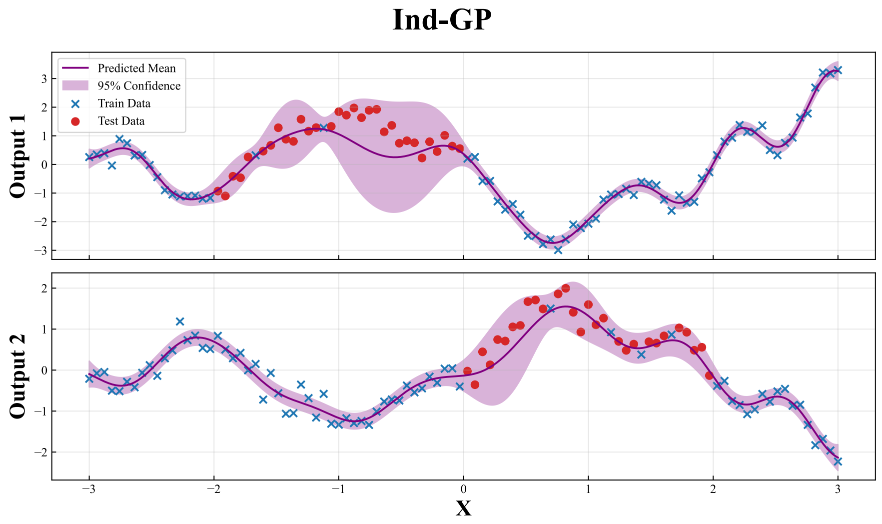
Transformed Latent Variable Multi-Output Gaussian Processes

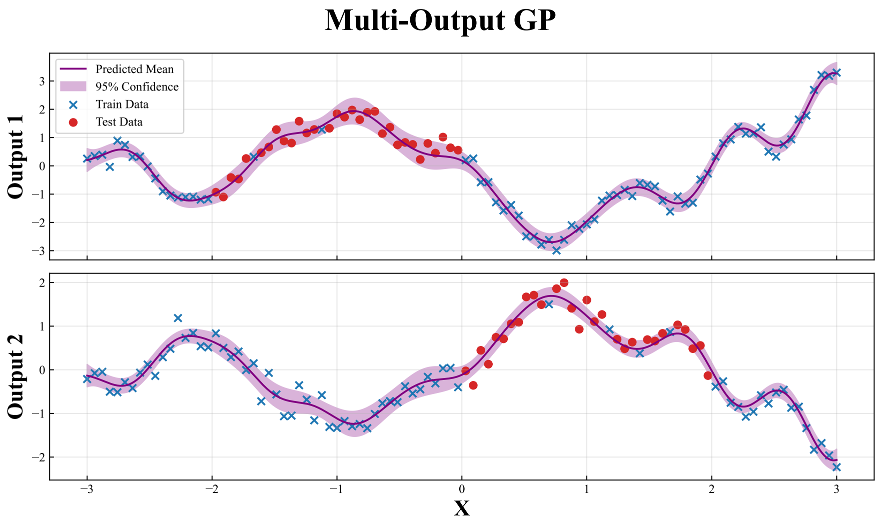
Xiaoyu Jiang, Xinxing Shi, Sokratia Georgaka, Magnus Rattray, Mauricio A Álvarez

