Deep kernel learning for geostatistics

Thomas Romary, Nicolas Desassis & Solal Raymondjean

thomas.romary@minesparis.psl.eu

Centre de Géosciences, Equipe Géostatistique

JDS 2025

GEOLEARNING
CHAIRE /// Data Science for the Environment

Geostatistics in a nutshell

Main objectives

- model a natural variable of interest, seen as a *regionalized variable* $z(x),\ x \in \mathcal{X} \subset \mathbb{R}^d$ over space(-time)
- make predictions at unobserved locations
- quantify uncertainty

Hypothesis

z is a realization of a random field Z

Geostatistics in a nutshell

Main objectives

- model a natural variable of interest, seen as a *regionalized variable* $z(x),\ x \in \mathcal{X} \subset \mathbb{R}^d$ over space(-time)
- make predictions at unobserved locations
- quantify uncertainty

Hypothesis

z is a realization of a random field Z

Gaussian Processes $Z(x),\ x \in \mathcal{X} \subset \mathbb{R}^d$

$Z = (Z(x_1), \ldots, Z(x_n))$ is a Gaussian vector
$Z \sim \mathcal{N}(\mu, \Sigma_\theta)$, with

- $\mu = \mathbb{E}(Z)$
- $(\Sigma_\theta)_{i,j} = \mathrm{Cov}(Z(x_i), Z(x_j)) = \mathbf{C}_\theta(|x_i - x_j|)$

Maximum-likelihood estimation

$(\widehat{\mu}, \widehat{\theta}) = \mathrm{argmin}_{(\mu,\theta)} \log(\det\Sigma_\theta) + (Z-\mu)^t \Sigma_\theta^{-1} (Z-\mu)$

Conditioning (prediction)

$Z(x_T)|Z(x_T) \sim \mathcal{N}(Z_T^\star, \Sigma_T^\star)$, with $T \cap D = \varnothing$ such that

- $Z^\star(x_T) = \mu_T + \Sigma_{TD}\Sigma_{DD}^{-1}(Z(x_D) - \mu_D)$
- $\Sigma_T^\star = \Sigma_{TT} - \Sigma_{TD}\Sigma_{DD}^{-1}\Sigma_{DT}$

Gaussian Processes $Z(x),\ x \in \mathcal{X} \subset \mathbb{R}^d$

$Z = (Z(x_1), \ldots, Z(x_n))$ is a Gaussian vector
$Z \sim \mathcal{N}(\mu, \Sigma_\theta)$, with

- $\mu = \mathbb{E}(Z)$
- $(\Sigma_\theta)_{i,j} = \mathrm{Cov}(Z(x_i), Z(x_j)) = \mathbf{C}_\theta(|x_i - x_j|)$

Maximum-likelihood estimation

$(\widehat{\mu}, \widehat{\theta}) = \mathrm{argmin}_{(\mu,\theta)} \log(\det\Sigma_\theta) + (Z-\mu)^t \Sigma_\theta^{-1}(Z-\mu)$

Conditioning (prediction)

$Z(x_T)|Z(x_T) \sim \mathcal{N}(Z_T^\star, \Sigma_T^\star)$, with $T \cap D = \varnothing$ such that

- $Z^\star(x_T) = \mu_T + \Sigma_{TD}\Sigma_{DD}^{-1}(Z(x_D) - \mu_D)$
- $\Sigma_T^\star = \Sigma_{TT} - \Sigma_{TD}\Sigma_{DD}^{-1}\Sigma_{DT}$

Gaussian Processes $Z(x),\ x \in \mathcal{X} \subset \mathbb{R}^d$

$Z = (Z(x_1), \ldots, Z(x_n))$ is a Gaussian vector
$Z \sim \mathcal{N}(\mu, \Sigma_\theta)$, with

- $\mu = \mathbb{E}(Z)$
- $(\Sigma_\theta)_{i,j} = \mathrm{Cov}(Z(x_i), Z(x_j)) = \mathbf{C}_\theta(|x_i - x_j|)$

Maximum-likelihood estimation

$(\widehat{\mu}, \widehat{\theta}) = \mathrm{argmin}_{(\mu,\theta)} \log(\det\Sigma_\theta) + (Z-\mu)^t \Sigma_\theta^{-1}(Z-\mu)$

Conditioning (prediction)

$Z(x_T)|Z(x_T) \sim \mathcal{N}(Z_T^\star, \Sigma_T^\star)$, with $T \cap D = \varnothing$ such that

