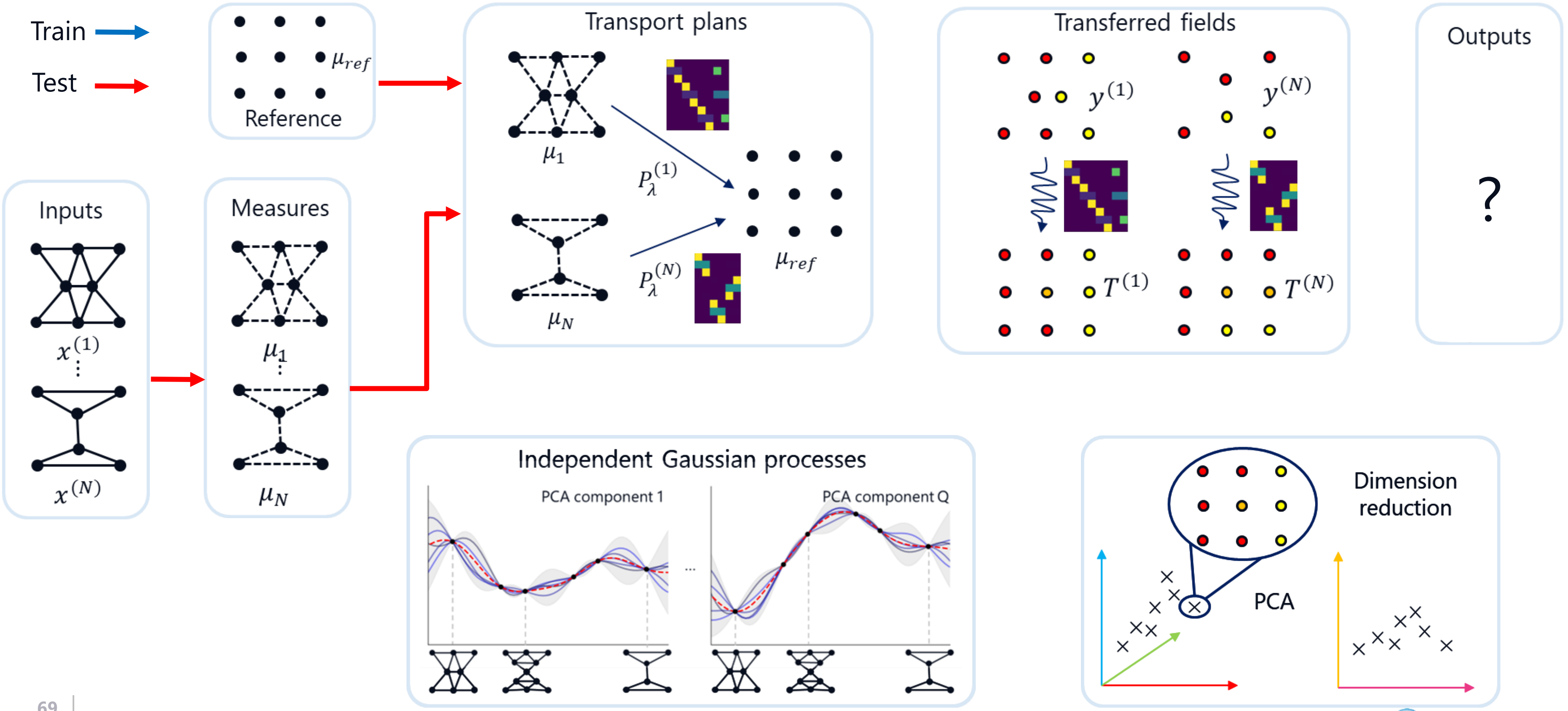
Independent Gaussian processes

Dimension reduction

PCA

TOS-GP: Transported Output Signal Gaussian Processes

[CP, Da Veiga, Garnier, Staber, 2025]



Train

Test

Reference

μ_{ref}

Transport plans

μ_1

μ_N

$P_\lambda^{(1)}$

$P_\lambda^{(N)}$

μ_{ref}

Transferred fields

$y^{(1)}$

$y^{(N)}$

$T^{(1)}$

$T^{(N)}$

Outputs

?

Inputs

$x^{(1)}$

$x^{(N)}$

Measures

μ_1

μ_N

Independent Gaussian processes

PCA component 1

...