The University of Manchester

Introduction

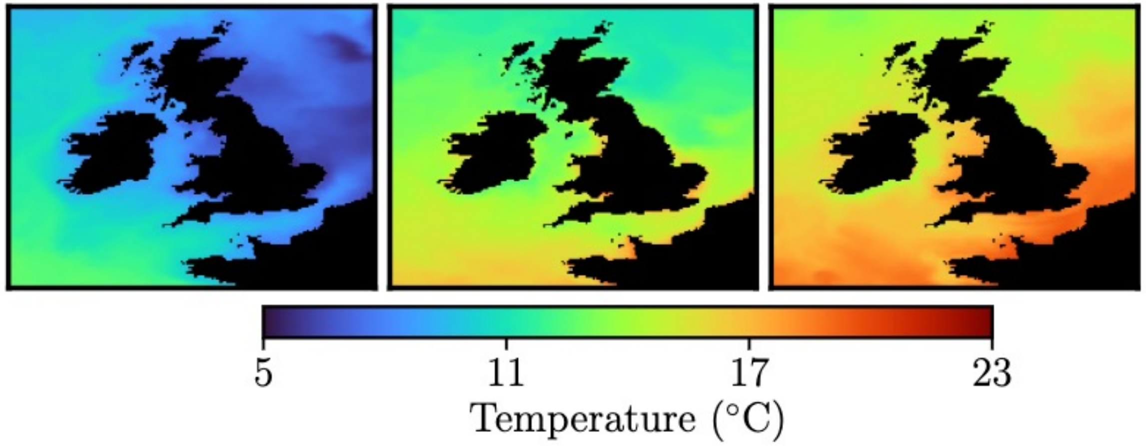
- **Gaussian Processes** (GP) are a principled and flexible probabilistic framework for learning unknown functions.
- **Multi-Output Gaussian Processes (MOGP)** extend this idea from one function to multiple correlated output functions, modelling not only input-dependent variation but also dependencies across outputs.

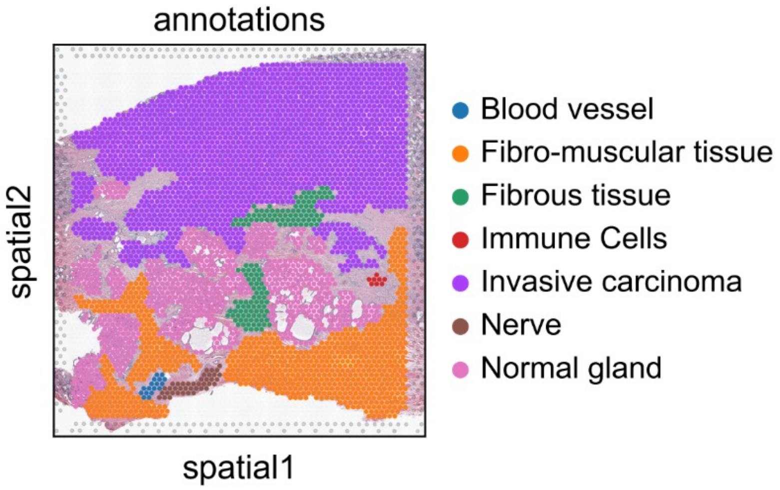




Challenges

- Many real-world dataset have **many correlated outputs**:
  - **Climate**: thousands of spatial locations;
  - **Spatial Transcriptomics**: thousands of genes;





- Inference and learning is $\boldsymbol{O(N^3P^3)}$, poor scalability when P is large.

Motivation

- Existing MOGPs reply on structural assumptions for scaling up to large output space.
  - **Low rank:** OILMM [Bruinsma et al., 2020]
  - **Grid observations:** SGPRN [Li et al., 2020]
  - **Sum-of-separable:** LMC [Van der Wilk et al., 2020], GS-LVMOGP [Jiang et al., 2025]

- Scalability techniques have been well studied in single-output GP literature.

⬇

💡 Propose a multi-output deep kernel without imposing explicit restrictive assumptions.
💡 Reformulates MOGPs as scalar GPs defined on a learned embedding space.
💡 Typical scalable GP techniques can be seamlessly reused to improve the scalability with respect to the number of outputs.

Bruinsma, W., Perim, E., Tebbutt, W., Hosking, S., Solin, A., and Turner, R., Scalable exact inference in multi-output Gaussian processes, **ICML 2020**
Li, S., Xing, W., Kirby, R. M., and Zhe, S., Scalable Gaussian process regression networks, **IJCAI 2020**
Jiang, X., Georgaka, S., Rattray, M., and Álvarez, M., Scalable multi-output Gaussian processes with stochastic variational inference, **TMLR 2025**

Our method: Key Idea

- In **LVMOGP** [Dai et al., 2017], a latent variable is assigned to each output, functioning as a "representation" for the output function.