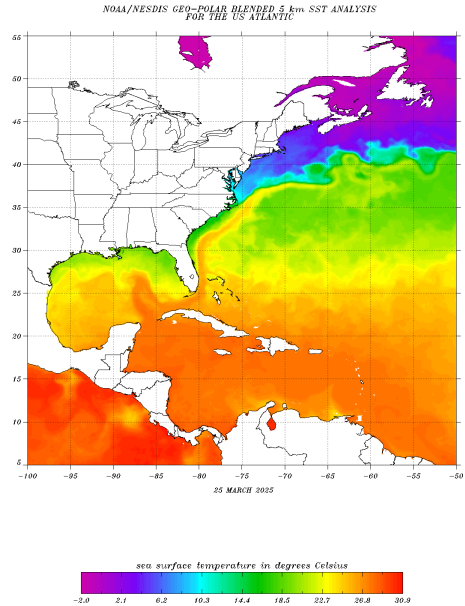
- $Z^\star(x_T) = \mu_T + \Sigma_{TD}\Sigma_{DD}^{-1}(Z(x_D) - \mu_D)$
- $\Sigma_T^\star = \Sigma_{TT} - \Sigma_{TD}\Sigma_{DD}^{-1}\Sigma_{DT}$

Limitations

- GPs generally assume a stationary covariance function, which may not be appropriate for all spatial data : $\text{Cov}(Z(x_i), Z(x_j)) = \mathbf{C}_\theta(|x_i - x_j|)$

  Matérn covariance
  $\mathbf{C}(x_i, x_j) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\|x_i - x_j\|}{\ell} \right)^\nu K_\nu \left(\frac{\|x_i - x_j\|}{\ell} \right)$

- GPs can be computationally expensive for large datasets



Non stationary covariance constructions

Convolution models

$$\mathsf{Cov}(Z(x), Z(y)) = \int_{\mathcal{T}} \int_{\mathbb{R}^d} f_x(u,t) f_y(u,t) f_T(t) du dt$$

$$C(x,y) = |\Sigma_x|^{1/4} |\Sigma_y|^{1/4} \left| \frac{\Sigma_x + \Sigma_y}{2} \right|^{-1/2} \frac{2^{1-\nu(x,y)} Q_{xy}(x-y)^{\nu(x,y)}}{\sqrt{\Gamma(\nu(x))\Gamma(\nu(y))}} K_{\nu(x,y)} \left(\sqrt{Q_{xy}(x-y)} \right)$$

Varying parameters in SPDE

$$(\kappa_x^2 - \nabla H_x \nabla)^{\alpha/2} Z(x) = W(x)$$

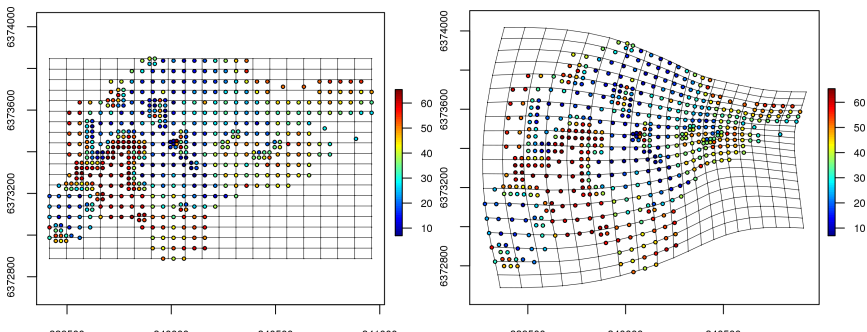
Space deformation

$$Z(x) = Z(\mathbf{f}(x)) \Rightarrow \mathsf{Cov}(Z(x), Z(y)) = \mathbf{C}(\mathbf{f}(x), \mathbf{f}(y))$$

Space Deformation

Relax the stationarity assumption

$$\mathbf{C}_\theta(\mathbf{x_i}, \mathbf{x_j}) = \mathbf{C}(|\mathbf{f}_\theta(\mathbf{x_i}) - \mathbf{f}_\theta(\mathbf{x_j})|)$$



Space deformation example: left geographical space, right deformed space

$\Rightarrow \mathbf{f}_\theta$ is a transport map

Formalization

- The sampling design $X = (x_1, \dots, x_n)$ is now considered random
- We want to learn a transport map f_θ (piecewise $\mathcal{C}^1$) such that the covariance function of $Z(x)$ is stationary and isotropic in the deformed space $\mathcal{X}_\theta = \{f_\theta(x),\ x \in \mathcal{X}\}$
- In other words, we want to learn the joint distribution of Z and X
- The likelihood writes

$$p(Z, X) = p(Z|X)p(X) = \mathcal{N}(Z; \mu, \Sigma_\theta)p_x(X)$$

given some prior p_x over $\mathcal{X}$ (e.g. uniform)

Formalization

- The sampling design $X = (x_1, \dots, x_n)$ is now considered random
- We want to learn a transport map f_θ (piecewise $\mathcal{C}^1$) such that the covariance function of $Z(x)$ is stationary and isotropic in the deformed space $\mathcal{X}_\theta = \{f_\theta(x),\ x \in \mathcal{X}\}$
- In other words, we want to learn the joint distribution of Z and X
- The likelihood writes

$$p(Z, X) = p(Z|X)p(X) = \mathcal{N}(Z; \mu, \Sigma_\theta)p_x(X)$$

given some prior p_x over $\mathcal{X}$ (e.g. uniform)