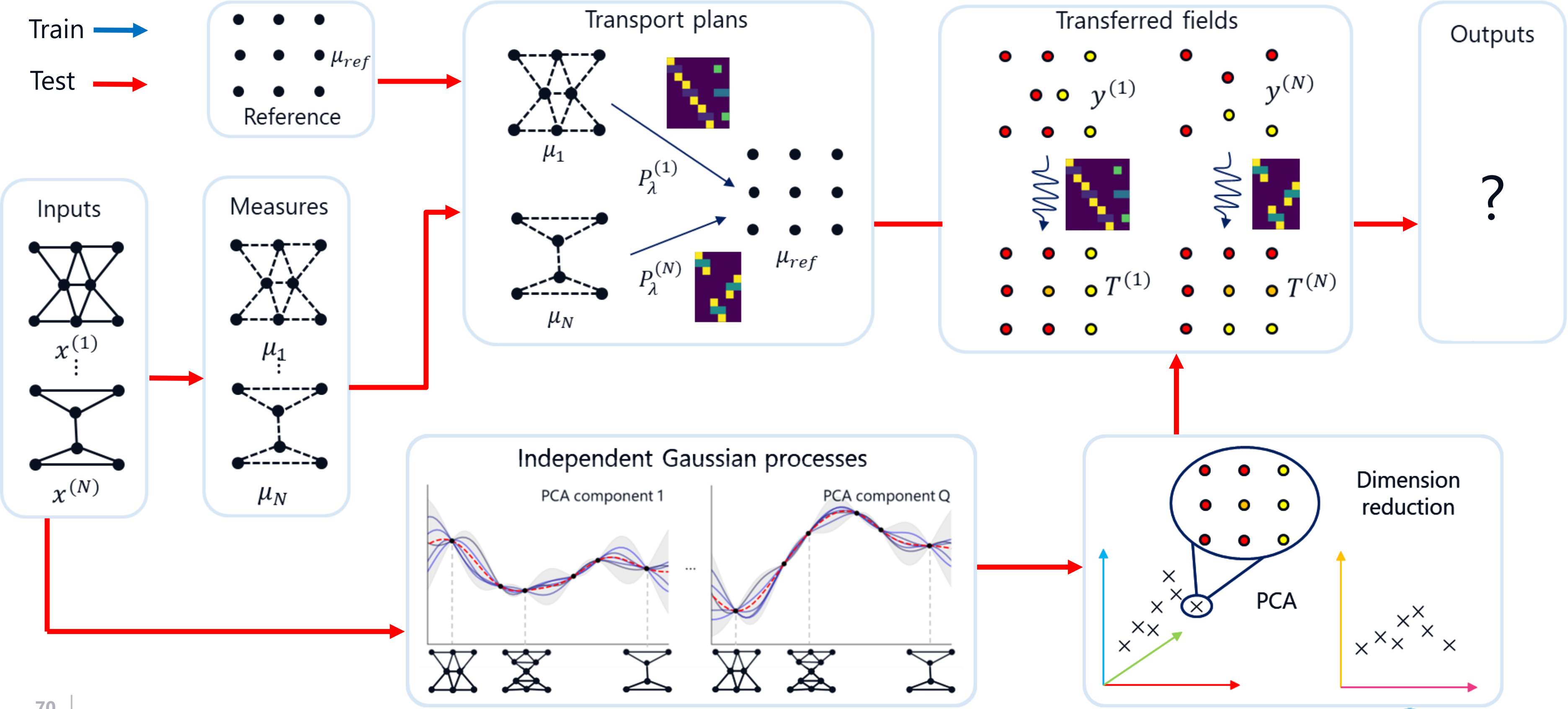
PCA component Q

Dimension reduction

PCA

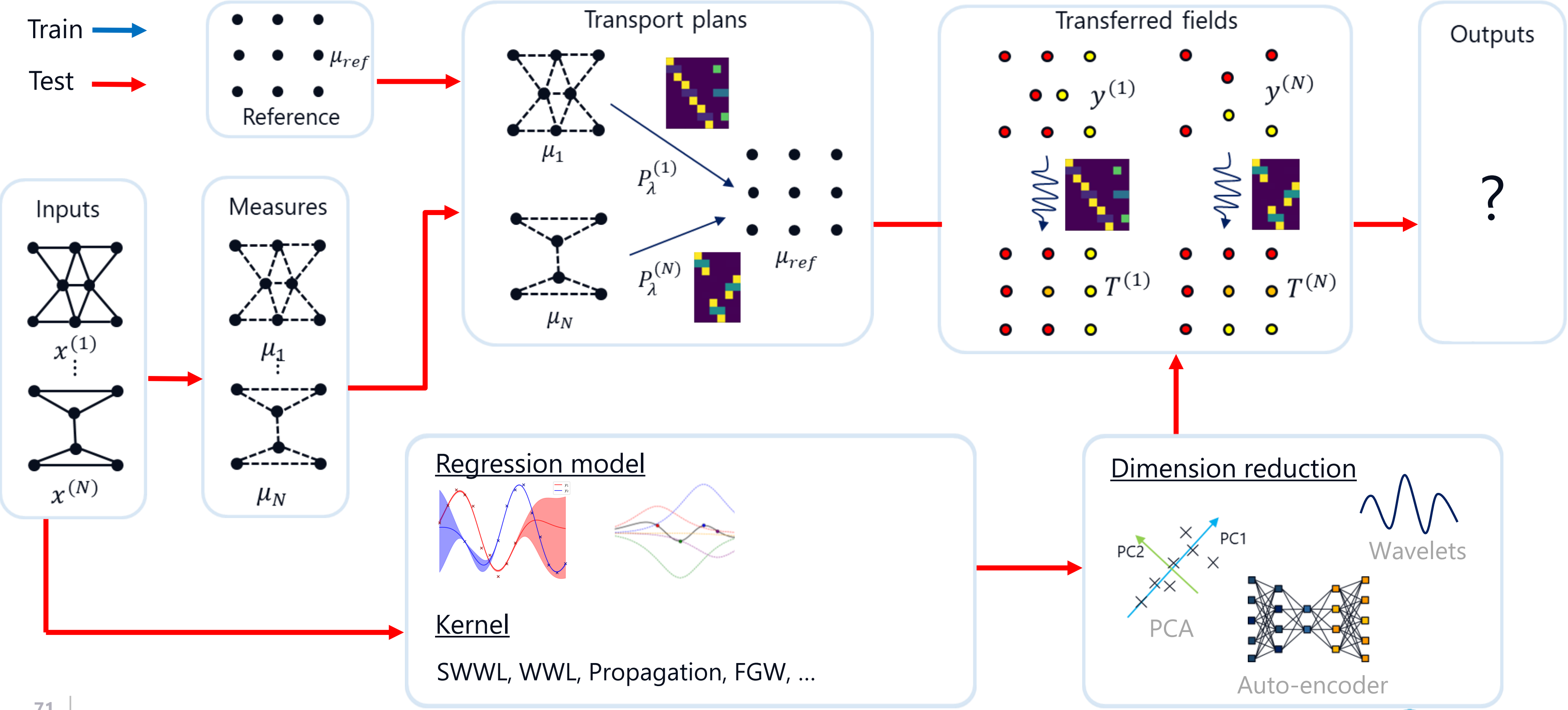
TOS-GP: Transported Output Signal Gaussian Processes

[CP, Da Veiga, Garnier, Staber, 2025]



TOS-GP: Transported Output Signal Gaussian Processes

[CP, Da Veiga, Garnier, Staber, 2025]



Train

Test

Reference

μ_{ref}

Transport plans

μ_1

μ_N

$P_\lambda^{(1)}$

$P_\lambda^{(N)}$

μ_{ref}

Transferred fields

$y^{(1)}$

$y^{(N)}$

$T^{(1)}$

$T^{(N)}$

Outputs

?

