The Gaussian Process Autoregressive Regression Model (GPAR)

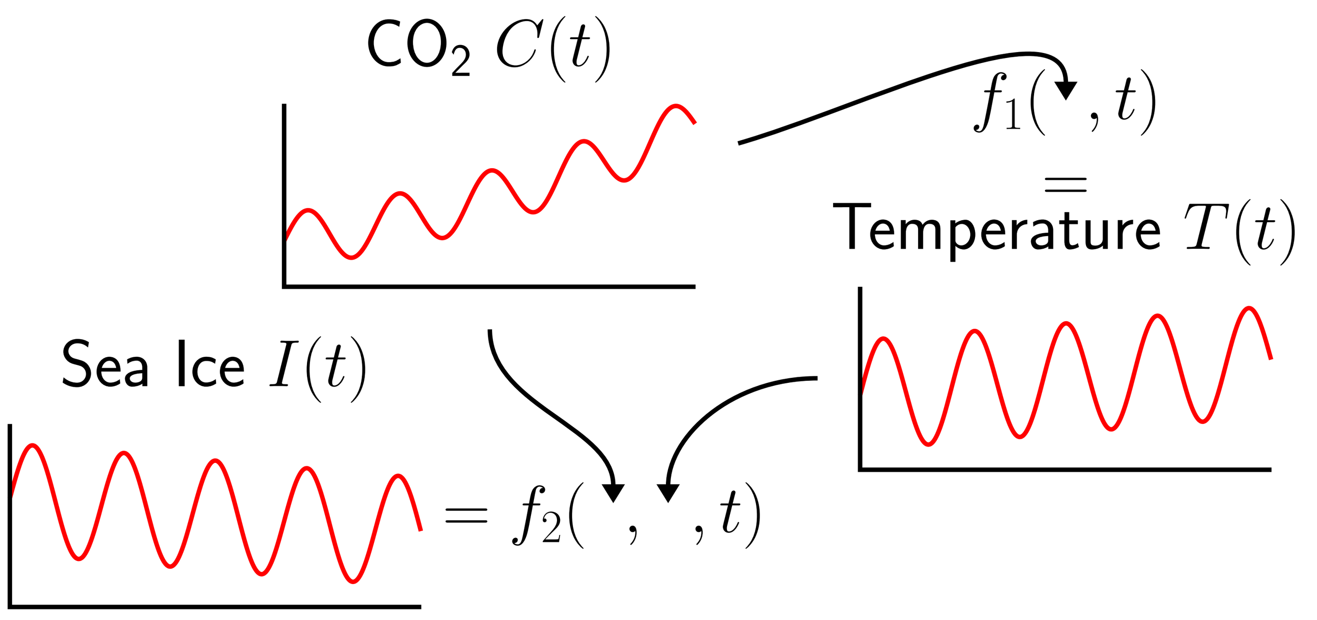
James Requeima[1,2], Will Tebbutt[1,2], Wessel Bruinsma[1,2], and Richard E. Turner[1]

[1]University of Cambridge and [2]Invenia Labs, Cambridge, UK

{jrr41, wct23, wpb23, ret26}@cam.ac.uk

- Multi-output Gaussian processes (MOGP): typically **computationally demanding** and **limited representational power**.
- GPAR is a **scalable** MOGP able to **capture nonlinear, possibly input-varying, dependencies between outputs**.
- Construction is **simple**: product rule decomposes the joint distribution over outputs; model conditionals with **standard GPs**.