- Building on LVMOGP, we learn an **embedding space** (through a trainable function Φ_θ) where a standard scalar GP kernel can capture flexible cross-output dependencies.

$$(x_n, \mathbf{h}_p) \xmapsto{\Phi_\theta} \tilde{x}_{n,p} \in \mathbb{R}^{D_T},$$

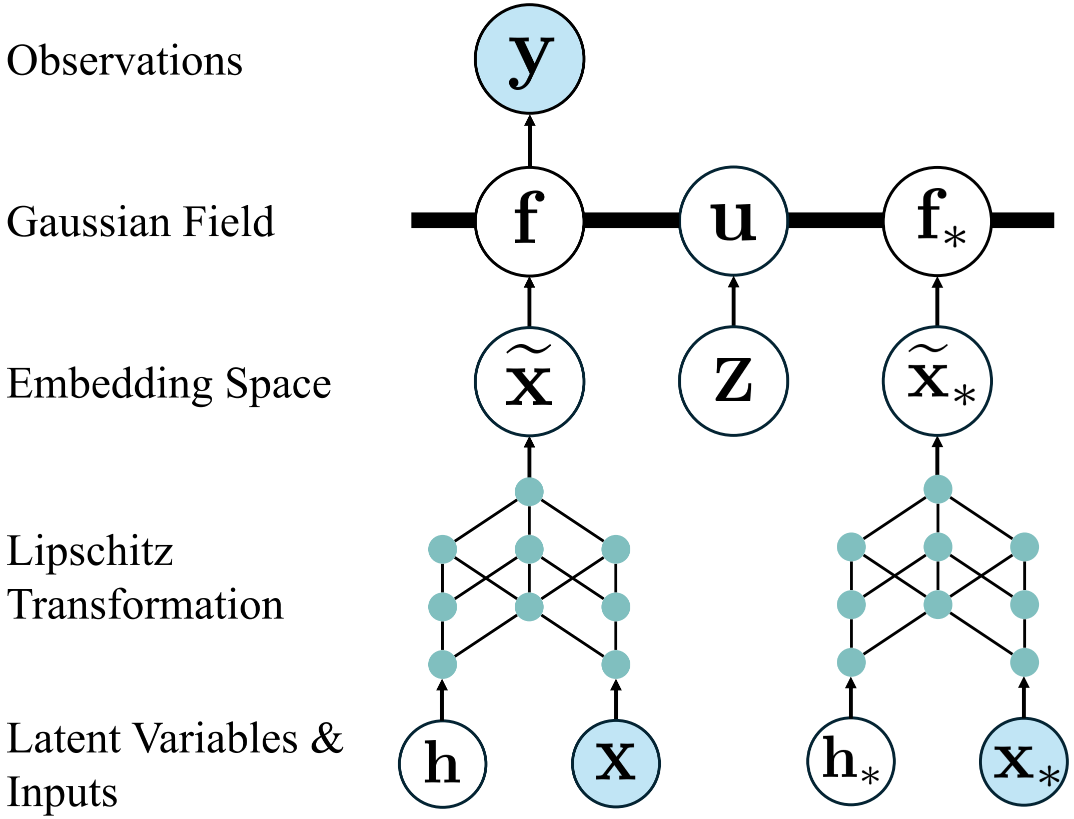
$$\operatorname{cov}\left[f_p\left(x_n\right), f_{p'}\left(x_{n'}\right)\right] = k_{\text{base}}\left(\tilde{x}_{n,p}, \tilde{x}_{n',p'}\right).$$

Dai, Z., Álvarez, M., and Lawrence, N, Efficient modeling of latent information in supervised learning using Gaussian processes, **NeurIPS 2017**