Inputs

$x^{(1)}$

$x^{(N)}$

Measures

μ_1

μ_N

Regression model

Kernel

SWWL, WWL, Propagation, FGW, ...

Dimension reduction

PC1

PC2

PCA

Wavelets

Auto-encoder

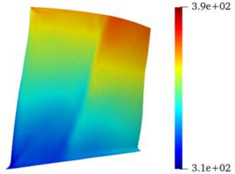
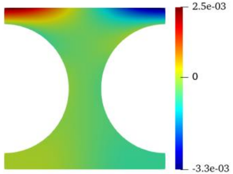
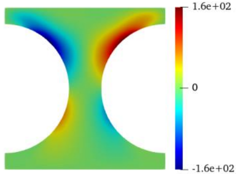
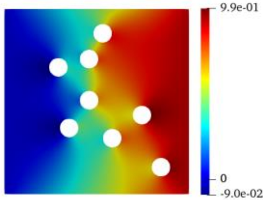
I) Scalar outputs

1- Gaussian process regression
2- Graph kernels
3- SWWL graph kernel

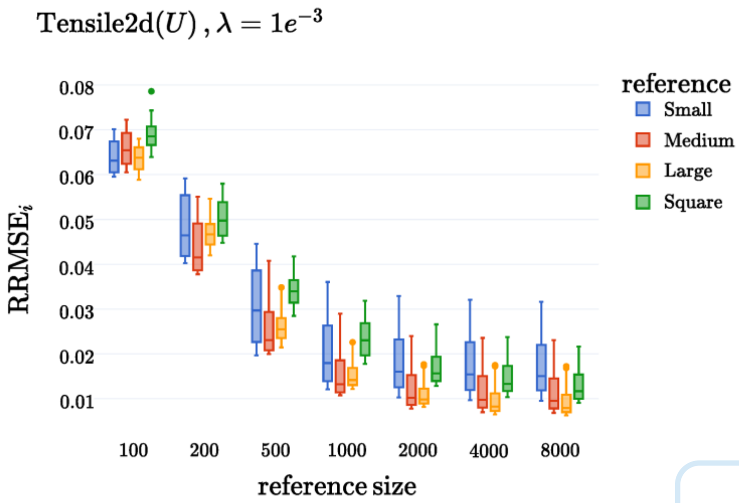
II) Signal outputs

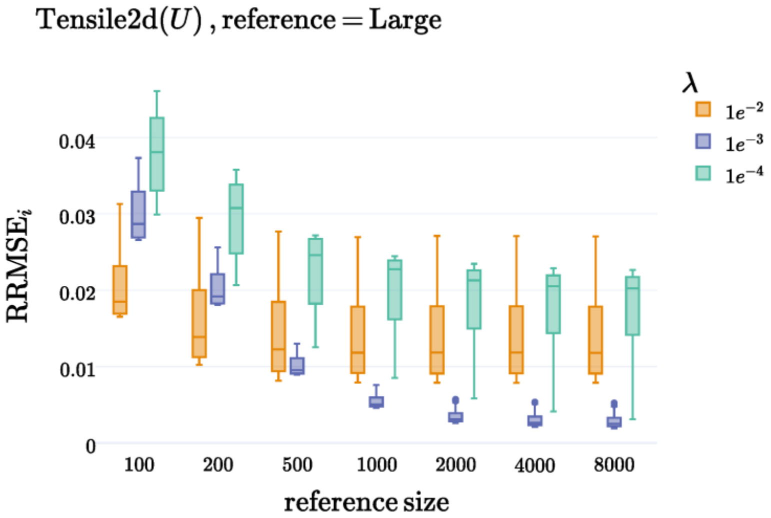
1- Problem statement
2- Related approaches
3- TOS-GP
4- Experiments

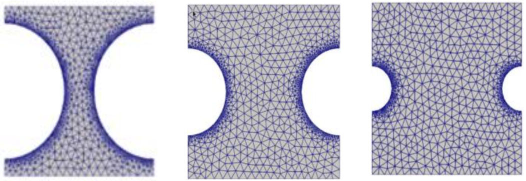
Datasets

Dataset name	Train/Test	Nodes	Output fields
Rotor37	1000 / 200	~30000	 Temperature (T)
Tensile2d	500 / 200	~9500	 H displacement (U)  Shear stress (σ12)
Multiscale	764 / 376	~4600	 H displacement (U)