Formalization

- The sampling design $X = (x_1, \dots, x_n)$ is now considered random
- We want to learn a transport map f_θ (piecewise $\mathcal{C}^1$) such that the covariance function of $Z(x)$ is stationary and isotropic in the deformed space $\mathcal{X}_\theta = \{f_\theta(x),\ x \in \mathcal{X}\}$
- In other words, we want to learn the joint distribution of Z and X
- The likelihood writes

$$p(Z, X) = p(Z|X)p(X) = \mathcal{N}(Z; \mu, \Sigma_\theta)p_x(X)$$

given some prior p_x over $\mathcal{X}$ (e.g. uniform)

Formalization

- The sampling design $X = (x_1, \dots, x_n)$ is now considered random
- We want to learn a transport map f_θ (piecewise $\mathcal{C}^1$) such that the covariance function of $Z(x)$ is stationary and isotropic in the deformed space $\mathcal{X}_\theta = \{f_\theta(x),\ x \in \mathcal{X}\}$
- In other words, we want to learn the joint distribution of Z and X
- The likelihood writes

$$p(Z, X) = p(Z|X)p(X) = \mathcal{N}(Z; \mu, \Sigma_\theta)p_x(X)$$

given some prior p_x over $\mathcal{X}$ (e.g. uniform)

Normalizing Flows

Based on a recursive application of the change of variable formula:

$$p_\theta(u) = p_x(f_\theta^{-1}(u))|\det J_{f_\theta^{-1}}(u)|$$

- f_θ is a diffeomorphism (piecewise $\mathcal{C}^1$)
- f_θ is a NN trained by maximum likelihood estimation

Example: RealNVP

Stack affine coupling layers of the form

$$\mathbf{y}_{1:d} = \mathbf{x}_{1:d}$$
$$\mathbf{y}_{d+1:D} = \mathbf{x}_{d+1:D} \odot \exp\left(s(\mathbf{x}_{1:d})\right) + t(\mathbf{x}_{1:d}),$$

where s and t are dense feedforward NN, alternating between the variables

Normalizing Flows

Based on a recursive application of the change of variable formula:

$$p_\theta(u) = p_x(f_\theta^{-1}(u))|\det J_{f_\theta^{-1}}(u)|$$

- f_θ is a diffeomorphism (piecewise $\mathcal{C}^1$)
- f_θ is a NN trained by maximum likelihood estimation

Example: RealNVP

Stack affine coupling layers of the form

$$\begin{aligned}\mathbf{y}_{1:d} &= \mathbf{x}_{1:d} \\ \mathbf{y}_{d+1:D} &= \mathbf{x}_{d+1:D} \odot \exp\left(s(\mathbf{x}_{1:d})\right) + t(\mathbf{x}_{1:d}),\end{aligned}$$

where s and t are dense feedforward NN, alternating between the variables

Scaling to large datasets

Several methods have been proposed to scale Gaussian processes to large datasets, including:

- Covariance tapering

$$C(x_i, x_j) = C(x_i, x_j) C^{\mathrm{CS}}(|x_i - x_j|)$$

- Low rank approximations, e.g. predictive processes/inducing points

$$C(x_i, x_j) = C(x_i, x^\star) C_{x^\star}^{-1} C(x^\star, x_j) + \tau^2 \delta_{ij}$$

- SPDE methods
- **Vecchia approximation**

Scaling to large datasets

Several methods have been proposed to scale Gaussian processes to large datasets, including:

- Covariance tapering

$$C(x_i, x_j) = C(x_i, x_j) C^{\mathrm{CS}}(|x_i - x_j|)$$

- Low rank approximations, e.g. predictive processes/inducing points

$$C(x_i, x_j) = C(x_i, x^\star) C_{x^\star}^{-1} C(x^\star, x_j) + \tau^2 \delta_{ij}$$

- SPDE methods
- **Vecchia approximation**

Scaling to large datasets

Several methods have been proposed to scale Gaussian processes to large datasets, including:

- Covariance tapering

$$C(x_i, x_j) = C(x_i, x_j) C^{\mathrm{CS}}(|x_i - x_j|)$$

- Low rank approximations, e.g. predictive processes/inducing points

$$C(x_i, x_j) = C(x_i, x^\star) C_{x^\star}^{-1} C(x^\star, x_j) + \tau^2 \delta_{ij}$$

- SPDE methods
- Vecchia approximation

Scaling to large datasets

Several methods have been proposed to scale Gaussian processes to large datasets, including:

- Covariance tapering

$$C(x_i, x_j) = C(x_i, x_j) C^{\mathrm{CS}}(|x_i - x_j|)$$

- Low rank approximations, e.g. predictive processes/inducing points

$$C(x_i, x_j) = C(x_i, x^\star) C_{x^\star}^{-1} C(x^\star, x_j) + \tau^2 \delta_{ij}$$