Motivation



$$p(I(t), T(t), C(t)) = p(C(t)) \underbrace{p(T(t)\,|\,C(t))}_{\text{models } f_1} \underbrace{p(I(t)\,|\,T(t), C(t))}_{\text{models } f_2}.$$

GPAR

Use product rule to decompose joint distribution over outputs:

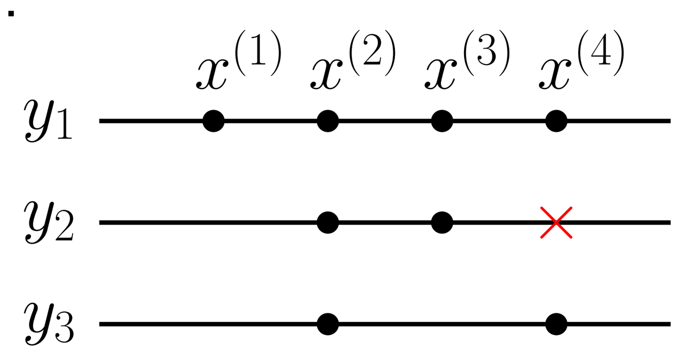
$$p(y_{1:M}(x)) = p(y_1(x)) \underbrace{p(y_2(x)\,|\,y_1(x))}_{\substack{y_2(x) \text{ as a random} \\ \text{function of } y_1(x)}} \cdots \underbrace{p(y_M(x)\,|\,y_{1:M-1}(x))}_{\substack{y_M(x) \text{ as a random} \\ \text{function of } y_{1:M-1}(x)}}.$$

Model conditionals with standard GPs:

$$\begin{aligned} y_1(x) &= f_1(x), & f_1 &\sim \mathcal{GP}(0, k_1), \\ y_2(x) &= f_2(y_1(x), x), & f_2 &\sim \mathcal{GP}(0, k_2), \\ &\vdots \\ y_M(x) &= f_M(y_{M-1}(x), \ldots, y_1(x), x) & f_M &\sim \mathcal{GP}(0, k_M). \end{aligned}$$

Inference and Learning

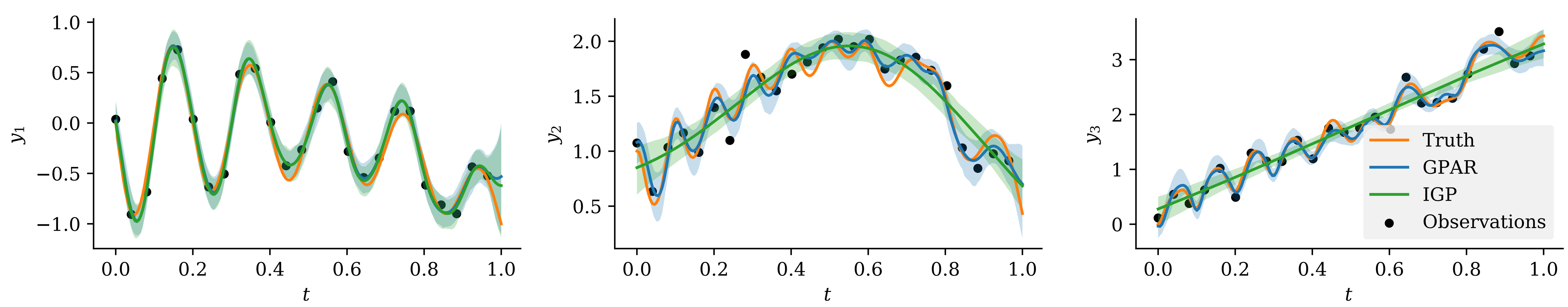
- *Definition:* Call a data set $\mathcal{D}$ **closed downwards** if $y_i^{(n)}(x^{(n)}) \in \mathcal{D}$ implies that $y_j^{(n)}(x^{(n)}) \in \mathcal{D}$ for all $j < i$.
- If observed data is closed downwards, inference and learning **decouple** into **single-output problems**:



$$p(f_{1:M}\,|\,(y_{1:M}^{(n)}, x^{(n)})_{n=1}^N) = \prod_{m=1}^M p(f_m\,|\,\underbrace{(y_m^{(n)})_{n=1}^N}_{\substack{\text{observations} \\ \text{for } f_m}}, \underbrace{(y_{1:m-1}^{(n)}, x^{(n)})_{n=1}^N}_{\substack{\text{input locations} \\ \text{of observations}}}).$$