TOS-GP: regression scores



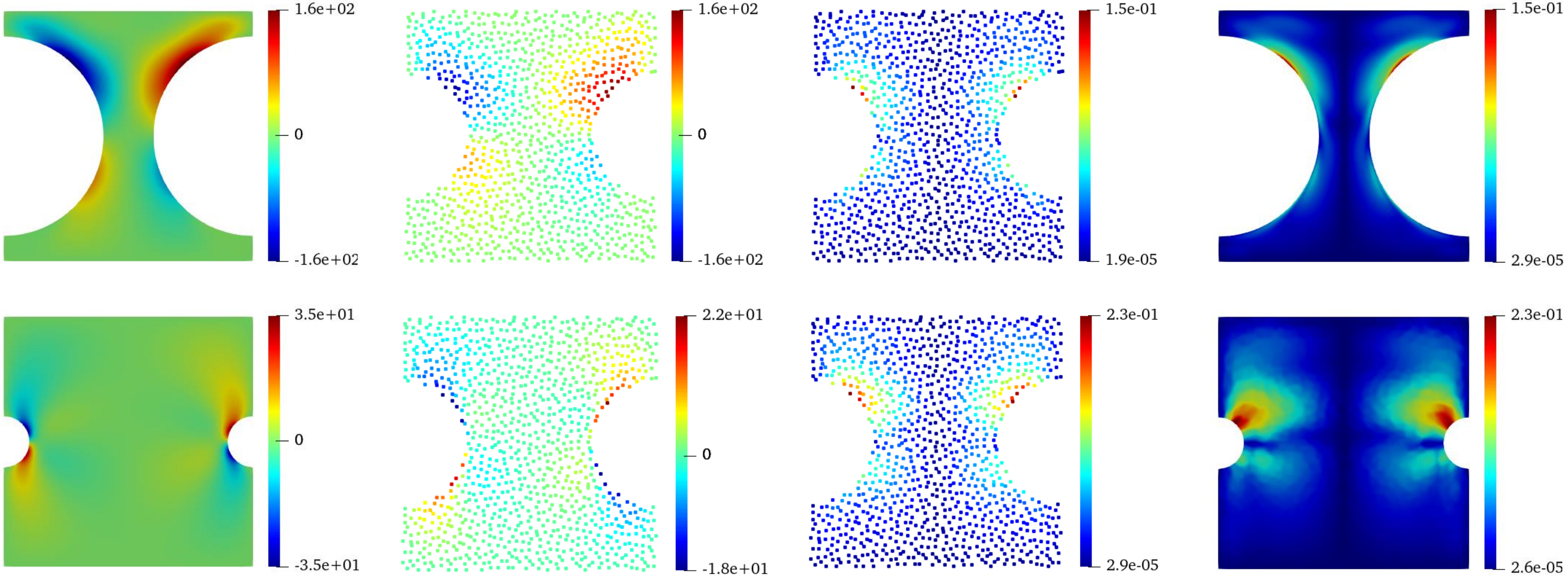




Small Medium Large

- The error decreases when the size of the reference increases
- It remains close to a constant beyond 1000 points
- The choice of the reference type has little importance for this problem
- The choice of the **regularization parameter** is **critical**

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



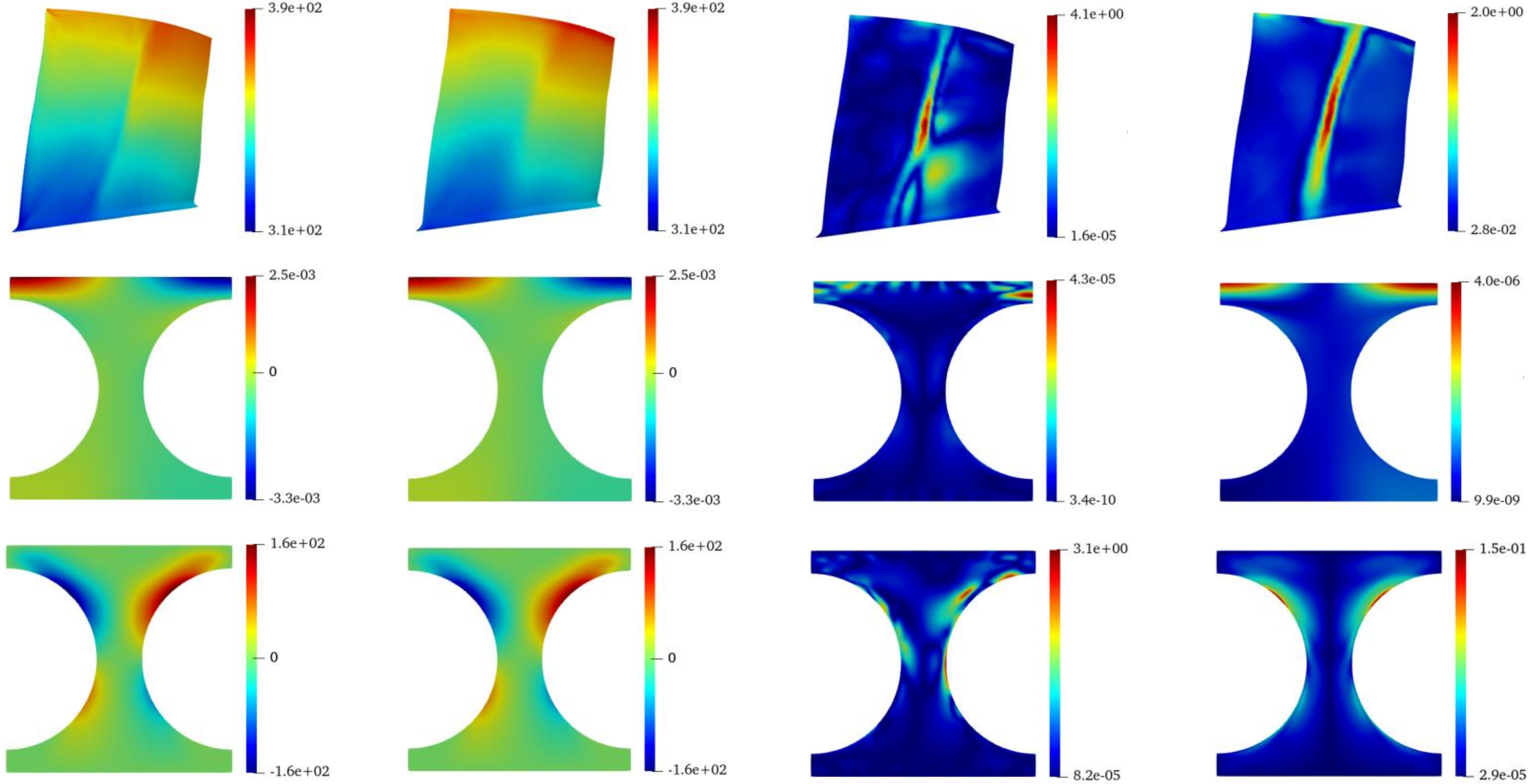
Ground truth

Prediction (transferred space)