- SPDE methods
- **Vecchia approximation**

Scaling to large datasets: Vecchia approximation

Based on the *chain rule of probability*

$$p(\mathrm{Z}) = p(Z_1)\prod_{i=2}^{n} p(Z_i|Z_{<i}) \approx p(Z_1)\prod_{i=2}^{n} p(Z_i|Z_{c(i)}),\ c(i) \subset \{< i\}$$

This provides $\Sigma^{-1} = UU'$, where U is a sparse upper triangular matrix such that

$$U_{j,i} = \begin{cases} \left(\sigma_i^2 - C_i\Sigma_{c(i)}^{-1}C_i^t\right)^{-1/2} & \text{if } i = j \\ -(\Sigma_{c(i)}^{-1}C_i)_j U_{i,i} & \text{if } j \in c(i) \\ 0 & \text{otherwise} \end{cases}$$

- $\sigma_i^2 = \mathsf{Cov}(Z(x_i), Z(x_i))$
- $C_i = \mathsf{Cov}(Z_i, Z_{c(i)})$
- $\Sigma_{c(i)} = \mathsf{Cov}(Z(x_{c(i)}), Z(x_{c(i)}))$

Scaling to large datasets: Vecchia approximation?

Low level implementation concerns

As sparse encoding (of covariance matrices) relies on non-contiguous memory accesses, it scales very poorly on parallel hardware

One solution

Application

Data generation

Simulate a stationary GP with a Matérn covariance function with $\nu = 1.5$ plus 0.05 variance noise, on 50000 uniform points (training) and on a 256 x 256 grid in $[-5, 5]^2$ and apply the transformation

1. $o + (x - o)\|x - o\|^2$, with o the origin
2. $5 \tanh(x)$,
3. $(r\cos(\theta + \frac{\pi r}{2}), r\sin(\theta + \frac{\pi r}{2}))$, in polar coordinates
4. $x + \sum_{k=0}^{2} \left(2 \cdot e^{-\frac{\|x_k - c_k\|^2}{1.6^2}} \cdot R_{\pi/2}(x_k - C_k) \right)$

- Learned model (Real NVP with 12 coupling layers, 2 hidden layers)
- Stationary model
- Transformed stationary model (fixed transformation)

Application

Data generation

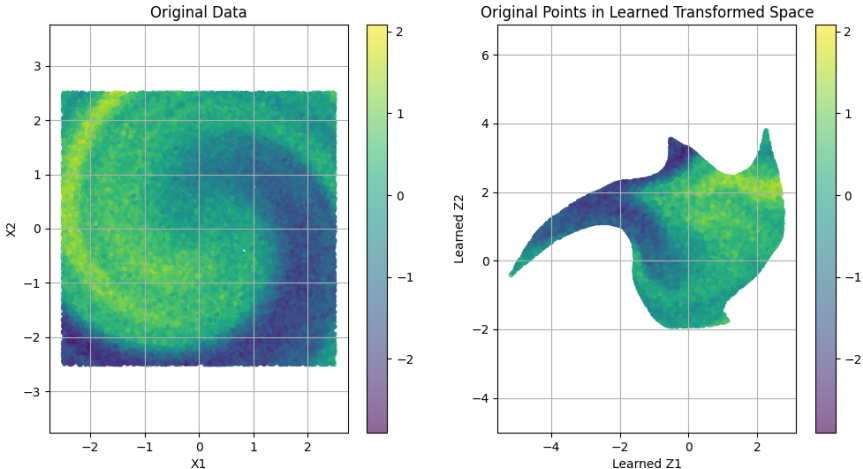
Simulate a stationary GP with a Matérn covariance function with $\nu = 1.5$ plus 0.05 variance noise, on 50000 uniform points (training) and on a 256 x 256 grid in $[-5, 5]^2$ and apply the transformation

1. $o + (x - o)\|x - o\|^2$, with o the origin
2. $5 \tanh(x)$,
3. $(r \cos(\theta + \frac{\pi r}{2}), r \sin(\theta + \frac{\pi r}{2}))$, in polar coordinates
4. $x + \sum_{k=0}^{2} \left(2 \cdot e^{-\frac{\|x_k - c_k\|^2}{1.6^2}} \cdot R_{\pi/2}(x_k - C_k) \right)$

- Learned model (Real NVP with 12 coupling layers, 2 hidden layers)
- Stationary model
- Transformed stationary model (fixed transformation)