Our method: T-LVMOGP

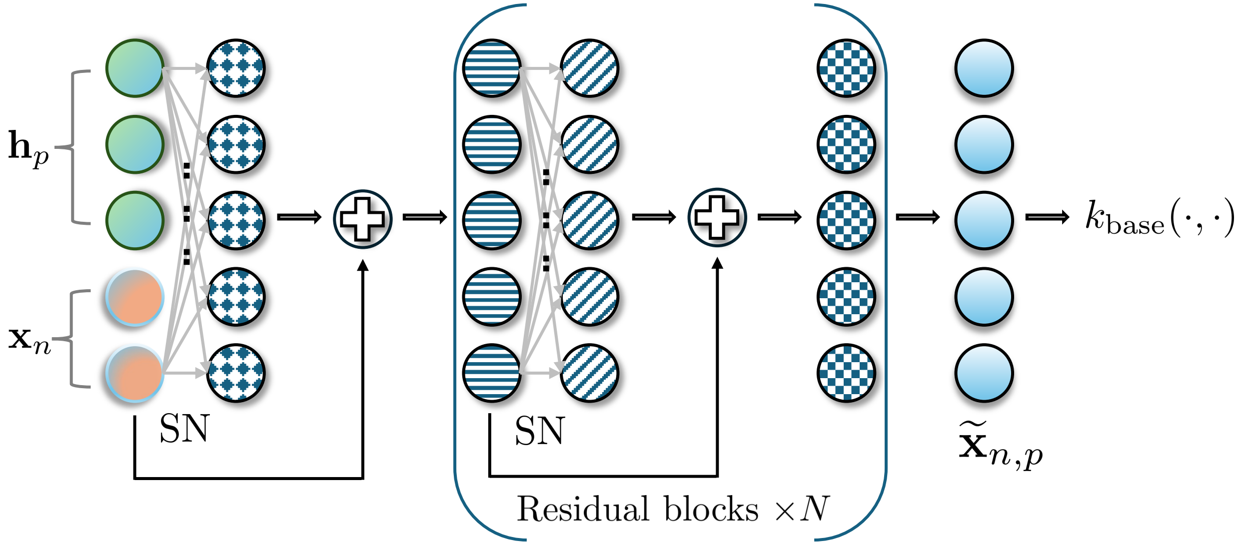
- **SVGP** with mini-batch training, inducing points are placed in the embedding space.
- **Lipschitz** neural network.

$$\mathcal{L}_3 = \sum_{n=1}^{N}\sum_{p=1}^{P}\mathbb{E}_{q(\mathbf{h}_p)q(f_p(\mathbf{x}_n)|\mathbf{x}_n,\mathbf{h}_p)}\left[\log p(y_{n,p} \mid f_p(\mathbf{x}_n))\right] - \mathrm{KL}[q(\mathbf{u}) \,||\, p(\mathbf{u})] - \sum_{p=1}^{P}\mathrm{KL}[q(\mathbf{h}_p) \,||\, p(\mathbf{h}_p)].$$



Lipschitz Neural Network

- To alleviate the overfitting issue of deep kernel: **spectral normalisation**, **residual connection**.
- By setting a proper **SN-UB**, this neural network is **Lipschitz continuous**.



Experiments: EEG

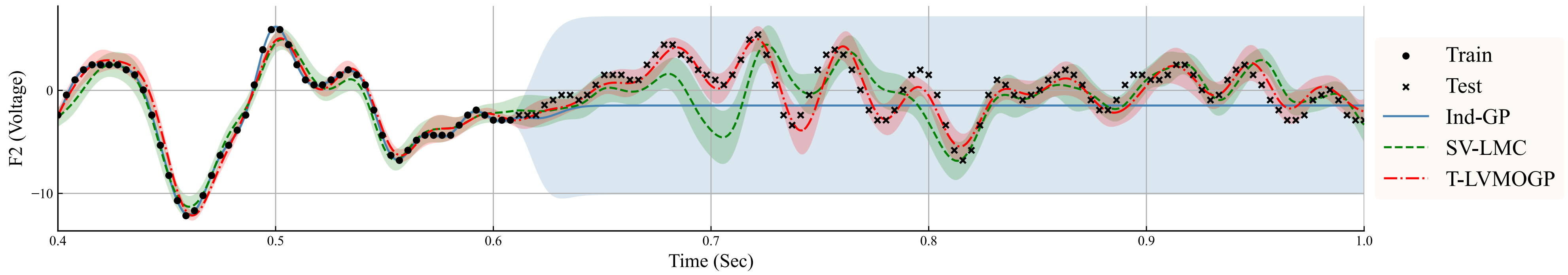


Table 1. Performance of different models on the EEG dataset. The ↓ symbol indicates that lower values are better.