Posterior std (transferred space)

Posterior std

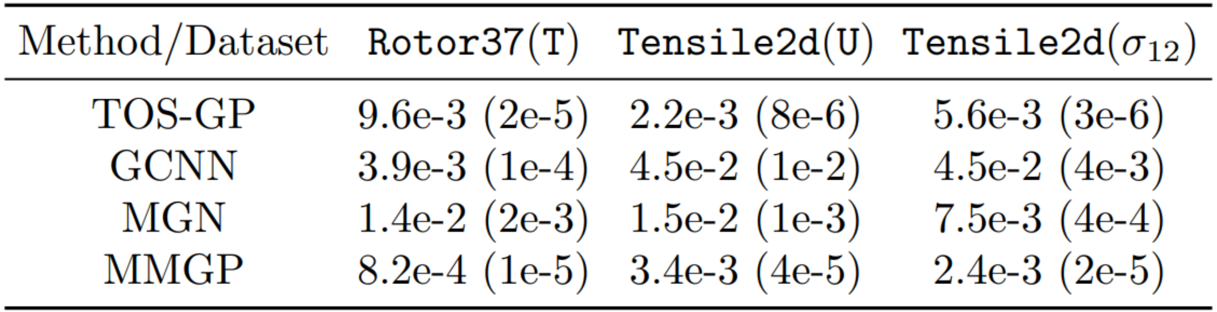
TOS-GP: predictions and uncertainties



Ground truth Prediction Absolute error Posterior std

TOS-GP: regression scores

RRMSE
(10 exp)

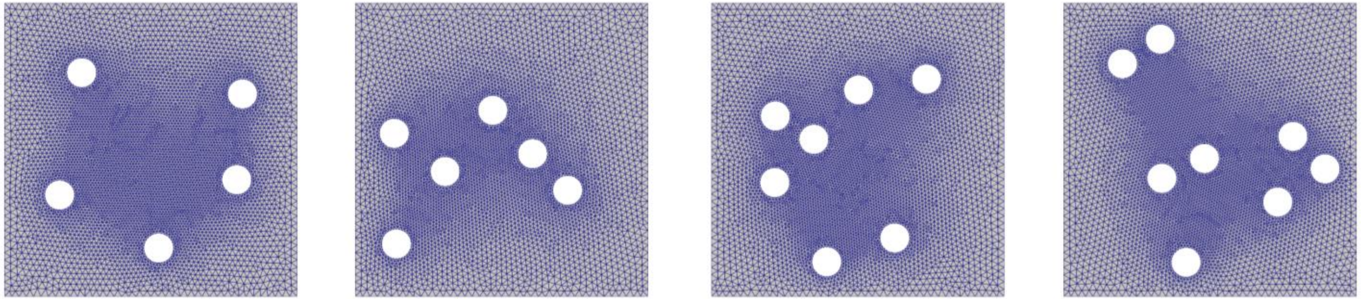


Method/Dataset	Rotor37(T)	Tensile2d(U)	Tensile2d(σ_{12})
TOS-GP	9.6e-3 (2e-5)	2.2e-3 (8e-6)	5.6e-3 (3e-6)
GCNN	3.9e-3 (1e-4)	4.5e-2 (1e-2)	4.5e-2 (4e-3)
MGN	1.4e-2 (2e-3)	1.5e-2 (1e-3)	7.5e-3 (4e-4)
MMGP	8.2e-4 (1e-5)	3.4e-3 (4e-5)	2.4e-3 (2e-5)

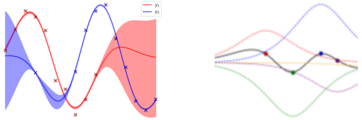
$$RRMSE^2\left(\{y^{(i)}\}_{i=1,\cdots,N_*},\{\hat{y}^{(i)}\}_{i=1,\cdots,N_*}\right)=\frac{1}{N_*}\sum_{i=1}^{N_*}RRMSE_i^2\left(y^{(i)},\hat{y}^{(i)}\right)$$