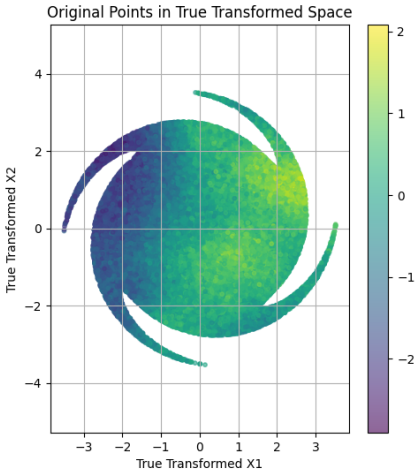
Application

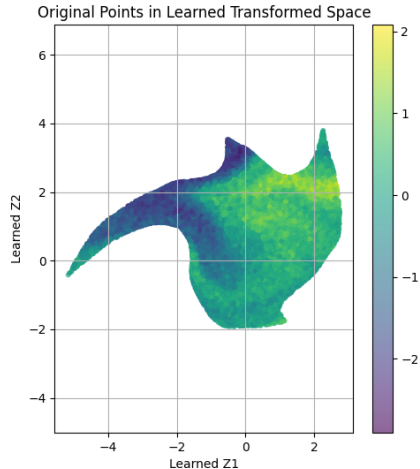
Results



Application

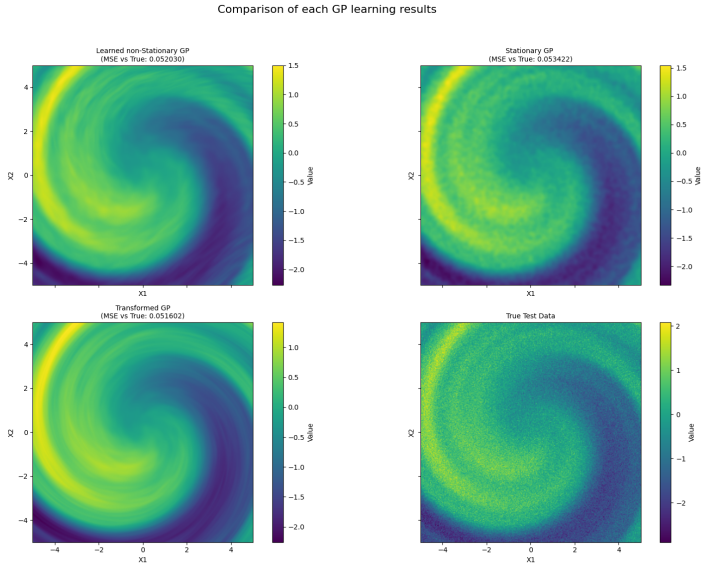
Results





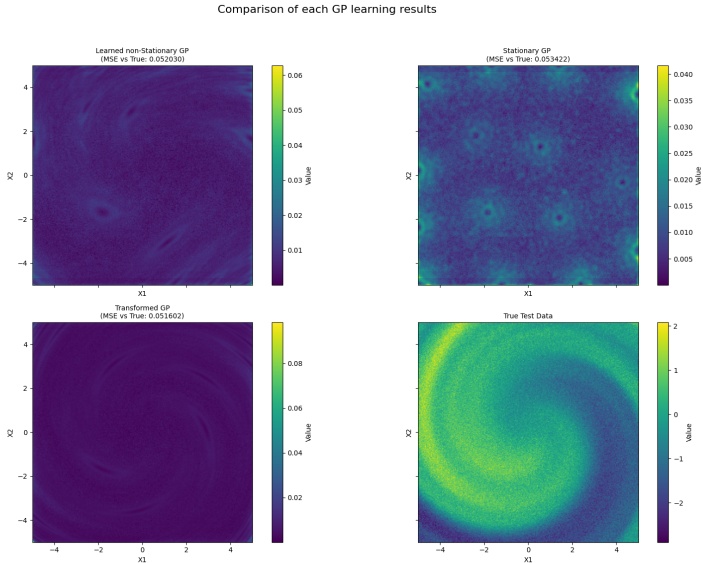
Application

Results



Application

Results



Application

Results

MSE (10^{-2}) computed on the test sets (256 x 256 grid)

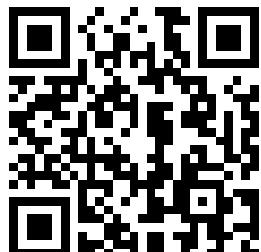
	1	2	3	4
Learned	**52.2**	**52.1**	**52.0**	**52.1**
Stationary	53.2	53.5	53.4	52.5
Transformed	52.8	52.5	51.6	52.0

Conclusion

A new look at space deformation models

- Non-stationary covariance functions and can be scaled to large datasets
- Implementation in PyTorch and GPyTorch
- Future work includes applying the framework to real-world datasets and exploring other hybrid models of the kind