Model	**MSE ↓**	**NLL ↓**
Baselines		
Ind-GP	0.466 ± 0.000	1.227 ± 0.000
SGPRN	0.473 ± 0.003	7.105 ± 2.110
G-MOGP	0.604 ± 0.099	1.337 ± 0.038
OILMM	0.372 ± 0.098	0.979 ± 0.159
GS-LVMOGP	0.366 ± 0.064	0.924 ± 0.081
SV-LMC	0.282 ± 0.193	0.857 ± 0.613
T-LVMOGP (ours)	$\mathbf{0.115 \pm 0.025}$	$\mathbf{0.814 \pm 0.310}$

Table 2. Performance of T-LVMOGP with different SN-UB values on the EEG experiment.

SN-UB	**MSE ↓**	**NLL ↓**
0.1	0.128 ± 0.087	1.339 ± 2.065
0.005	$\mathbf{0.115 \pm 0.025}$	$\mathbf{0.814 \pm 0.310}$
0.001	0.145 ± 0.053	1.371 ± 1.120

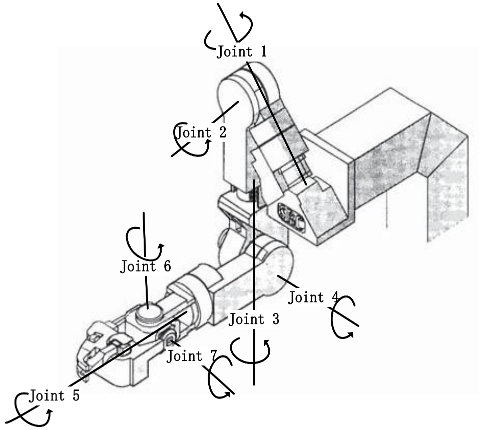
Experiments: SARCOS

Table 3. Performance comparison of different models on the SARCOS dataset.

Model	MSE ↓	NLL ↓	Training Time (s/epoch) ↓
Baselines			
OILMM (Bruinsma et al., 2020)	0.140 ± 0.010	0.864 ± 0.090	8.951 ± 0.070
GS-LVMOGP (Jiang et al., 2025)	0.037 ± 0.001	-0.220 ± 0.011	10.241 ± 0.096
SV-LMC (Van der Wilk et al., 2020)	0.033 ± 0.000	-0.297 ± 0.003	6.185 ± 0.086
G-MOGP (Dai et al., 2024)	0.023 ± 0.001	-0.483 ± 0.017	5.892 ± 0.084
T-LVMOGP (ours)	$\mathbf{0.022 \pm 0.000}$	$\mathbf{-0.485 \pm 0.009}$	$\mathbf{5.263 \pm 0.194}$

Table 4. Performance of T-LVMOGP with different SN-UB values on the SARCOS experiment.

SN-UB	MSE ↓	NLL ↓
1.5	0.023 ± 0.001	-0.470 ± 0.016
1.0	$\mathbf{0.022 \pm 0.000}$	$\mathbf{-0.485 \pm 0.009}$
0.1	0.028 ± 0.000	-0.363 ± 0.005