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

Varying topologies: 2D_multiscale_hyperelasticity



Regression model

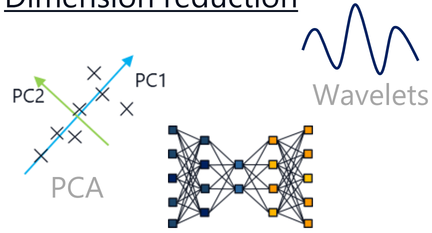


Kernel

SWWL, WWL, Propagation, FGW, …

Squared-exp using the MMD distance between the centers of the pores

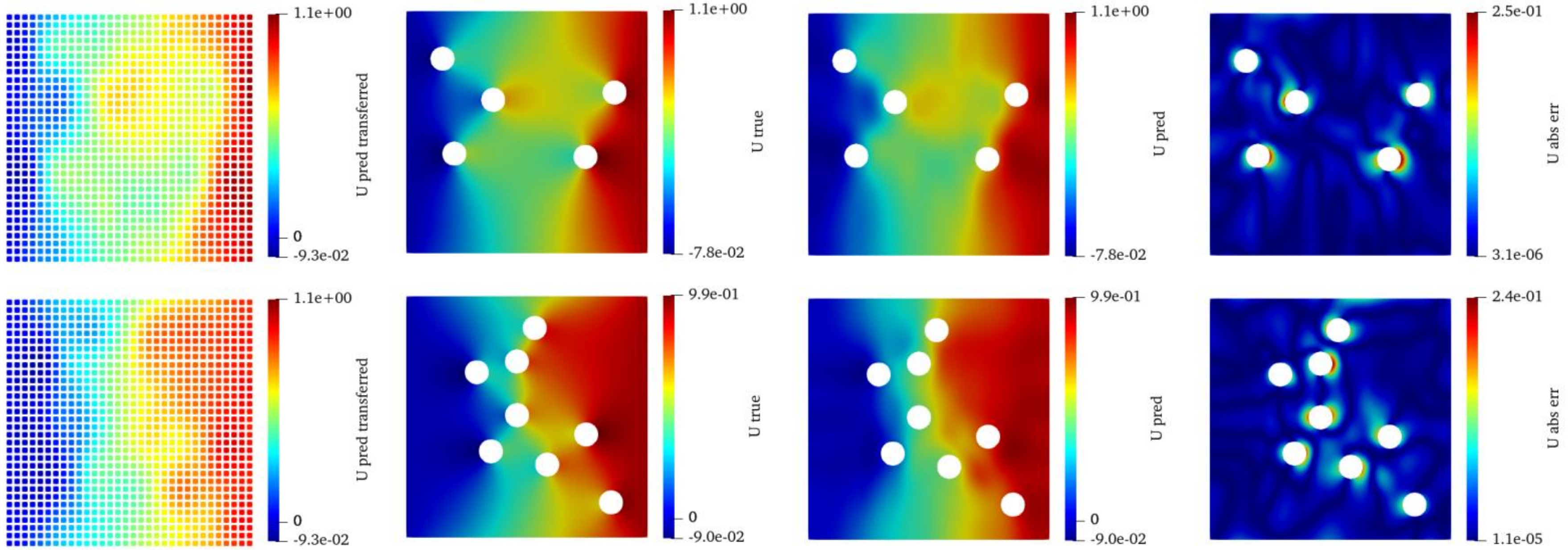
Dimension reduction



Auto-encoder

Auto-encoder with convolutional and 2D discrete Fourier layers [Li et al. 2020]

Varying topologies: 2D_multiscale_hyperelasticity



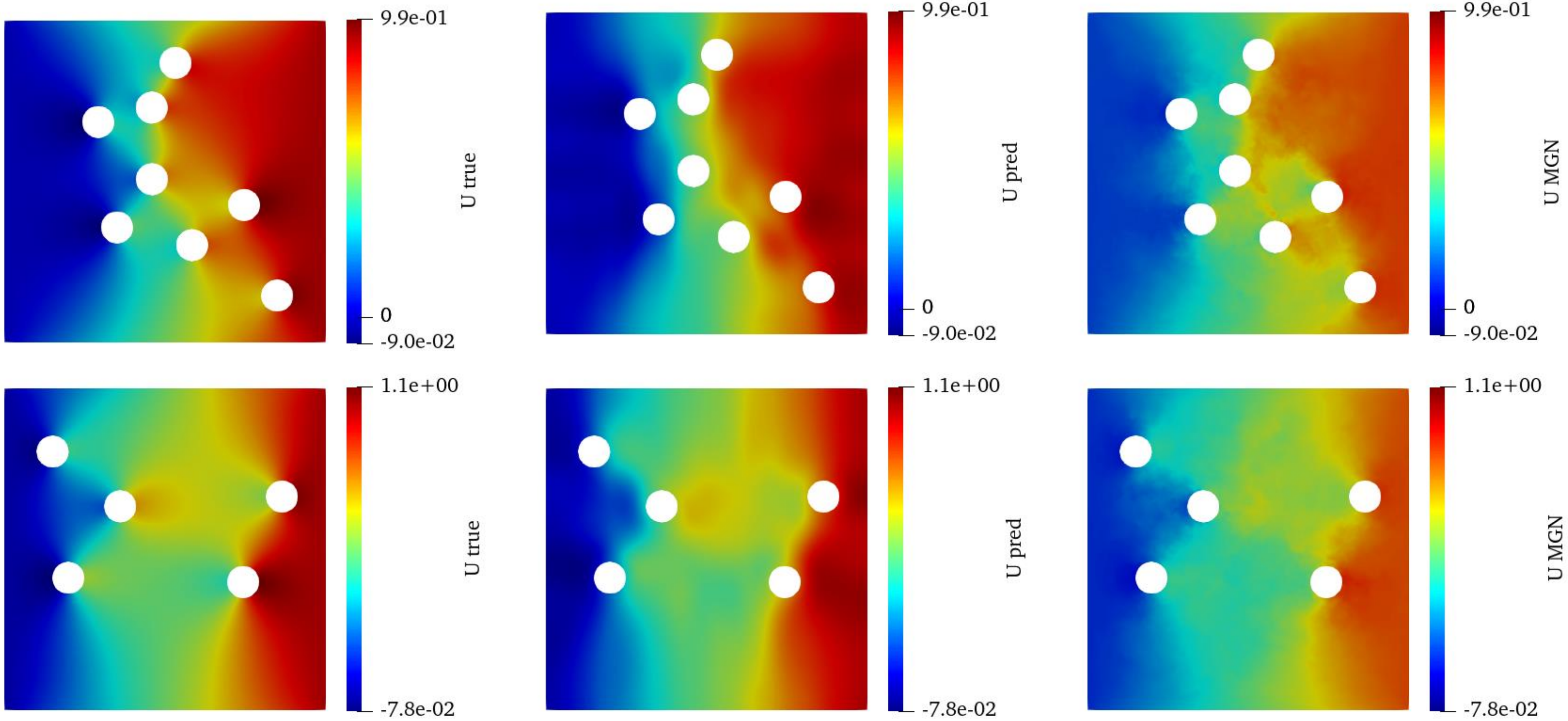
Prediction (transferred space)