Journées de Géostatistique 2025

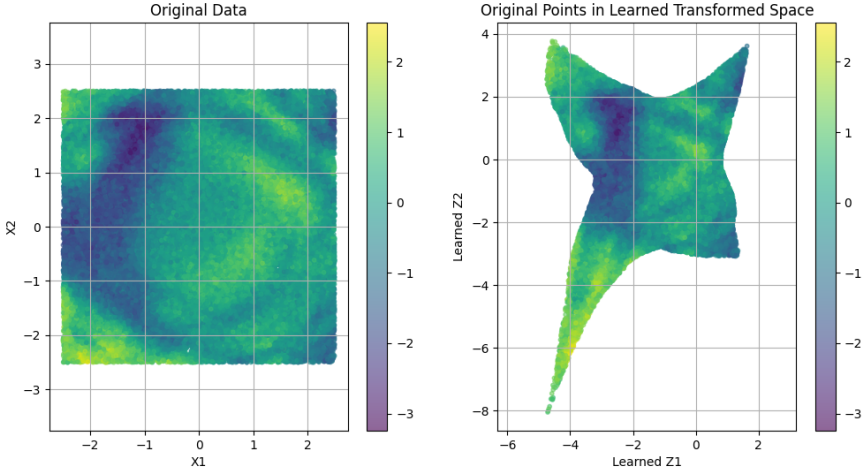


Journées de Géostatistique 2025, 3-5 Septembre 2025, Fontainebleau, France

https://geostat25.sciencesconf.org/

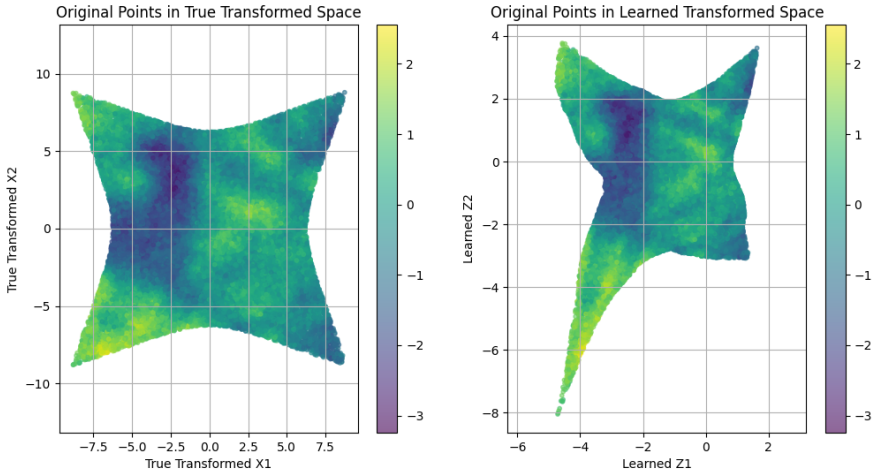
Application

Results



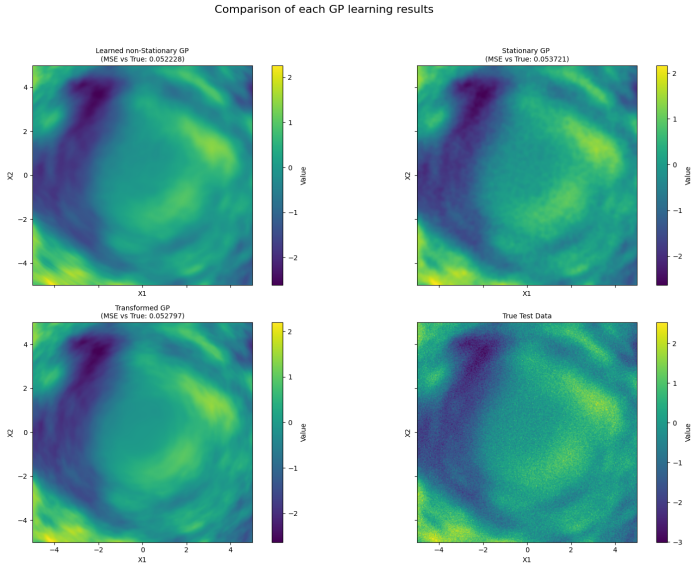
Application

Results