Experiments: ERA5

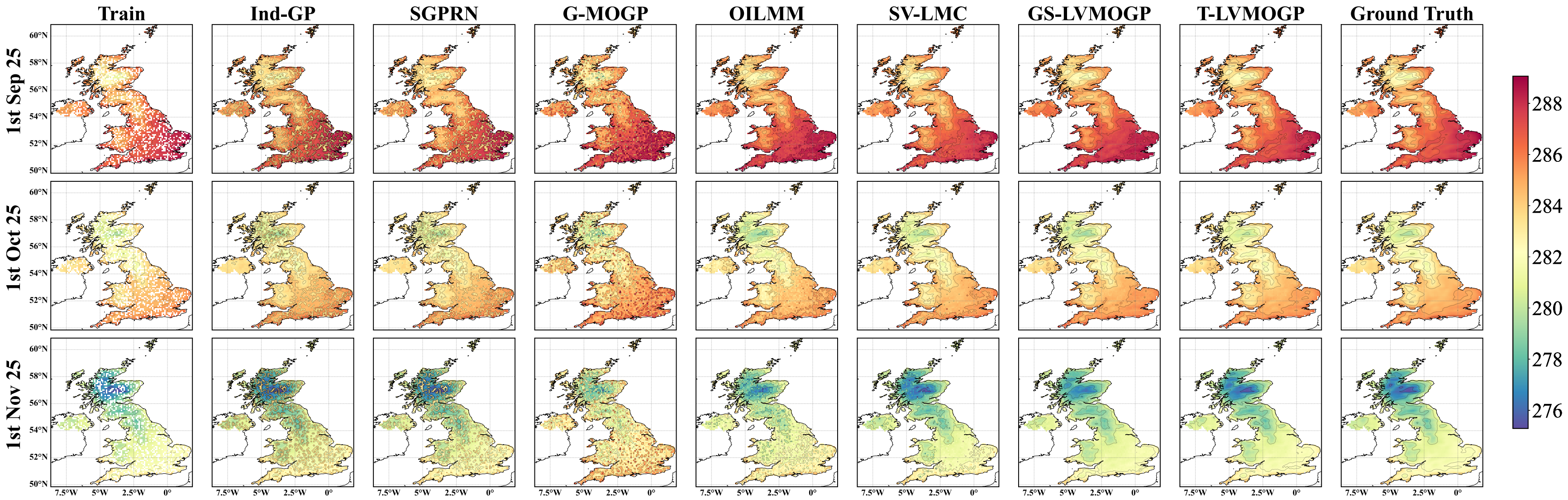


Table 5. Performance comparison of different models on the ERA5 dataset with random splitting.

Model	**MSE ↓**	**NLL ↓**	**Training Time (s/epoch) ↓**
Baselines			
Ind-GP (Hensman et al., 2013)	0.997 ± 0.000	1.418 ± 0.000	31.518 ± 0.366
SGPRN (Li et al., 2020)	0.897 ± 0.124	51.005 ± 7.411	0.780 ± 0.001
G-MOGP (Dai et al., 2024)	0.316 ± 0.035	0.910 ± 0.058	0.685 ± 0.018
OILMM (Bruinsma et al., 2020)	0.123 ± 0.052	0.483 ± 0.403	2.332 ± 0.001
SV-LMC (Van der Wilk et al., 2020)	0.012 ± 0.004	0.758 ± 0.428	$\mathbf{0.570 \pm 0.015}$
GS-LVMOGP (Jiang et al., 2025)	0.014 ± 0.003	-0.699 ± 0.084	1.323 ± 0.028
T-LVMOGP (ours)	$\mathbf{0.002 \pm 0.000}$	$\mathbf{-1.564 \pm 0.024}$	0.708 ± 0.013

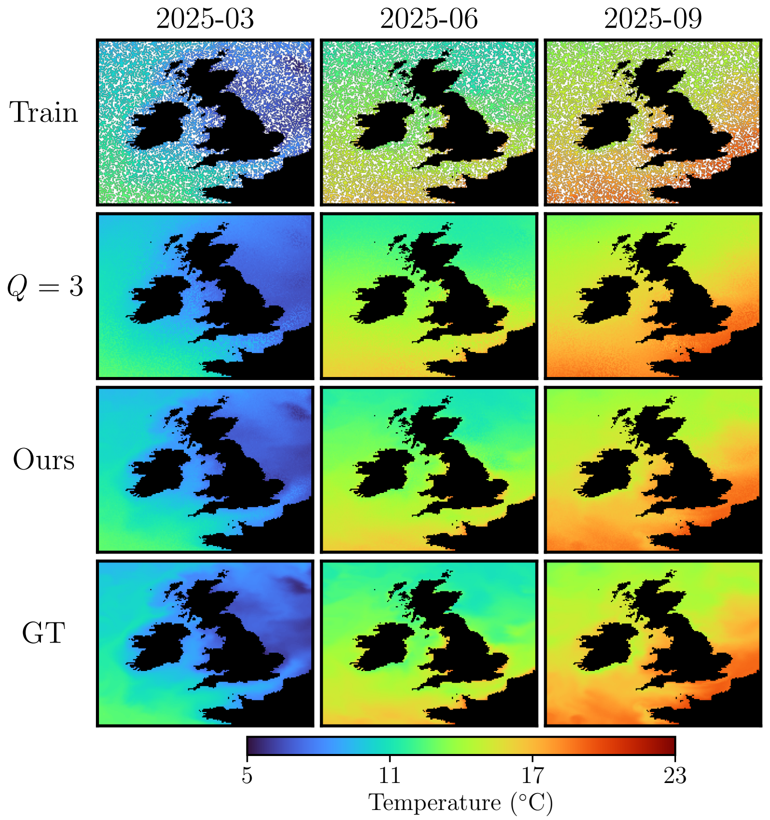
Experiments: CM and ST

Table 6. Performance comparison on the Copernicus Marine dataset for the output extrapolation task.