Ground truth

Prediction

Absolute error

Varying topologies: 2D_multiscale_hyperelasticity

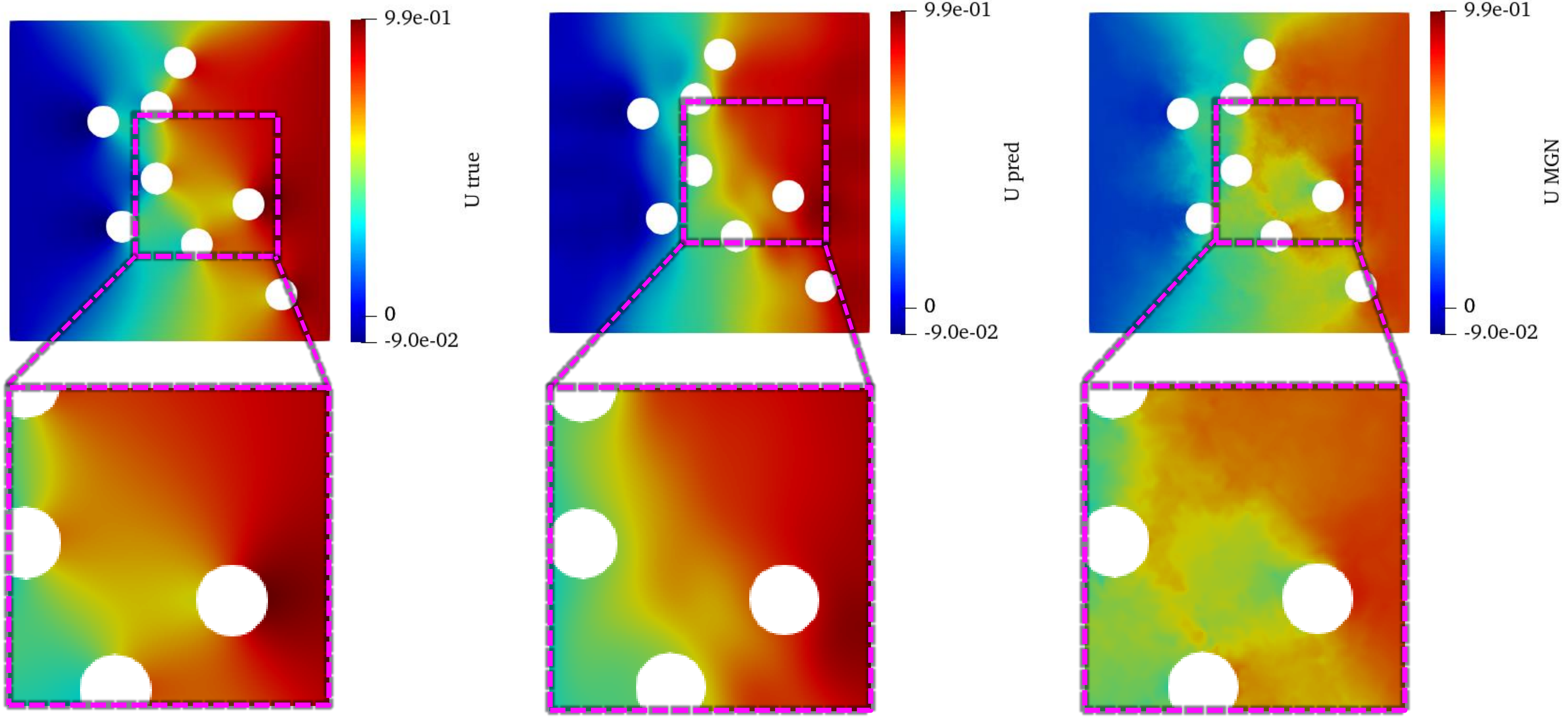


Ground truth

Prediction (TOS-GP)

Prediction (MGN)

Varying topologies: 2D_multiscale_hyperelasticity



Ground truth

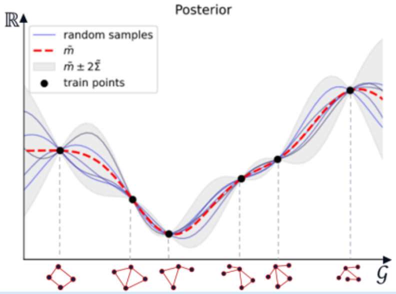
Prediction
(TOS-GP)

Prediction
(MGN)

Conclusion

Inputs = Graphs, Outputs = **Scalars**

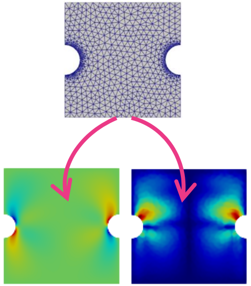
- SWWL graph kernel
  ✓ Positive definite
  ✓ Can consider very large graphs



Inputs = Graphs, Outputs = **Signals**

- Classical techniques impossible to use directly
  MOGP, OVGP, GSP, dimension reduction, …

- TOS-GP: Transported Output Signal GP
  Optimal transport + Dimension reduction
  ✓ Flexible (change kernel/dimension reduction)
  ✓ No assumption on the data (mesh/topology)
  ✓ Few hyperparameters: λ, ref. measure, WL iter.



- Future work
  Consider more discontinuous signals
  Optimal transport variants

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). Distribution Regression with Sliced Wasserstein Kernels.
  - Bonneel, N., Rabin, J., Peyr´e, G., and Pfister, H. (2015). Sliced and radon Wasserstein barycenters of measures

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.