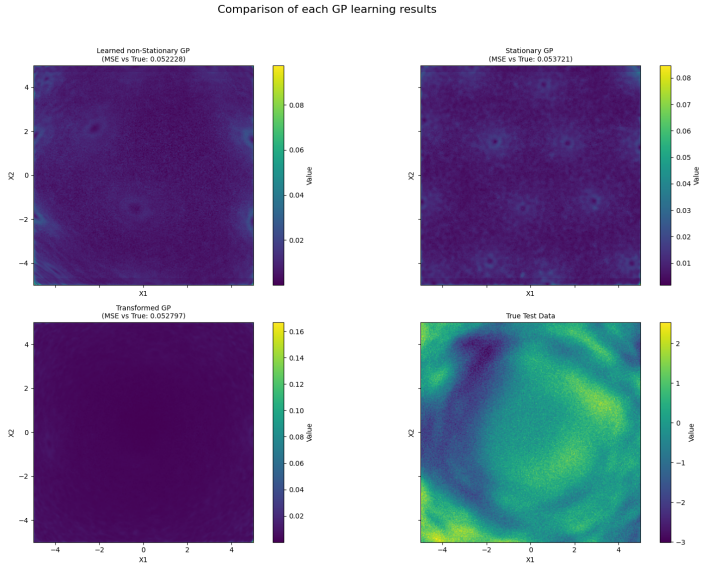
Application

Results



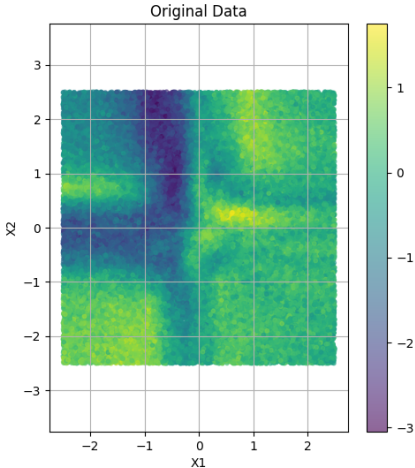
Application

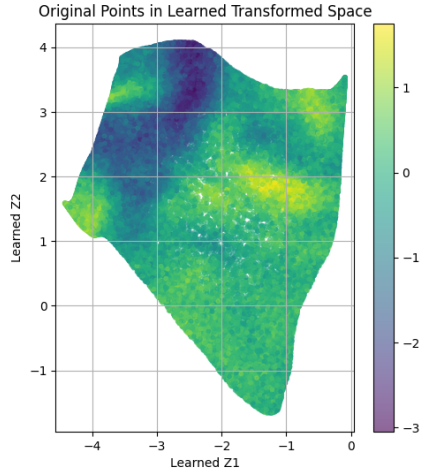
Results



Application

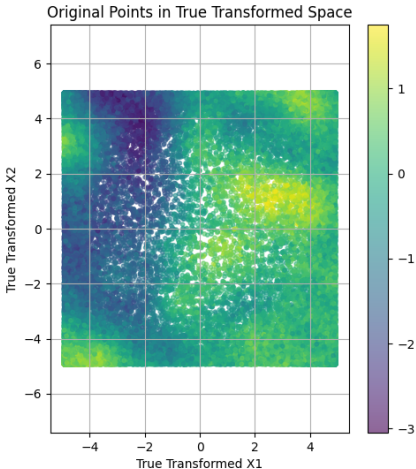
Results

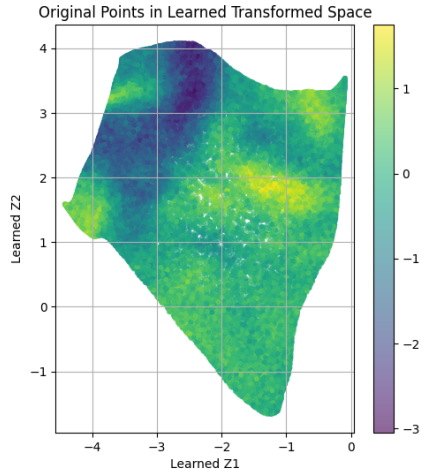




Application

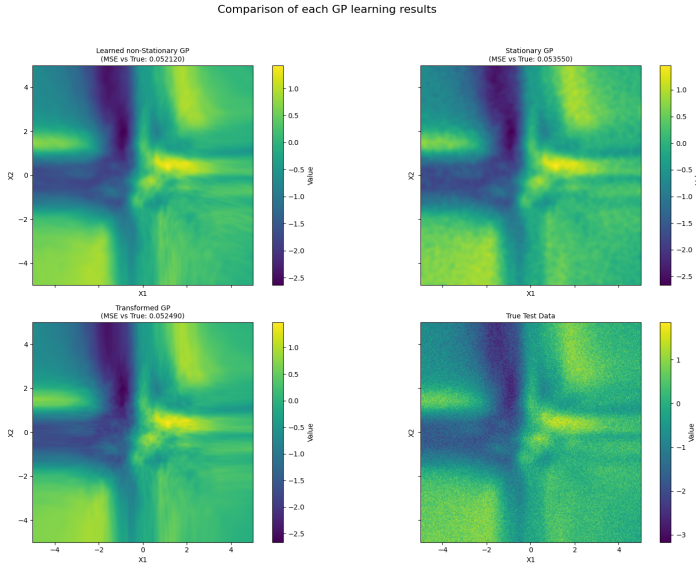