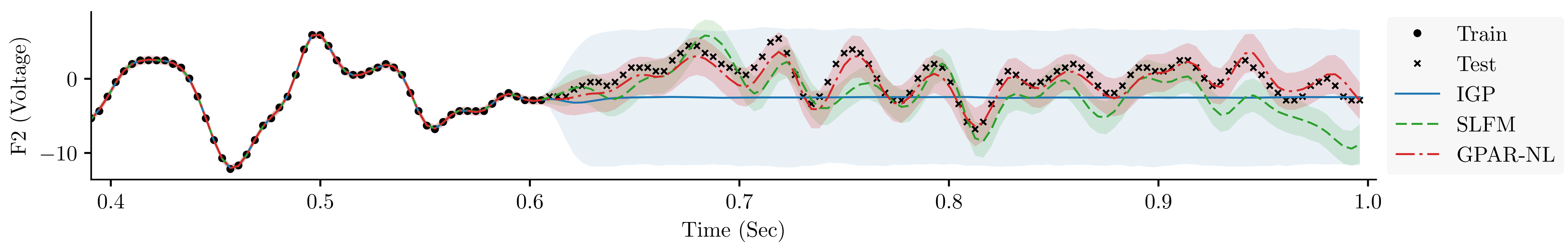
- GPAR is trivially compatible with off-the-shelf GP scaling techniques [1] to scale to **large numbers of data points**.
- Two deficiencies: unable to handle **noisy** and **missing** data. Simple approximations possible and empirically effective.

Synthetic Data



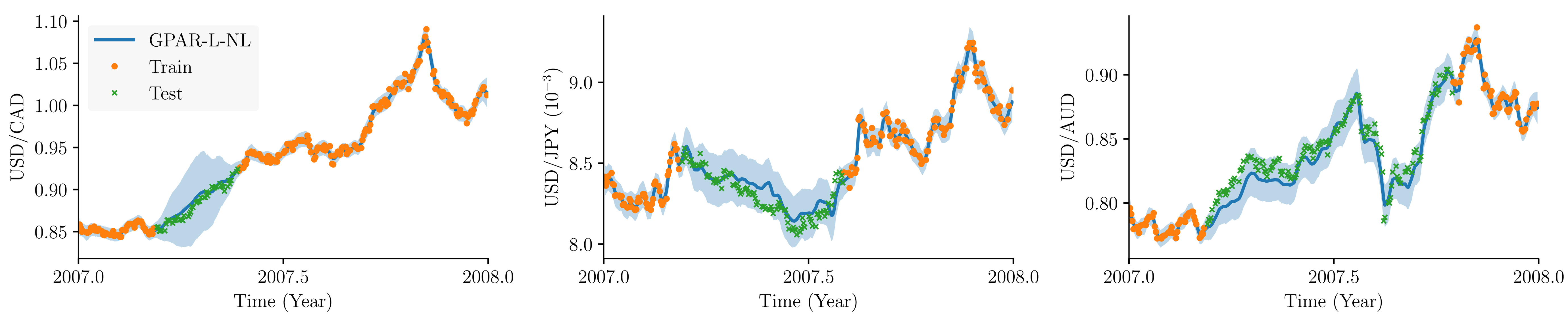
$$\begin{aligned} y_1(t) &= -\frac{\sin(10\pi(t+1))}{2t+1} \\ &\quad - t^4 + \varepsilon_1, \\ y_2(t) &= \cos^2(y_1(t)) \\ &\quad + \sin(3t) + \varepsilon_2, \\ y_3(t) &= y_2(t) y_1^2(t) \\ &\quad + 3t + \varepsilon_3. \end{aligned}$$

Experiment: Electroencephalogram (EEG) Data Set



Model	SMSE
IGP	1.75
SLFM	1.06
GPAR-NL	**0.26**

Experiment: Exchange Rates Data Set



Model	SMSE
IGP	0.60
CMOGP	0.24
CGP	0.21
GPAR-L-NL	**0.03**

Python: `https://github.com/wesselb/gpar`. Julia: `https://github.com/willtebbutt/GPAR.jl`.

[1] Titsias, M. K. (2009). Variational learning of inducing variables in sparse Gaussian processes. Artificial Intelligence and Statistics, 12:567–574.