	GS-LVMOGP (Jiang et al., 2025)			**T-LVMOGP** (ours)
Metric	Q=1	Q=2	Q=3	RCNN
MSE ↓	0.040 ± 0.002	0.036 ± 0.001	0.035 ± 0.002	$\mathbf{0.029 \pm 0.000}$
NLL ↓	5.230 ± 0.640	5.086 ± 0.297	4.975 ± 0.923	$\mathbf{-0.439 \pm 0.011}$
Training Time (s/epoch) ↓	1.321 ± 0.043	1.750 ± 0.045	2.076 ± 0.048	$\mathbf{1.230 \pm 0.012}$

Table 7. Predictive performance of GS-LVMOGP and T-LVMOGP on a Spatial Transcriptomics dataset.

	GS-LVMOGP (Jiang et al., 2025)			**T-LVMOGP** (ours)
Metric	Q=1	Q=2	Q=3	RCNN
MSE ↓	11.616 ± 0.773	11.396 ± 0.314	11.024 ± 0.350	$\mathbf{9.189 \pm 0.342}$
NLL ↓	0.677 ± 0.002	0.675 ± 0.001	$\mathbf{0.674 \pm 0.001}$	$\mathbf{0.674 \pm 0.001}$
Training Time (s/epoch) ↓	$\mathbf{65.507 \pm 0.820}$	73.639 ± 0.505	78.245 ± 1.131	73.047 ± 0.318



Experiments: Ablation study and tighter variational bounds

Table 8. Test NLL comparison of T-LVMOGP with and without spectral normalisation.

Dataset	w/o SN	w/ SN
EEG	4.109 ± 2.141	$\mathbf{0.814 \pm 0.310}$
SARCOS	0.112 ± 0.856	$\mathbf{-0.485 \pm 0.01}$
ERA5 (random)	-1.401 ± 0.13	$\mathbf{-1.564 \pm 0.03}$
ERA5 (block-wise)	-0.895 ± 0.85	$\mathbf{-1.503 \pm 0.03}$
Copernicus Marine	-0.400 ± 0.08	$\mathbf{-0.439 \pm 0.01}$

Table 9. Test NLL comparison of T-LVMOGP with and without neural network architecture.

Dataset	w/o NN	w/ NN
EEG	1.153 ± 0.437	$\mathbf{0.814 \pm 0.310}$
SARCOS	-0.336 ± 0.01	$\mathbf{-0.485 \pm 0.01}$
ERA5 (random)	-1.554 ± 0.03	$\mathbf{-1.564 \pm 0.03}$
ERA5 (block-wise)	-1.474 ± 0.06	$\mathbf{-1.503 \pm 0.03}$
Copernicus Marine	-0.420 ± 0.01	$\mathbf{-0.439 \pm 0.01}$

Table 10. Test NLL comparison of T-LVMOGP with standard and tighter variational bounds. †: for non-Gaussian likelihood, tighter bounds might not be effectively tighter in practice; more explanations are provided in Appendix F.7.

Dataset	Standard	Tighter
SARCOS	-0.485 ± 0.01	$\mathbf{-0.502 \pm 0.01}$
Copernicus Marine	-0.439 ± 0.01	$\mathbf{-0.443 \pm 0.01}$
Spatial Transcriptomics†	0.674 ± 0.001	0.674 ± 0.002

Conclusion

- Flexible **multi-output deep kernels** that scale MOGPs to high-dimensional outputs.
- A **generalisation** of GS-LVMOGP, but without explicit restrictive structural assumptions.
- **Stochastic Variational Inference** procedure within the SVGP framework, mini-batch training over both inputs and outputs.
- Naturally accommodates **non-Gaussian likelihoods** and **tighter variational bounds**.
- The code is open-sourced: https://github.com/XiaoyuJiang17/T-LVMOGP-official