Results





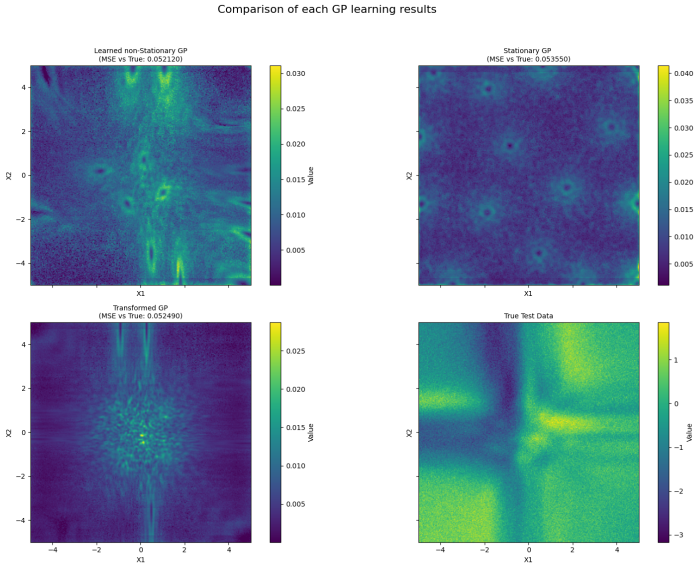
Application

Results



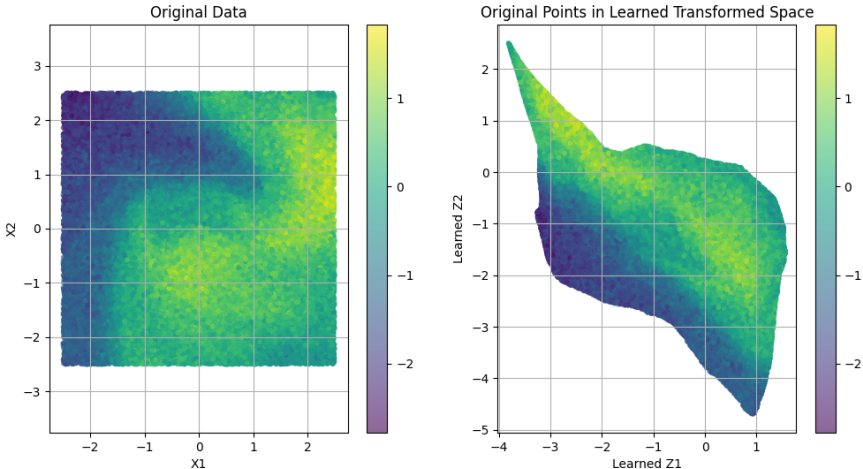
Application

Results



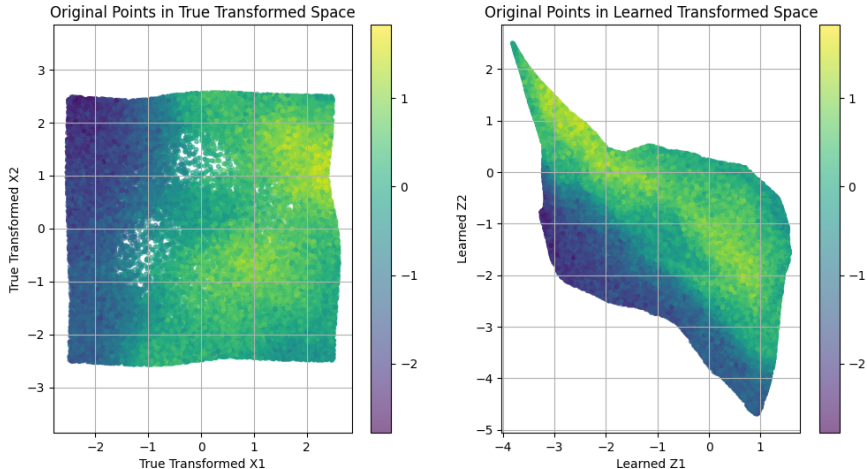
Application

Results



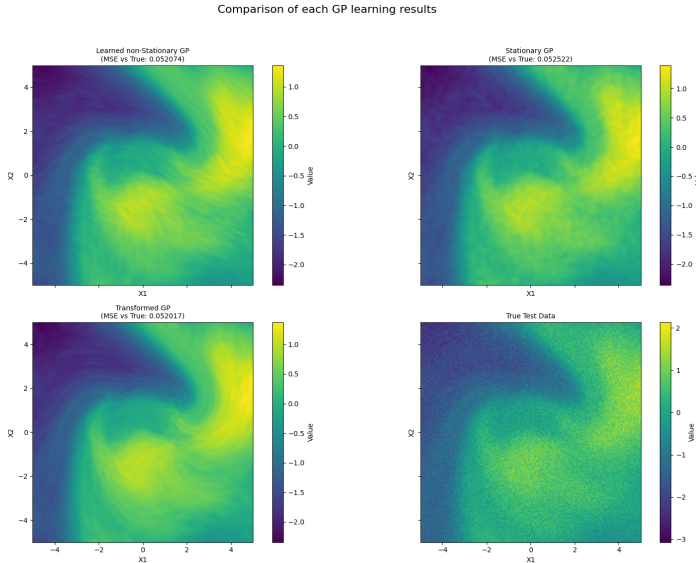
Application

Results



Application

Results



Application

Results