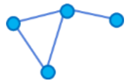
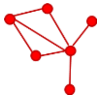
Gaussian process regression

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

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

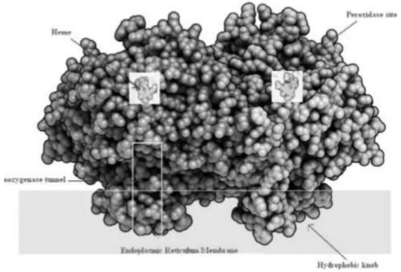
Posterior distribution:

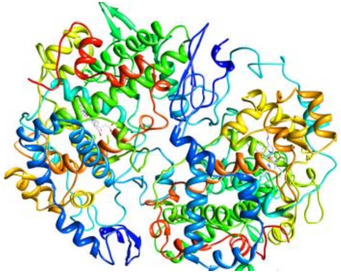
$$\boldsymbol{f}_* \mid \boldsymbol{G}, \boldsymbol{y}, \boldsymbol{G}^* \sim \mathcal{N}(\overline{\boldsymbol{m}}, \overline{\boldsymbol{\Sigma}})$$

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

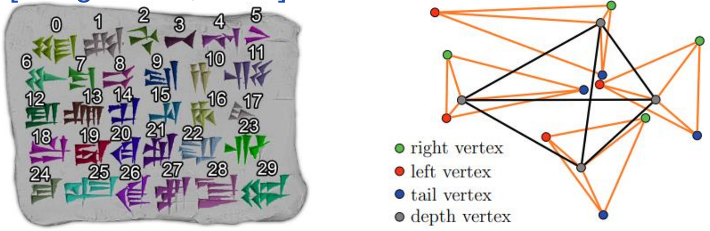
uncertainties $\overline{\boldsymbol{\Sigma}} = \boldsymbol{K}_{**} - \boldsymbol{K}_*(\boldsymbol{K} + \sigma^2 I)^{-1}\boldsymbol{K}_*^T$

SWWL kernel: experiments



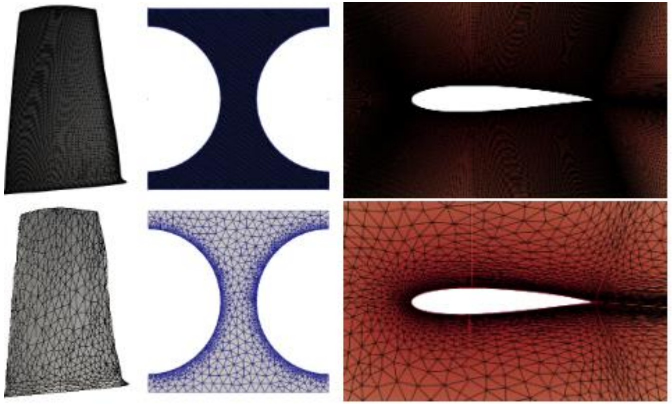


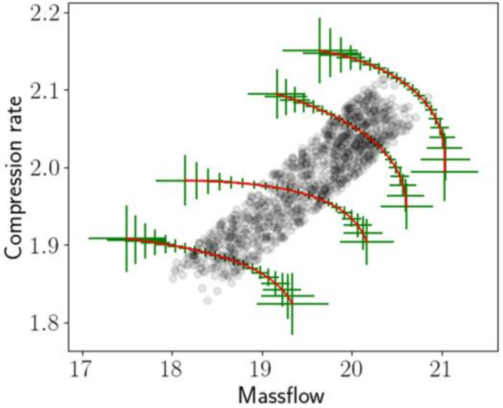
[Kriege et al., 2019]

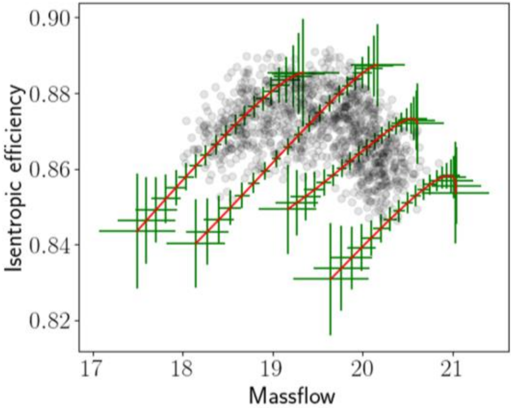


(a) Cuneiform tablet

(b) Graph representation







MMD subsampling procedure

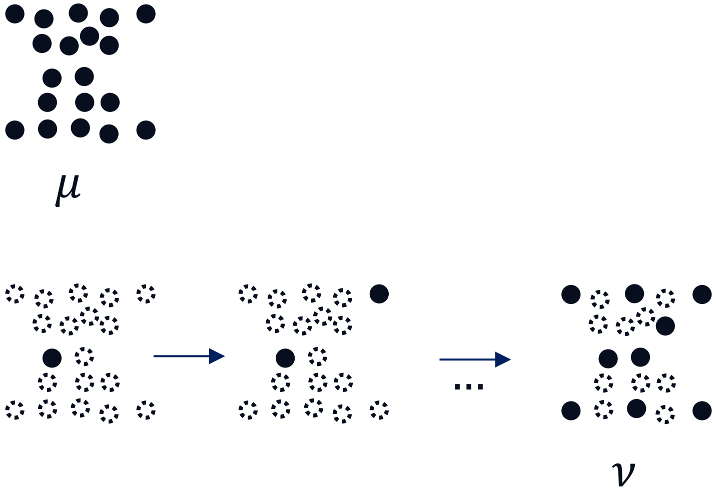
Maximum mean discrepancy:

$$MMD_k(\mu,\nu) = \mathbb{E}_{x\sim\mu,x'\sim\mu}[k(x,x')] + \mathbb{E}_{y\sim\nu,y'\sim\nu}[k(y,y')] - 2\mathbb{E}_{x\sim\mu,y\sim\nu}[k(x,y)]$$

<u>Input:</u> μ a given measure in the train set.
<u>Output:</u> ν the subsampled measure.

$\nu = \emptyset$

At each iteration, choose the point x in the support of μ that minimizes the MMD between μ and $\nu + \delta_x$, and update ν.



POWERED BY TRUST